Support Vector Machines and Reinforcement Learning

Dr. Peter C. Y. Chen

Associate Professor
Department of Mechanical Engineering
Faculty of Engineering
National University of Singapore
Email: mpechenp@nus.edu.sg
https://sites.google.com/view/peter-chen/home

These lecture slides were prepared using LaTex and the *Beamer* class

- Schedule (Dates are for info only. All lectures now in e-learning mode)

Lect.	*Subject*	*Date*	*Topic*
1	Support	11/3	Introduction. Margins. Primal problem
2	Vector	18/3	Dual problems. Soft margin
3	Machines	25/3	Kernels. Demo for SVM project
4	Reinforcement	01/4	Introduction. Markov processes
5	Learning	08/4	Dynamic programming
6		15/4	Q-Learning. Demo for RL project

- Assessment
  - Projects for Part II (20% CA) – Reports due on 23/04/2021
  - Other assessment – Check latest announcement
- Material and Reference
  - This set of lecture slides
  - Supplementary notes
  - Haykin, Neural Networks, Prentice-Hall, 3rd ed, 2008
  - Sutton and Barto, Reinforcement Learning, MIT Press, 2nd ed., 2018

Support vector machines

Vladimir N. Vapnik

U. of London

Reinforcement learning

Richard Bellman

1920-1984

Richard Sutton

U. of Alberta

Andrew Barto

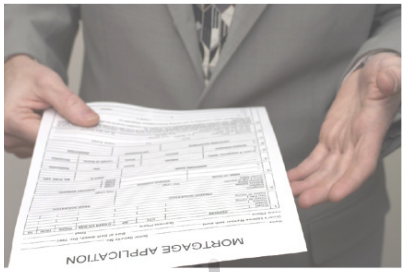
U. of Massachusetts

Chris Watkins

U. of London

SVM: Processing mortgage loan application as a classification problem

Application



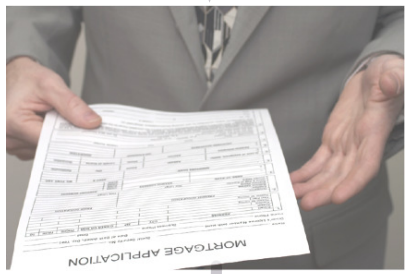
Evaluation

- age
- income
- education
- marital status
- debt
- etc

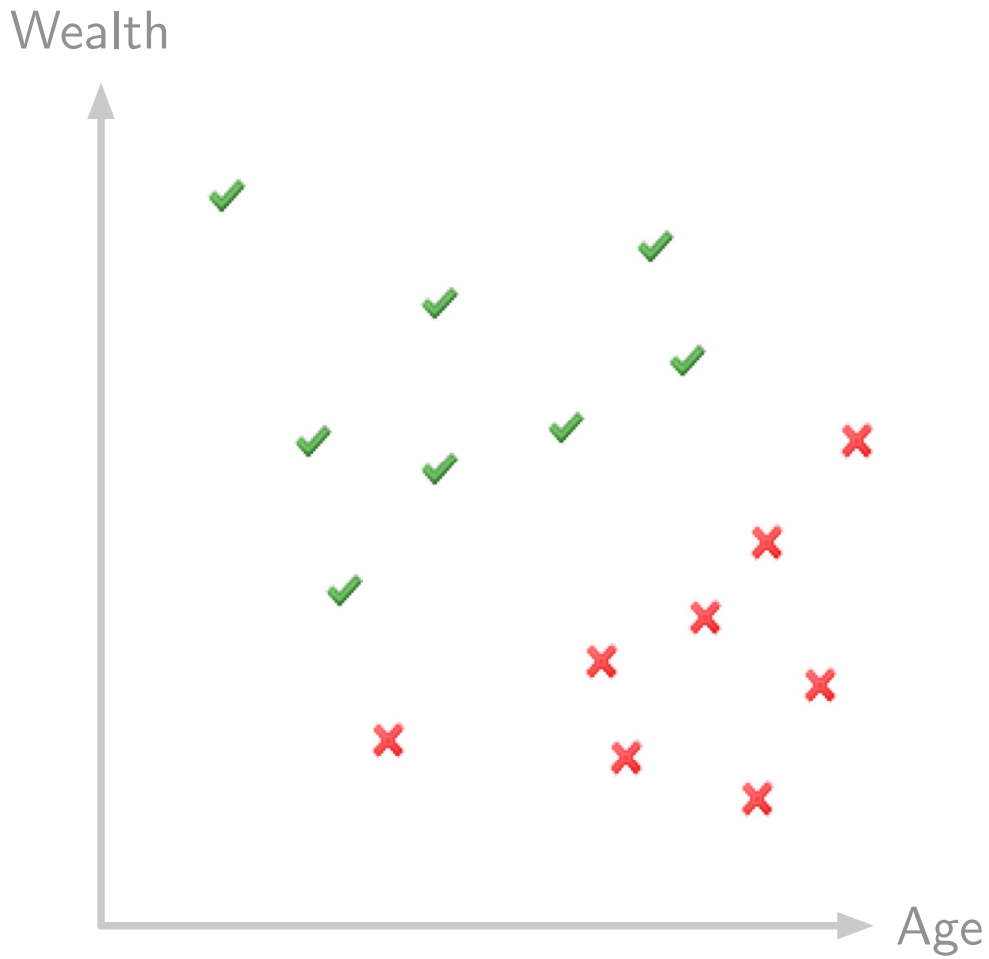
Decision

✔ Approve ✘ Reject

Mortgage loan application: Use historical data with decisions made by human



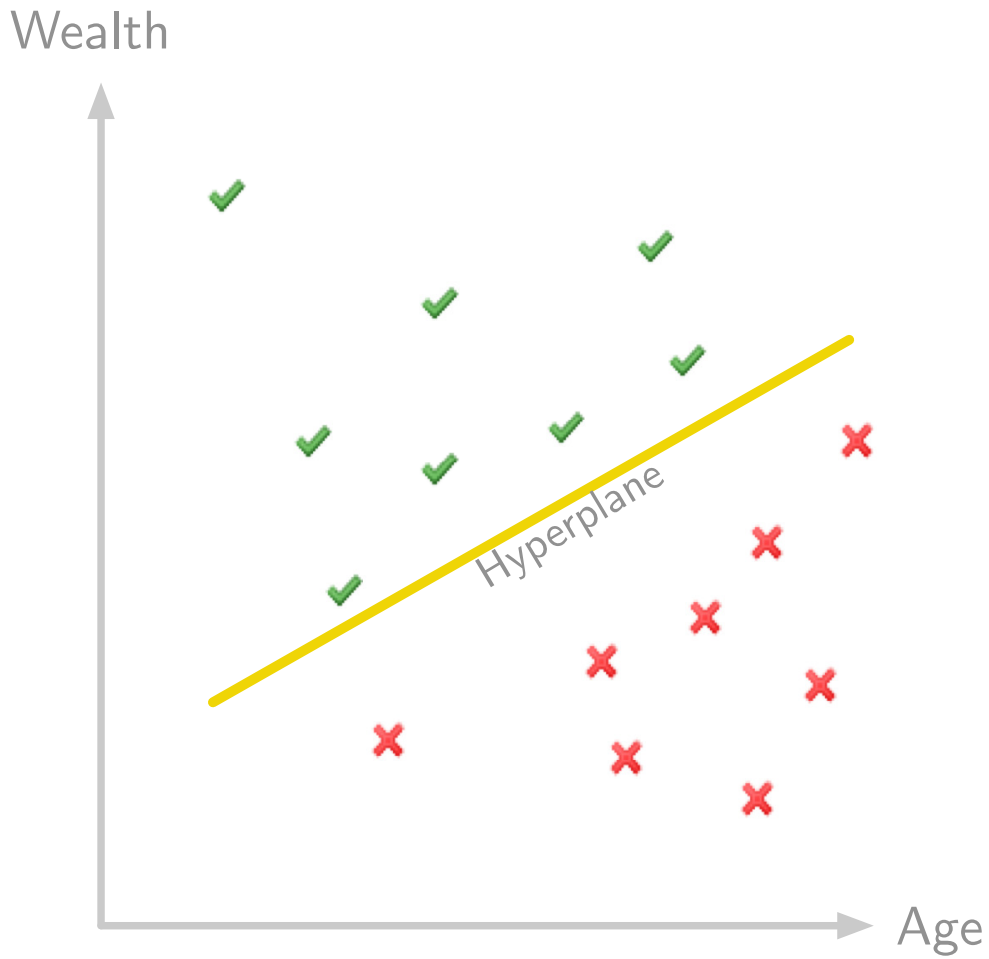
✓ Approve ✗ Reject



Mortgage loan application: Use AI system to "learn" human decisions

AI-based System

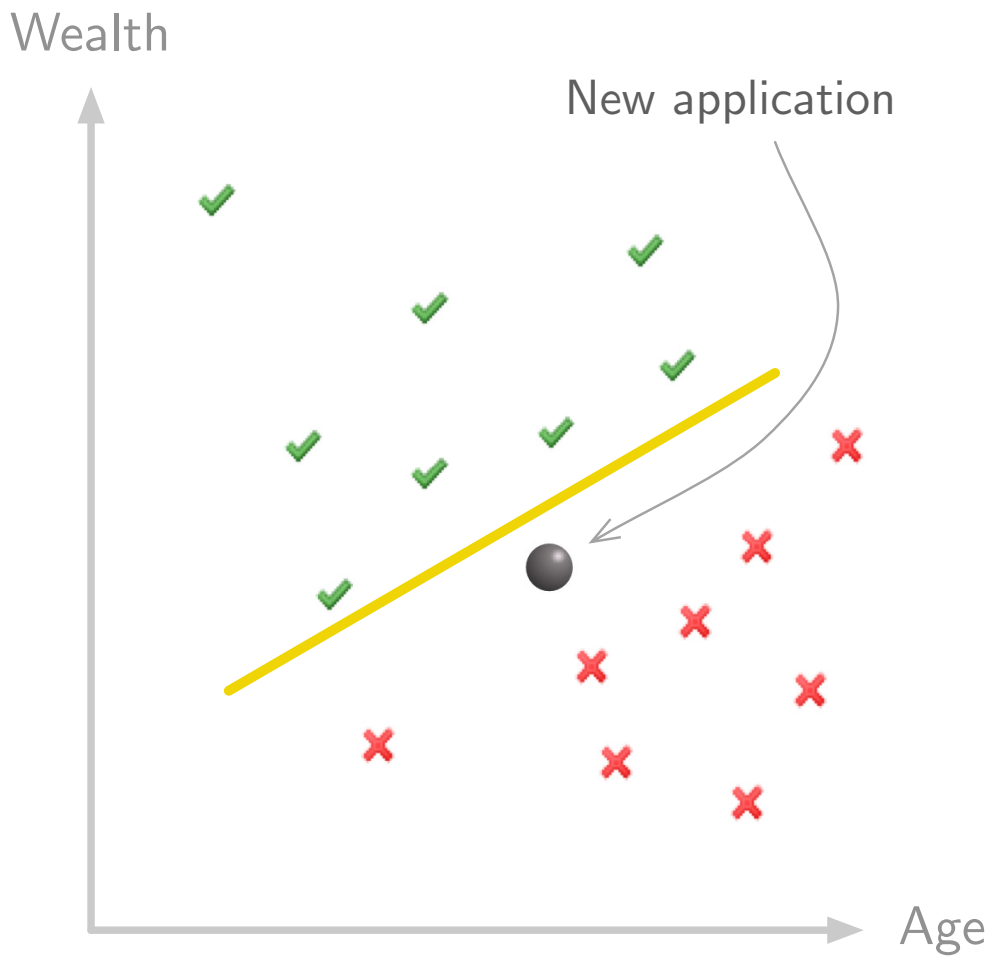
✓ Approve ✘ Reject



Mortgage loan application: Trained system (i.,e., weights found and fixed) generalizes

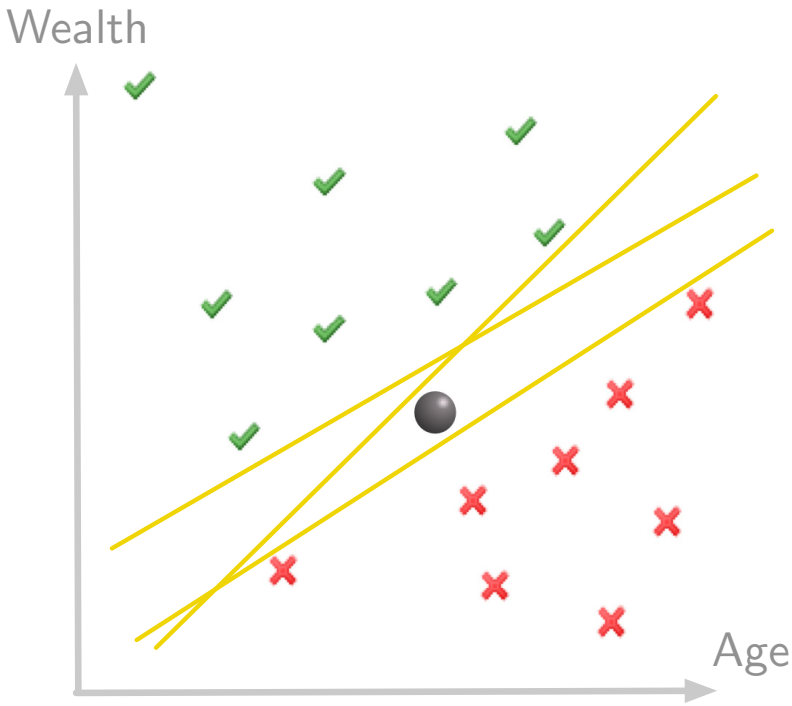
AI-based System

✓ Approve ✘ Reject



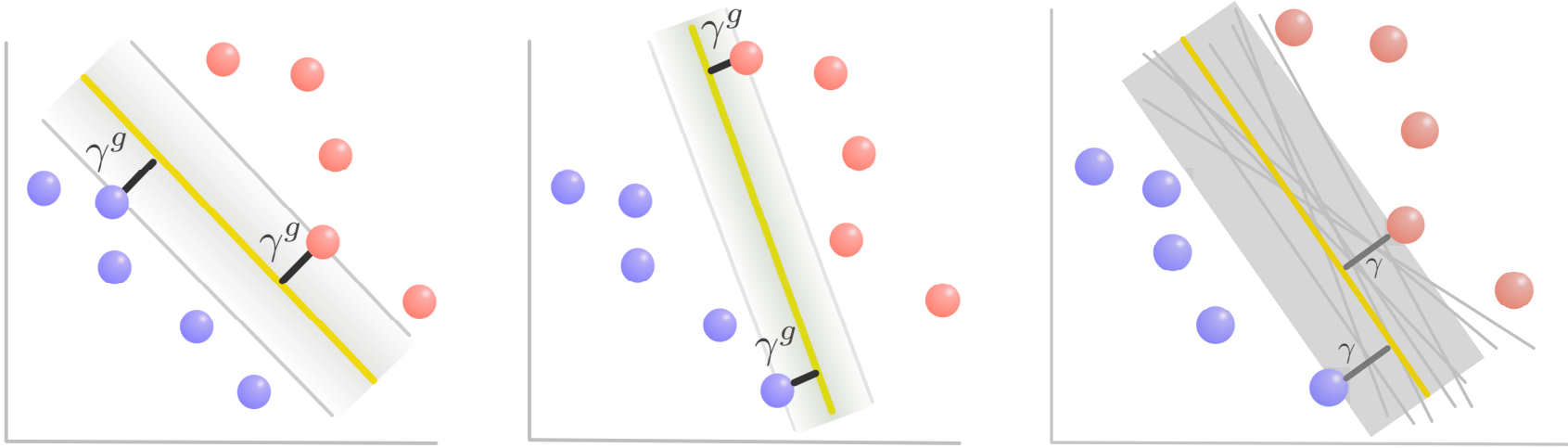
Mortgage loan application: But is the new decision "right"?

- Many ways to separate the data
- Is there an "optimal" way ?



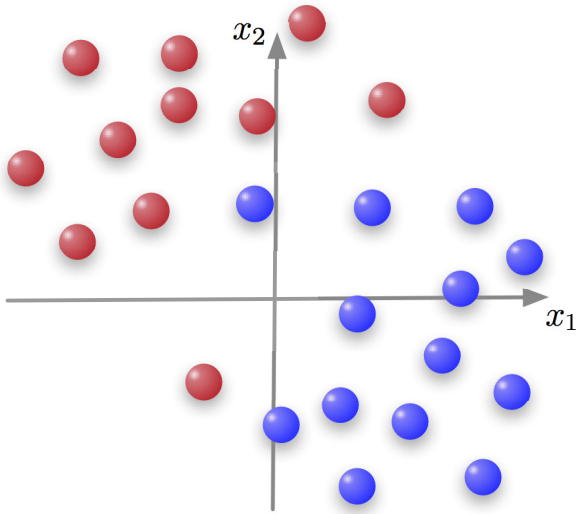
The theory of support vector machines provides a systematic method for separating the data "optimally"

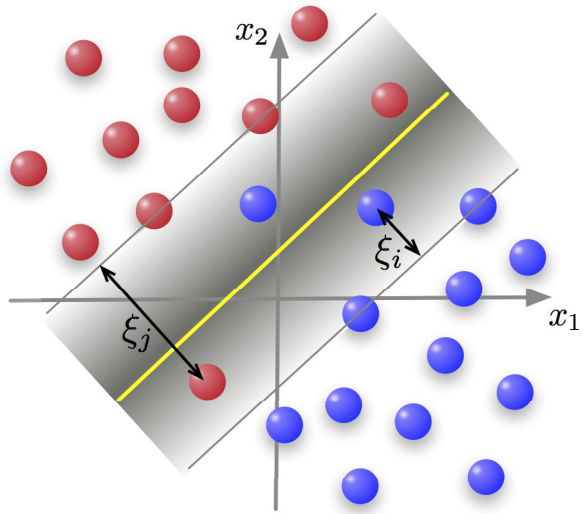
- Many hyperplanes correctly classify given data set S
- S has a geometric margin γ^g with respect to hyperplane



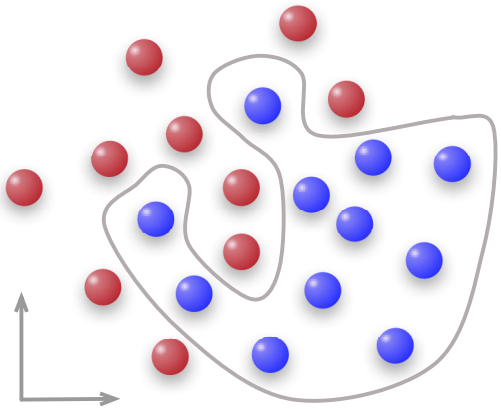
- The optimal hyperplane for a given S is one that gives $\gamma \triangleq \max \gamma^g$ over all possible hyperplanes
- A SVM implements the optimal hyperplane for a given S

1. Find optimal hyperplane to minimize classification error

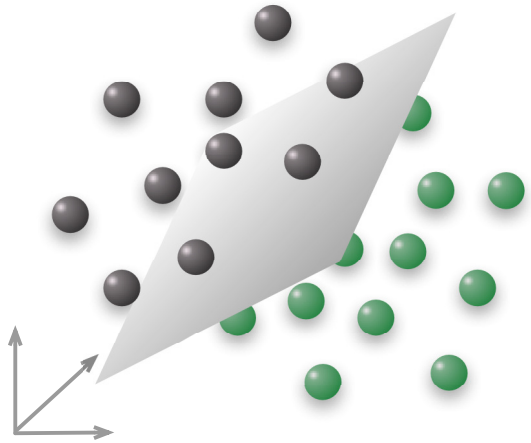




2. Transform data into higher dimension space for separation

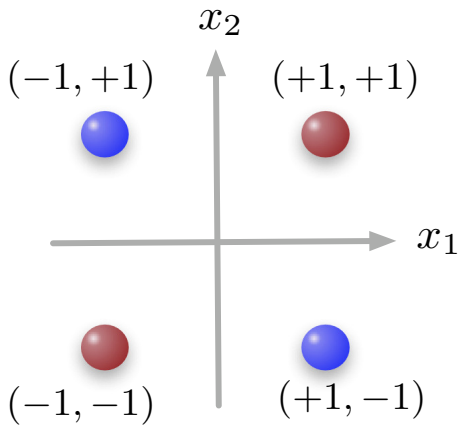


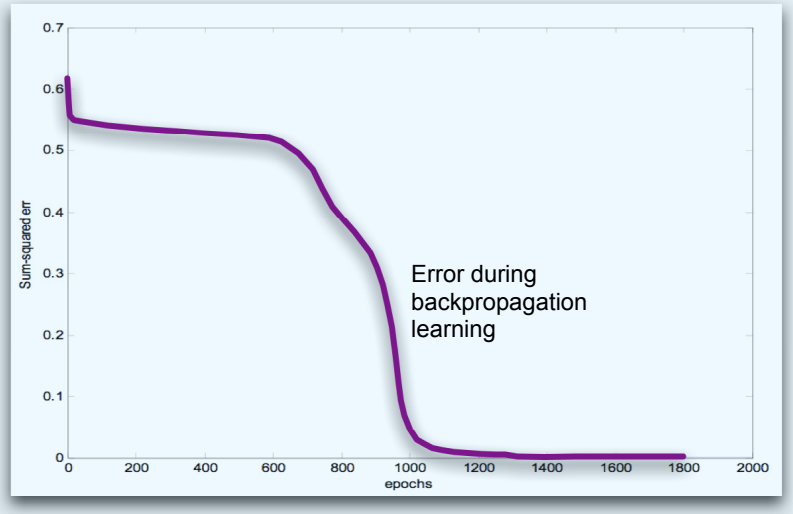
Input space



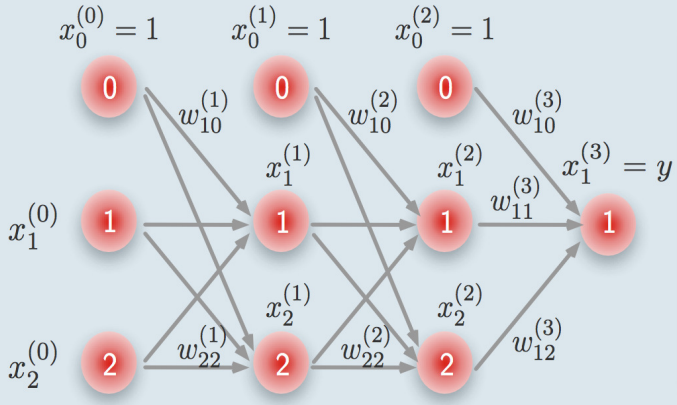
Feature space

Advantage of SVM: Comparison of MLP and SVM solutions to XOR problem





MLP solution

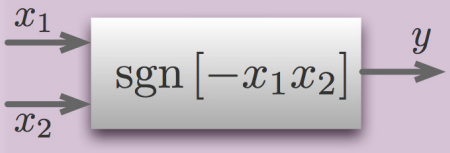


$$\boldsymbol{w}^{(1)} = \begin{matrix}\text{bias}\\ \end{matrix}\begin{bmatrix} -1.87 & 5.17 & 5.10 \\ -4.66 & 3.06 & 3.04 \end{bmatrix}$$

$$\boldsymbol{w}^{(2)} = \begin{matrix}\text{bias}\\ \end{matrix}\begin{bmatrix} -1.06 & 3.69 & -5.29 \\ 2.53 & -4.67 & 3.54 \end{bmatrix}$$

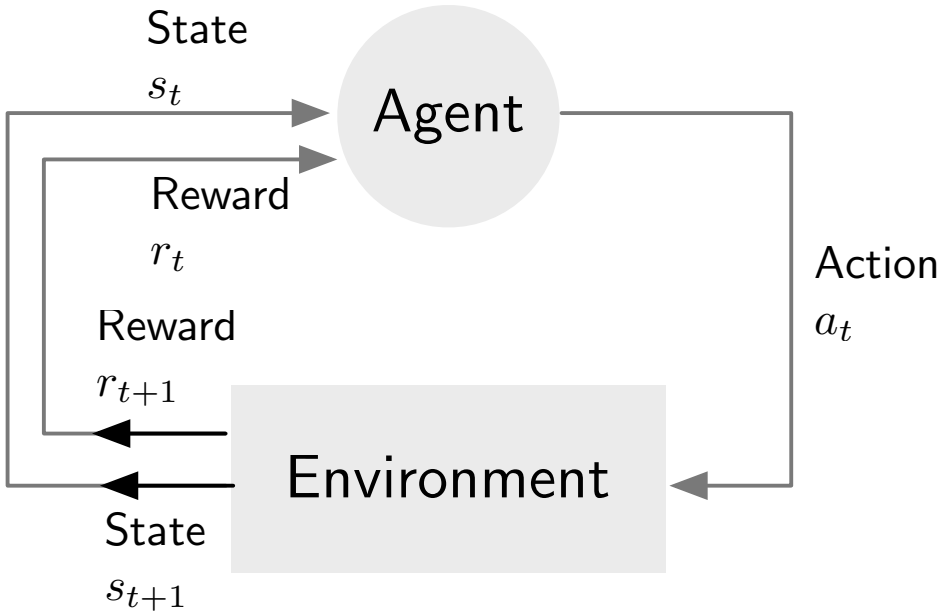
$$\boldsymbol{w}^{(3)} = \begin{matrix}\text{bias}\\ \end{matrix}[-0.36,\ 6.49,\ -6.5147]$$

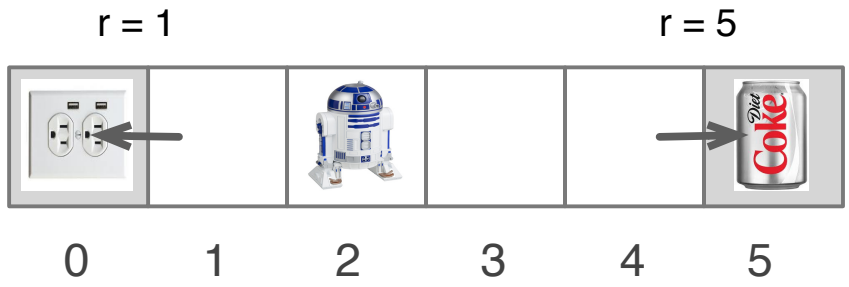
SVM solution

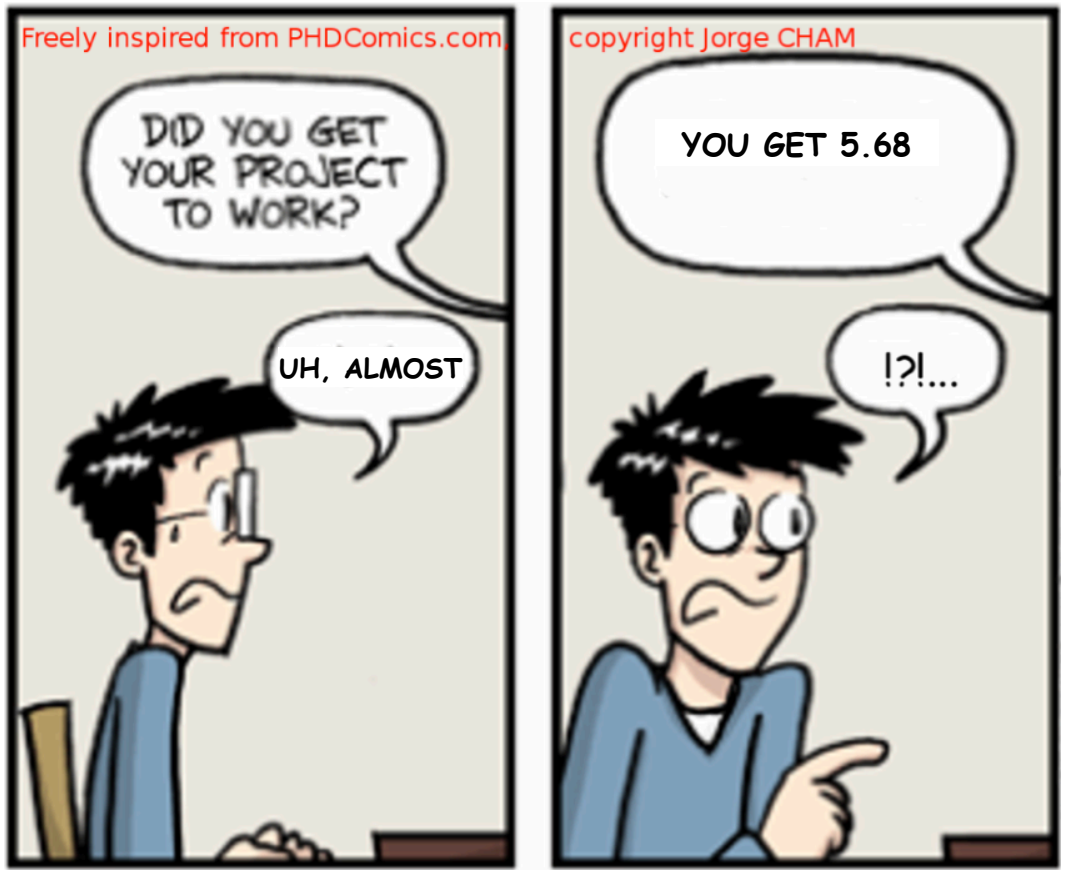


MLP	SVM
A solution among many	Optimal solution
Complicated structure	Simpler structure
A "blackbox" (Convergence not guaranteed)	Tractable solution process (Intuitive visualization for 2D problems)

Agent learns to maximize reward when completing task

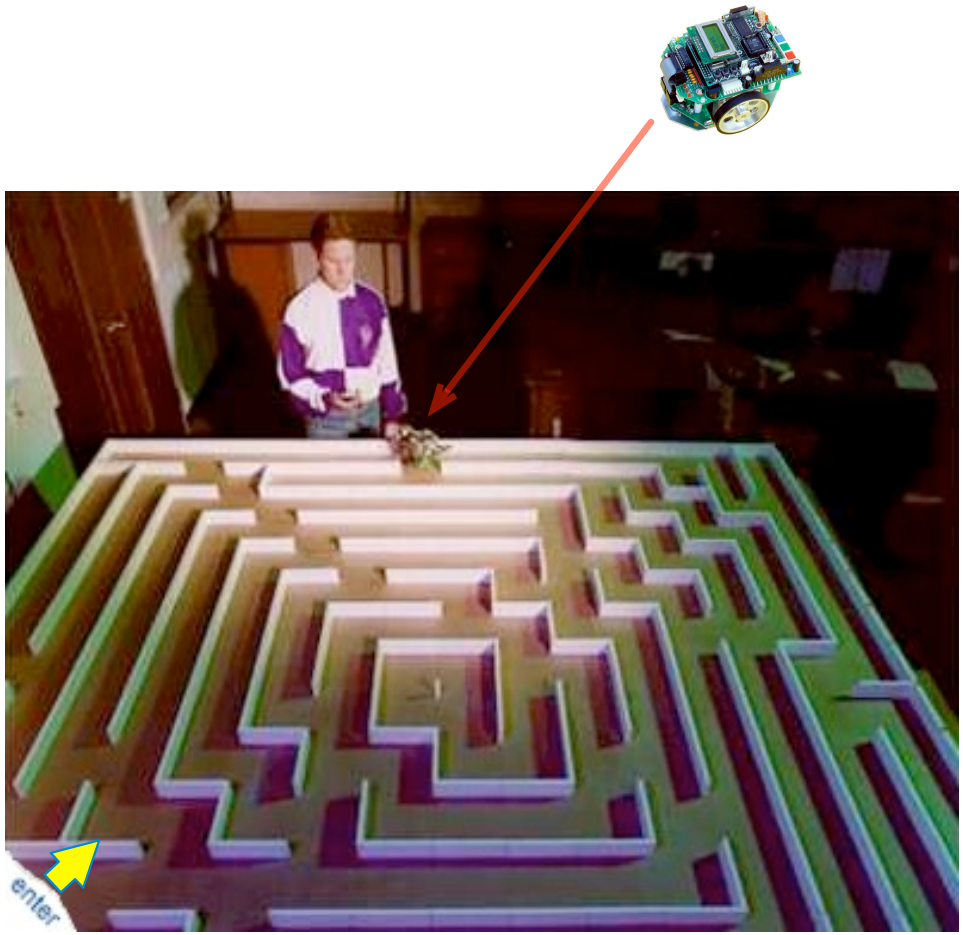






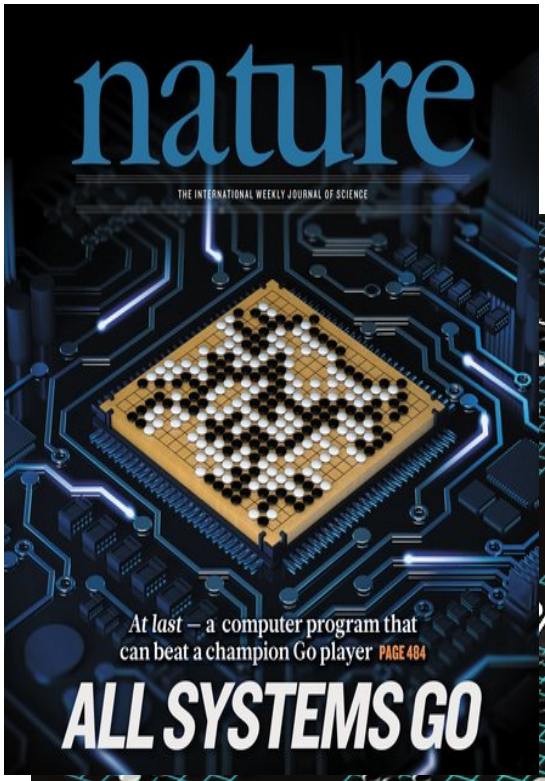
Adapted from slides by Olivier Sigaud

- Is 5.68 good or bad, how good or how bad?
- The score does not tell you how to get the project to work
- You have to learn how yourself by maximizing your scores



- Robot has no knowledge about maze before learning starts
- Robot explores maze and receives reward if it reaches goal
- Robot learns the "optimal" route to reach goal

AlphaGo used RL to beat world No. 1 ranked player in 3-game match in 2017



On successful completion of this part of module, students will be able to

1. Formulate pattern classification problems in the framework of support vector machines and apply optimization methods to find a solution

2. Develop computational solutions based on Markov decision theory for practical machine-learning problems involving optimization of outcomes

$\mathbb{R}$:	Set of real numbers
$\mathbf{w} \in \mathbb{R}^{n \times 1}$:	n-dimensional real column vector
$\mathbf{A} \in \mathbb{R}^{n \times n}$:	$n \times n$ matrix

The signum function (also called sign function) is defined as

$$\text{sgn}\,[u] = \begin{cases} +1 & \text{if } u \geq 0 \\ -1 & \text{if } u < 0 \end{cases}$$

The gradient of a function $f(\mathbf{w})$ is defined as

$$\frac{\partial f(\mathbf{w})}{\partial \mathbf{w}} = \begin{bmatrix} \dfrac{\partial f(\mathbf{w})}{\partial w_1} \\ \vdots \\ \dfrac{\partial f(\mathbf{w})}{\partial w_n} \end{bmatrix}$$

The inner product of two vectors $\mathbf{x} \in \mathbb{R}^{n\times 1}$ and $\mathbf{y} \in \mathbb{R}^{n\times 1}$ is defined as

$$\langle \mathbf{x} \cdot \mathbf{y} \rangle = \sum_{i=1}^{n} x_i \, y_i = x_1 \, y_1 + x_2 \, y_2 + \cdots + x_n \, y_n = \mathbf{x}^T \mathbf{y}$$

Two vectors $\mathbf{u}$ and $\mathbf{v}$ are said to be orthogonal if

$$\langle \mathbf{u} \cdot \mathbf{v} \rangle = \mathbf{u}^T \mathbf{v} = 0$$

Back

The Euclidean norm (also called the l_2-norm) of a vector $\mathbf{x} \in \mathbb{R}^{n\times 1}$, denoted by $\|\mathbf{x}\|$, is defined as

$$\|\mathbf{x}\| = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2} = \sqrt{\langle \mathbf{x} \cdot \mathbf{x} \rangle} \equiv \sqrt{\mathbf{x}^T \mathbf{x}}$$

For vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^{n \times 1}$ and a matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$, we have the following rules

$$\mathbf{u}^T \mathbf{v} = \mathbf{v}^T \mathbf{u}$$

$$\frac{\partial \left(\mathbf{u}^T \mathbf{v} \right)}{\partial \mathbf{v}} = \mathbf{u}$$

$$\frac{\partial \left(\mathbf{u}^T \mathbf{A} \mathbf{u} \right)}{\partial \mathbf{u}} = \left(\mathbf{A} + \mathbf{A}^T \right) \mathbf{u}$$

If $\mathbf{A} = \mathbf{I}$, where $\mathbf{I}$ denotes the identity matrix, then

$$\frac{\partial \left(\mathbf{u}^T \mathbf{A} \mathbf{u} \right)}{\partial \mathbf{u}} = \left(\mathbf{A} + \mathbf{A}^T \right) \mathbf{u} = \left(\mathbf{I} + \mathbf{I}^T \right) \mathbf{u} = 2 \, \mathbf{u}$$

Back

Consider the equation

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$$

The non-zero vectors $\mathbf{x}$ are called the eigenvectors of $\mathbf{A}$, and the corresponding values of λ are called the eigenvalues of $\mathbf{A}$

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x} \quad \Rightarrow \quad \lambda\mathbf{x} - \mathbf{A}\mathbf{x} = 0 \quad \Rightarrow \quad (\lambda I - \mathbf{A})\,\mathbf{x} = 0$$

Existence of non-trivial solution requires that

$$|\lambda I - \mathbf{A}| = 0$$

whose solution gives eigenvalues λ of $\mathbf{A}$

Back

Example: For a matrix $\mathbf{A} = \begin{bmatrix} 3 & -2 \\ -1 & 4 \end{bmatrix}$, we have

$$|\lambda I - \mathbf{A}| = \begin{vmatrix} \lambda - 3 & 2 \\ 1 & \lambda - 4 \end{vmatrix} = (\lambda - 3)(\lambda - 4) - 2 = 0$$

Solving for λ yields $\lambda_1 = 5$ and $\lambda_2 = 2$. Substituting λ_1 and λ_2 back into the equation $\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$ then solving those equations yields the eigenvectors

$$\mathbf{x}_1 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}, \quad \mathbf{x}_2 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

Suppose that function $f(x)$ has maximum M

The set of values of x for which $f(x)$ attains M is expressed as

$$\underset{x}{\operatorname{argmax}}\left(f(x)\right)$$

Example: For $f(x) = \sin x$

$$M = \max\left[\sin x\right] = 1$$

and

$$\underset{x}{\operatorname{argmax}}\left(\sin x\right) = \{360^\circ k + 90^\circ\}$$

with $k = 0, 1, 2, \ldots$

Back

- Suppose a random variable X can take:

Value	x_1	x_2	$\dots$	x_k
With probability	p_1	p_2	$\dots$	p_k

- The expectation of X is defined as

$$\mathbb{E}[X] = x_1 p_1 + x_2 p_2 + \cdots x_k p_k$$

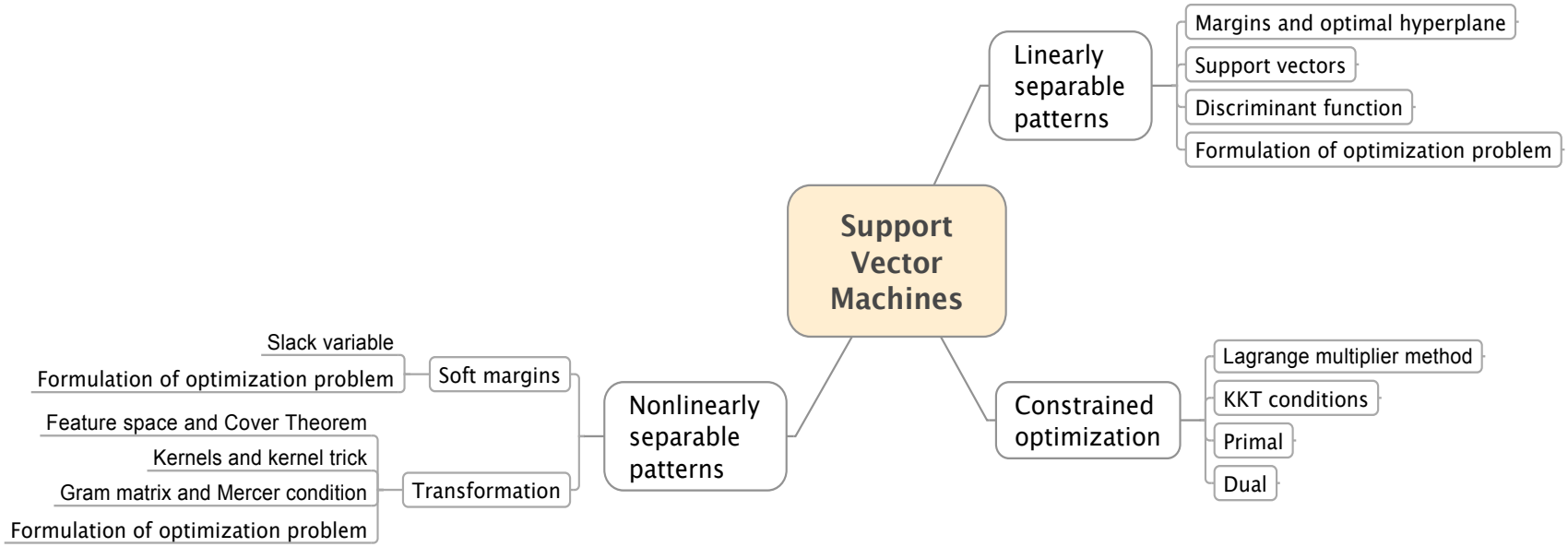
$\mathbb{E}$ is also called the expectation operator

- $\mathbb{E}$ is linear with the following properties

$$\begin{aligned} \mathbb{E}[X+Y] &= \mathbb{E}[X] + \mathbb{E}[Y] \\ \mathbb{E}[X+c] &= \mathbb{E}[X] + c \\ \mathbb{E}[cX] &= c\mathbb{E}[X] \end{aligned}$$

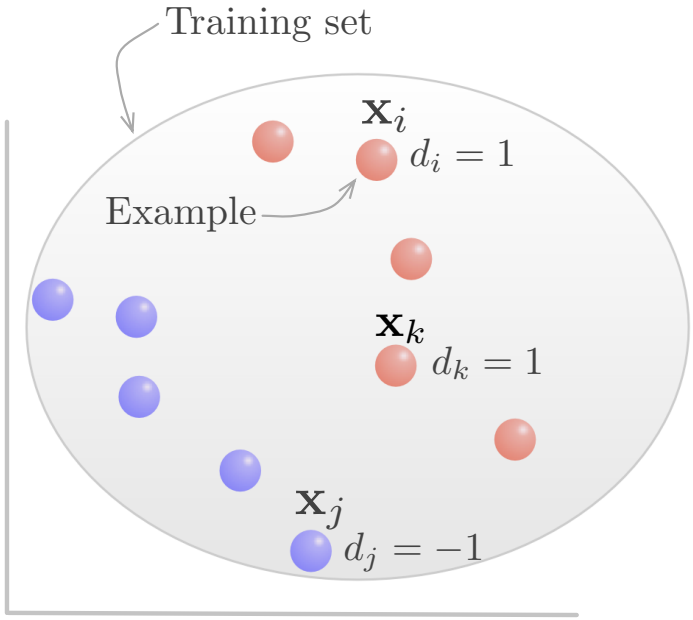
where c is a real constant

Support vector machines: Outline



Example: An n-dimensional vector

$$\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$$



assigned with a label

$$d = \pm 1$$

We write an example as: $(\mathbf{x}, d)$

Training set: A set S of N examples, i.e.,

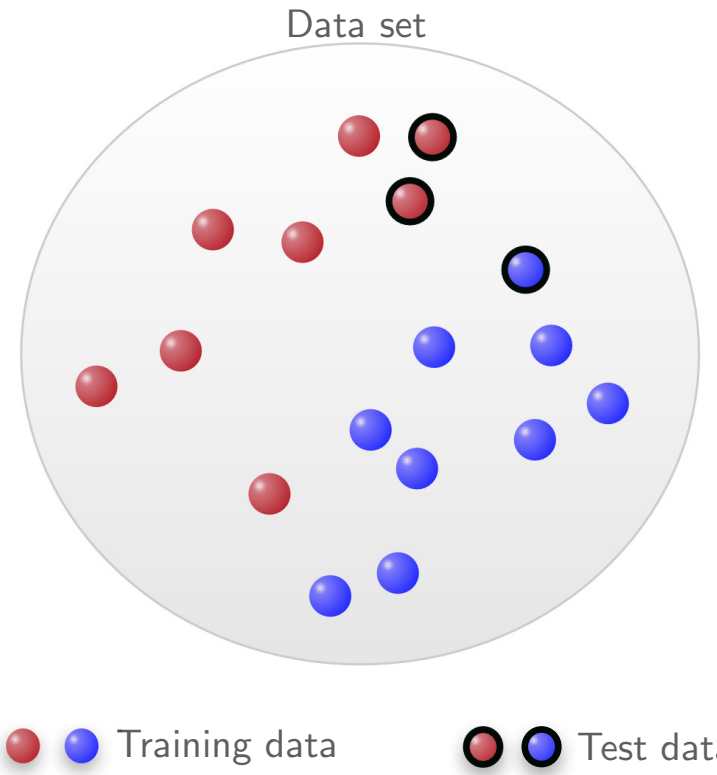
$$S = \left\{ (\mathbf{x}_1, d_1), (\mathbf{x}_2, d_2), \ldots, (\mathbf{x}_N, d_N) \right\}$$

Data set Σ containing $(N + \bar{N})$ examples is partitioned into

$$\text{a training set:} \quad S = \{(\mathbf{x}_1, d_1), \ldots, (\mathbf{x}_N, d_N)\}$$

$$\text{a test set:} \quad \bar{S} = \{(\bar{\mathbf{x}}_1, \bar{d}_1), \ldots, (\bar{\mathbf{x}}_{\bar{N}}, \bar{d}_{\bar{N}})\}$$

That is, $\Sigma = S \cup \bar{S}$, with $S \cap \bar{S} = \emptyset$

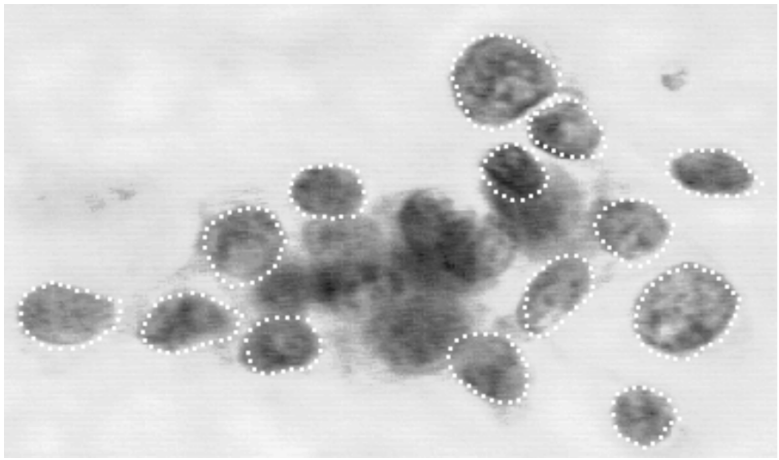


- Training data is for constructing SVM
- Test data is for evaluating performance of SVM

More on training and testing later

Breast Cancer Wisconsin (Diagnostic) Data Set

http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29



883852, B, 11.3, 18.19, 73.93, 389.4, 0.09592, 0.1325, 0.1548, 0.02854, 0.2054, 0.07669, 0.2428, 1.642, 2.369, 16.39, 0.006663, 0.05914, 0.0888, 0.01314, 0.01995, 0.008675, 12.58, 27.96, 87.16, 472.9, 0.1347, 0.4848, 0.7436, 0.1218, 0.3308, 0.1297
842302, M, 17.99, 10.38, 122.8, 1001, 0.1184, 0.2776, 0.3001, 0.1471, 0.2419, 0.07871, 1.095, 0.9053, 8.589, 153.4, 0.006399, 0.04904, 0.05373, 0.01587, 0.03003, 0.006193, 25.38, 17.33, 184.6, 2019, 0.1622, 0.6656, 0.7119, 0.2654, 0.4601, 0.1189

First number=Patient ID. B=Benign. M=Malignant

Each sample has 30 real-valued features. Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image.

W.N. Street, W.H. Wolberg and O.L. Mangasarian, Nuclear feature extraction for breast tumor diagnosis. IS&T/SPIE 1993 International Symposium on Electronic Imaging: Science and Technology, volume 1905, pages 861-870, San Jose, CA, 1993.

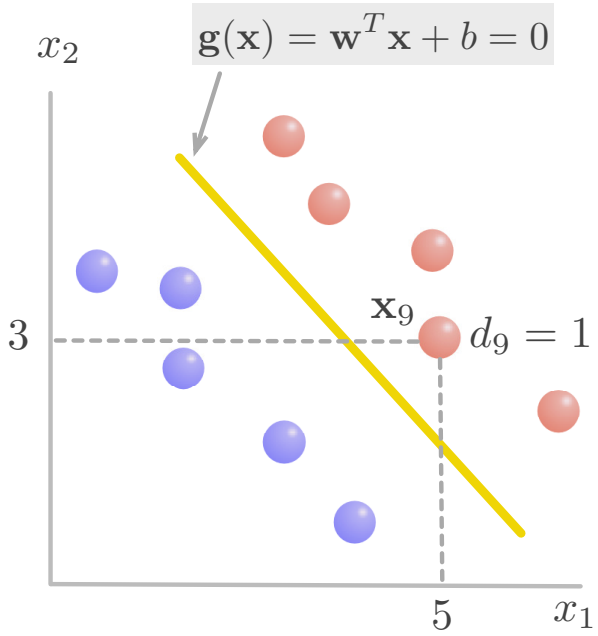
Spam Email Data Set

http://archive.ics.uci.edu/ml/datasets/spambase

train_data = [[0.00000 0.01043 0.00000 0.00000
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
0.01043 0.01043 0.02105 0.00000 0.00000 0.00000
0.00000 0.03166 0.06332 0.00000 0.02105 0.00000
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
0.00000 0.00000 0.00000 0.00000 0.00000 0.00196
0.00000 0.00000 0.02601 0.12811 0.98827], [· · ·],
· · ·]

train_label = [1, ...]

Each example has a feature vector with 57 real-valued attributes. Most of the attributes indicate whether a particular word or character was frequently occurring in the e-mail.



A hyperplane, denoted by $(\mathbf{w}, b)$, can be expressed as

$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b = 0$$

Hyperplane classifies a given $\mathbf{x}_i$ with

$$\text{sgn}\left[g(\mathbf{x}_i)\right] = \begin{cases} +1 & \text{if} \quad g(\mathbf{x}_i) > 0 \\ -1 & \text{if} \quad g(\mathbf{x}_i) < 0 \end{cases}$$

Hyperplane classifies an example $(\mathbf{x}_i, d_i)$ correctly if

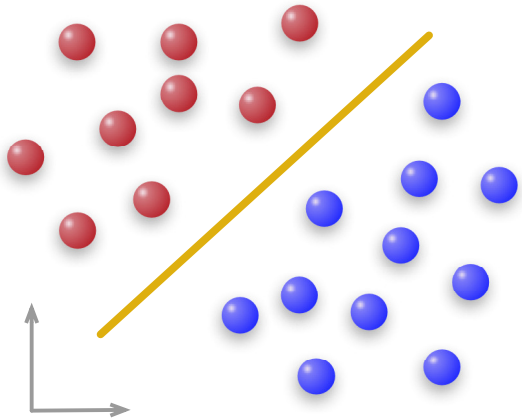
$$\text{sgn}\left[g(\mathbf{x}_i)\right] = d_i \quad \text{or} \quad \text{sgn}\left[d_i\, g(\mathbf{x}_i)\right] = 1$$

Problem: Let $\mathbf{w}^T = \begin{bmatrix} 7 & 6 \end{bmatrix}$ and $b = -42$. For $\mathbf{x}_9^T = \begin{bmatrix} 5 & 3 \end{bmatrix}$

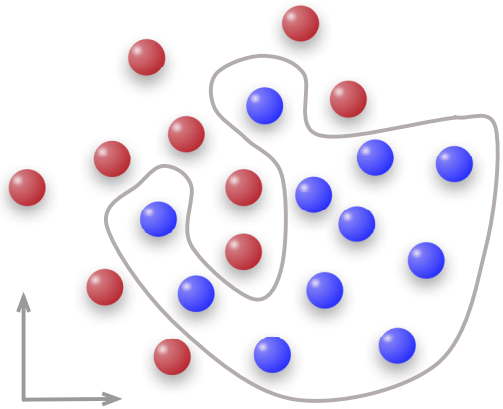
$$\mathbf{g}(\mathbf{x}_9) = \mathbf{w}^T\mathbf{x}_9 + b = \begin{bmatrix} 7 & 6 \end{bmatrix} \begin{bmatrix} 5 \\ 3 \end{bmatrix} - 42 = 11 > 0$$

$$\text{sgn}[\mathbf{g}(\mathbf{x}_9)] = \text{sgn}[11] = 1 = d_9$$

- Two classes of data are linearly separable if and only if there exists a hyperplane $\mathbf{w}^T\mathbf{x} + b = 0$ that separates the two classes
- Example in 2D:



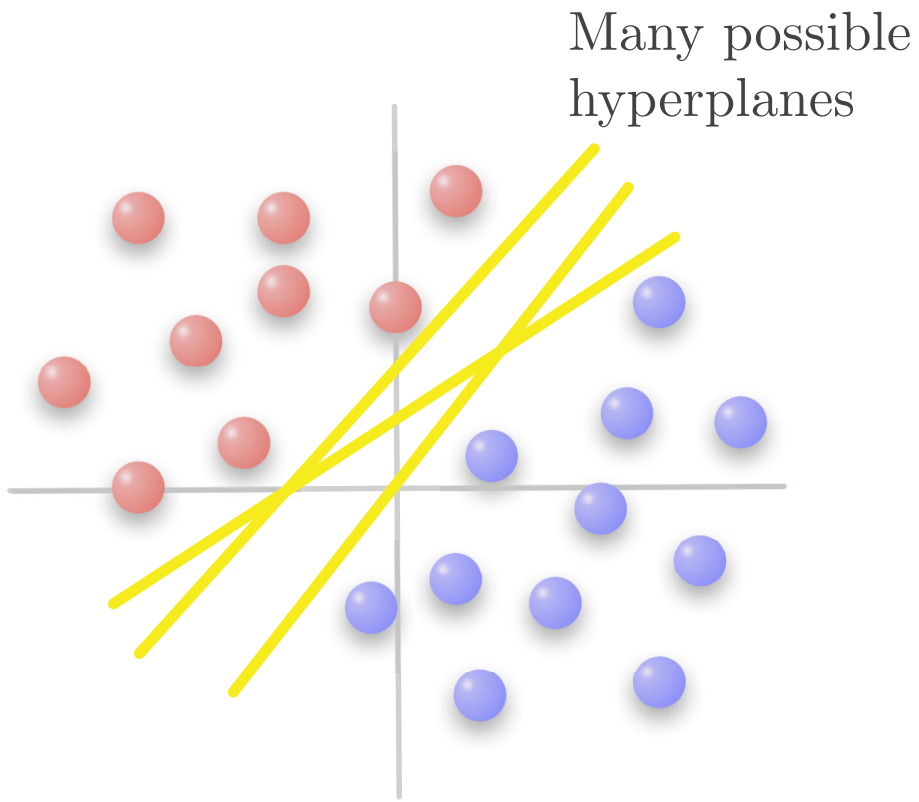
Linearly separable



Nonlinearly separable

We consider the separable case first

Is there an "optimal" way to separate the data?



The theory of support vector machines provides a systematic method for separating the data "optimally"

It involves a key concept called margin

γ_i^f: Functional margin of an example $(\mathbf{x}_i, d_i)$

γ_i^g: Geometric margin of an example $(\mathbf{x}_i, d_i)$

γ^f: Functional margin of a training set

γ^g: Geometric margin of a training set

γ: Margin of a training set

	Example	Training set
Functional margin	γ_i^f	γ^f
Geometric margin	γ_i^g	γ^g
Margin	–	γ

γ_i^f, γ^f, γ_i^g, and γ^g are defined w.r.t. a given hyperplane $(\mathbf{w}, b)$

The functional margin of an example $(\mathbf{x}_i, d_i)$ with respect to a hyperplane $(\mathbf{w}, b)$ is defined as

$$\gamma_i^f = d_i \left(\mathbf{w}^T \mathbf{x}_i + b\right)$$

Problem:

Training set with only 1 example $\mathbf{x}_1 = \begin{bmatrix} 1 \\ 4 \end{bmatrix}, \; d_1 = +1$

Hyperplane defined by $\mathbf{w} = \begin{bmatrix} 5 \\ 3 \end{bmatrix}, \; b = 6$

Determine γ_1^f with respect to given hyperplane

Solution:

$$\gamma_1^f = d_1 \left(\mathbf{w}^T \mathbf{x}_1 + b\right) = 1 \times \left(\begin{bmatrix} 5 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 4 \end{bmatrix} + 6 \right) = 1 \times 23 = 23$$

The geometric margin of an example $(\mathbf{x}_i, d_i)$ with respect to a hyperplane $(\mathbf{w}, b)$ is defined as

$$\gamma_i^g = d_i \left(\frac{1}{\|\mathbf{w}\|} \mathbf{w}^T \mathbf{x}_i + \frac{1}{\|\mathbf{w}\|} b \right)$$

Problem:

Training set with only 1 example

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ 4 \end{bmatrix}, \; d_1 = +1$$

Hyperplane defined by

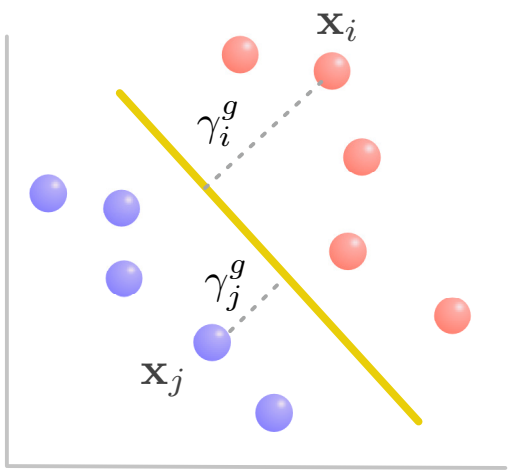
$$\mathbf{w} = \begin{bmatrix} 5 \\ 3 \end{bmatrix}, \; b = 6$$

Determine γ_1^g with respect to given hyperplane

Solution:

$$\gamma_1^g = d_1 \left(\frac{\mathbf{w}^T \mathbf{x}_1}{\|\mathbf{w}\|} + \frac{b}{\|\mathbf{w}\|} \right) = \frac{\begin{bmatrix} 5 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 4 \end{bmatrix} + 6}{\sqrt{5^2 + 3^2}} = \frac{23}{\sqrt{34}}$$

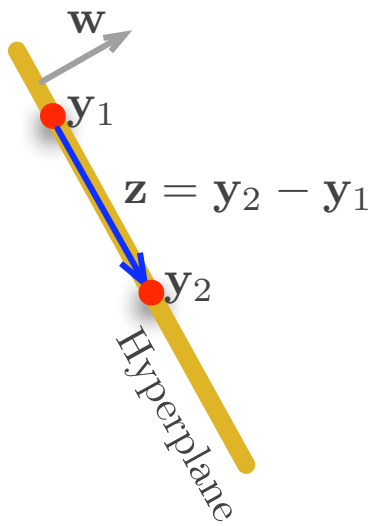
$$\gamma_i^g = d_i \left(\frac{1}{\|\mathbf{w}\|} \mathbf{w}^T \mathbf{x}_i + \frac{1}{\|\mathbf{w}\|} b \right)$$



γ_i^g is the Euclidean distance from the point defined by $\mathbf{x}_i$ to the hyperplane $(\mathbf{w}, b)$.

γ_i^g is the Euclidean distance from $\mathbf{x}_i$ to hyperplane $(\mathbf{w}, b)$

Proof: We first show that the unit vector $\frac{\mathbf{w}}{\|\mathbf{w}\|}$ is perpendicular to hyperplane $(\mathbf{w}, b)$. Consider points $\mathbf{y}_1$ and $\mathbf{y}_2$ on the hyperplane. By definition



$$\begin{aligned} \langle \mathbf{w} \cdot \mathbf{y}_1 \rangle + b &= \mathbf{w}^T \mathbf{y}_1 + b = 0 \\ \langle \mathbf{w} \cdot \mathbf{y}_2 \rangle + b &= \mathbf{w}^T \mathbf{y}_2 + b = 0 \end{aligned}$$

The vector from $\mathbf{y}_1$ to $\mathbf{y}_2$ is $\mathbf{z} = \mathbf{y}_2 - \mathbf{y}_1$. Now

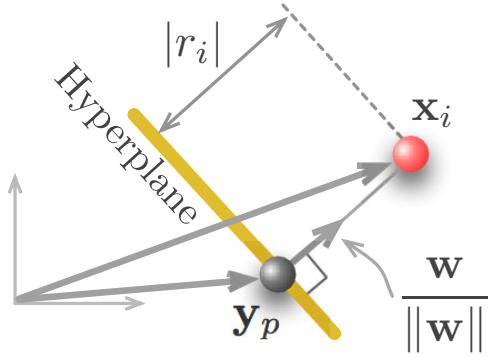
$$\begin{aligned} \langle \mathbf{w} \cdot \mathbf{z} \rangle &= \langle \mathbf{w} \cdot (\mathbf{y}_2 - \mathbf{y}_1) \rangle = \langle \mathbf{w} \cdot \mathbf{y}_2 \rangle - \langle \mathbf{w} \cdot \mathbf{y}_1 \rangle \\ &= \mathbf{w}^T \mathbf{y}_2 - \mathbf{w}^T \mathbf{y}_1 = -b - (-b) \\ &= 0 \end{aligned}$$

▸ View

Hence, $\mathbf{w} \perp$ hyperplane $(\mathbf{w}, b)$.

We can now express $\mathbf{x}_i$ as

$$\mathbf{x}_i = \mathbf{y}_p + r_i \left(\frac{\mathbf{w}}{\|\mathbf{w}\|} \right)$$



where $|r_i|$ is the distance from $\mathbf{x}_i$ to hyperplane.

Note that $\mathbf{w}^T \mathbf{y}_p + b = 0$, or, $\mathbf{w}^T \mathbf{y}_p = -b$ (since $\mathbf{y}_p$ is on hyperplane)

$$\begin{aligned} \mathbf{w}^T \mathbf{x}_i + b &= \mathbf{w}^T \left(\mathbf{y}_p + \frac{r_i \mathbf{w}}{\|\mathbf{w}\|} \right) + b = \mathbf{w}^T \mathbf{y}_p + \frac{r_i \mathbf{w}^T \mathbf{w}}{\|\mathbf{w}\|} + b \\ &= -b + \frac{r_i \|\mathbf{w}\|^2}{\|\mathbf{w}\|} + b = r_i \|\mathbf{w}\| \end{aligned}$$

So

$$r_i = \frac{\mathbf{w}^T \mathbf{x}_i + b}{\|\mathbf{w}\|} = \left(\frac{1}{\|\mathbf{w}\|} \mathbf{w}^T \mathbf{x}_i \right) + \frac{b}{\|\mathbf{w}\|}$$

Note that $\gamma_i^g = d_i \, r_i$. Since $d_i = \pm 1$, and d_i and $(\mathbf{w}^T \mathbf{x}_i + b)$ have the same sign (for a correctly classified example), we can conclude that $\gamma_i^g = |r_i|$, which is the Euclidian distance from $\mathbf{x}_i$ to the hyperplane.

The equation representing a hyperplane is invariant when scaled by an arbitrary real constant.

The hyperplane described by the equation

$$\langle \mathbf{w} \cdot \mathbf{x} \rangle + b = \mathbf{w}^T \mathbf{x} + b = 0$$

can also be described by

$$\langle c\,\mathbf{w} \cdot \mathbf{x} \rangle + c\,b = c\,\mathbf{w}^T \mathbf{x} + c\,b = 0$$

where c is a real number.

Problem: (Note that from earlier slide, $\gamma_1^f = 23$ and $\gamma_1^g = \frac{23}{\sqrt{34}}$)

Training set with only 1 example

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ 4 \end{bmatrix}, \; d_1 = +1$$

Given hyperplane

$$\mathbf{w} = \begin{bmatrix} 5 \\ 3 \end{bmatrix}, \; b = 6$$

Find scaling constant c such that $\gamma_1^f = 5$ with respect to the given hyperplane, and calculate γ_1^g associated with $c\mathbf{w}$ and cb.

Solution:

$$d_1 \left(c\, \mathbf{w}^T \mathbf{x}_1 + c\, b \right) = 1 \left(c \begin{bmatrix} 5 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 4 \end{bmatrix} + 6c \right) = 23\, c = 5 \;\Rightarrow\; c = \frac{5}{23}$$

Thus

$$c\mathbf{w} = \frac{5}{23} \begin{bmatrix} 5 \\ 3 \end{bmatrix} = \begin{bmatrix} 25/23 \\ 15/23 \end{bmatrix}, \quad cb = \frac{30}{23}$$

and the geometric margin of the example with respect to $c\mathbf{w}$ and cb is

$$\gamma_1^g = \frac{\gamma_1^f}{\|c\mathbf{w}\|} = \frac{5}{\sqrt{\left(\frac{25}{23}\right)^2 + \left(\frac{15}{23}\right)^2}} = \frac{5}{\frac{5\sqrt{34}}{23}} = \frac{23}{\sqrt{34}}$$

Functional margin of an example

$$\gamma_i^f = d_i \left(\mathbf{w}^T \mathbf{x}_i + b \right)$$

Geometric margin of an example

$$\gamma_i^g = d_i \left(\frac{1}{\|\mathbf{w}\|} \mathbf{w}^T \mathbf{x}_i + \frac{1}{\|\mathbf{w}\|} b \right)$$

Therefore, by definition:

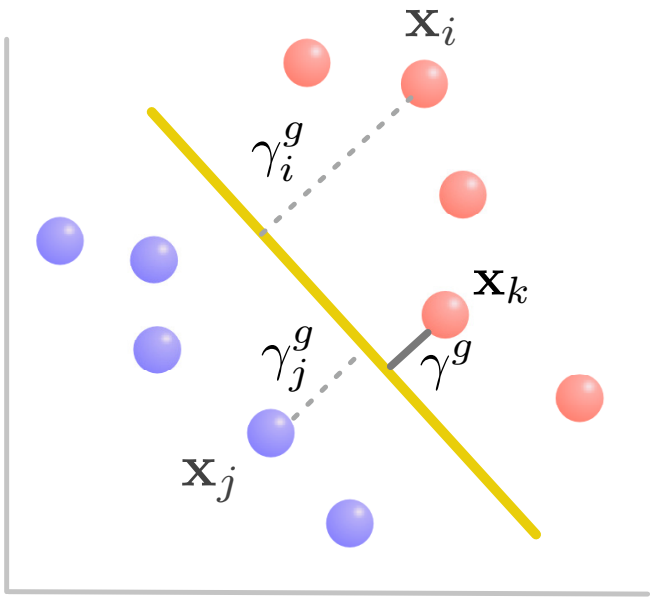
$$\gamma_i^g = \frac{\gamma_i^f}{\|\mathbf{w}\|}$$

The functional margin of a training set S, with respect to a hyperplane $(\mathbf{w}, b)$, is the minimum of all the functional margins of the individual examples in S, i.e.,

$$\gamma^f = \min_{1 \leq i \leq N} \left\{ \gamma_i^f \right\}$$

where

$$\gamma_i^f = d_i \left(\mathbf{w}^T \mathbf{x}_i + b \right)$$



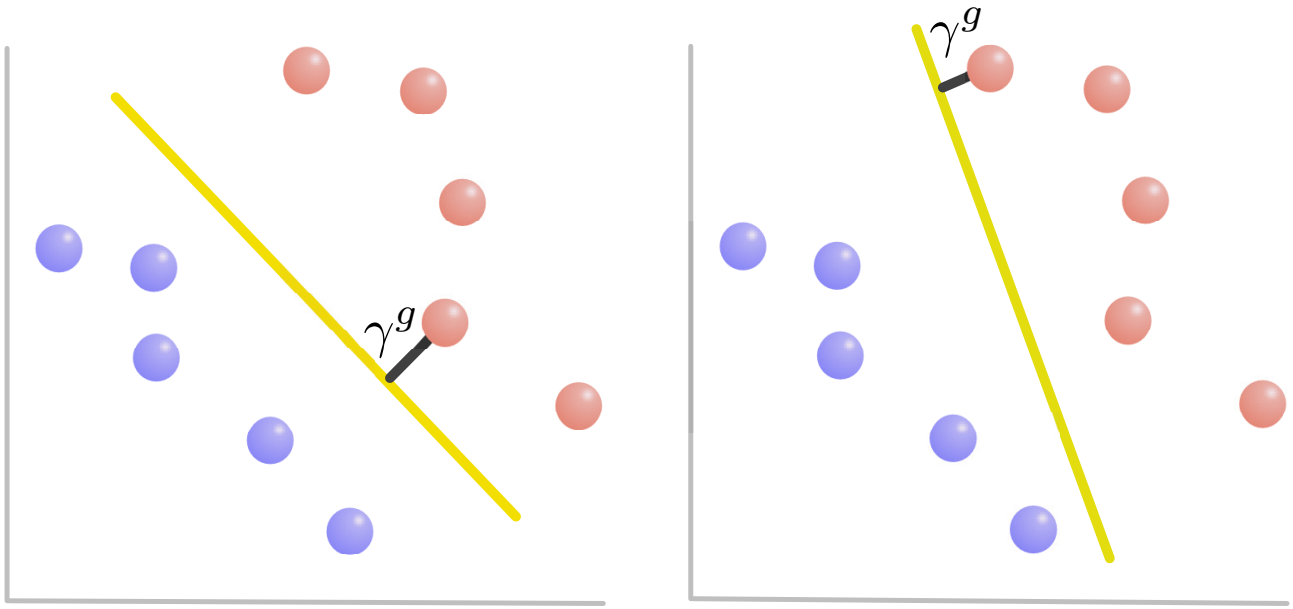
The geometric margin of a training set S, with respect to a hyperplane $(\mathbf{w}, b)$, is the minimum of all the geometric margins of the individual examples in S, i.e.,

$$\gamma^g = \min_{1 \le i \le N} \{\gamma_i^g\}$$

where

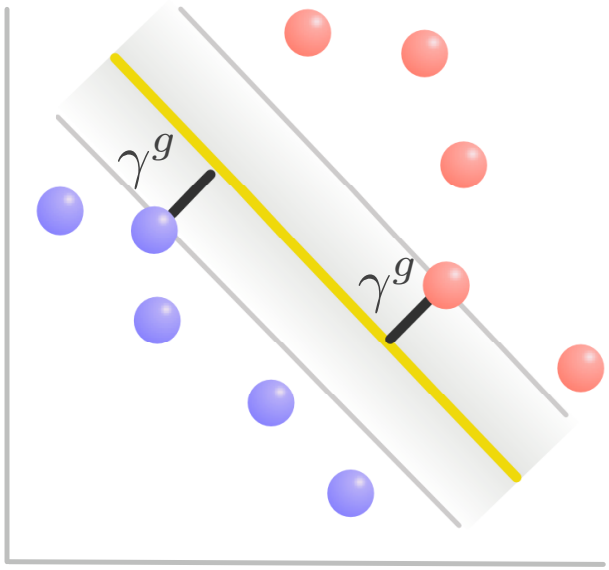
$$\gamma_i^g = d_i \left(\frac{1}{\|\mathbf{w}\|} \mathbf{w}^T \mathbf{x}_i + \frac{1}{\|\mathbf{w}\|} b \right)$$

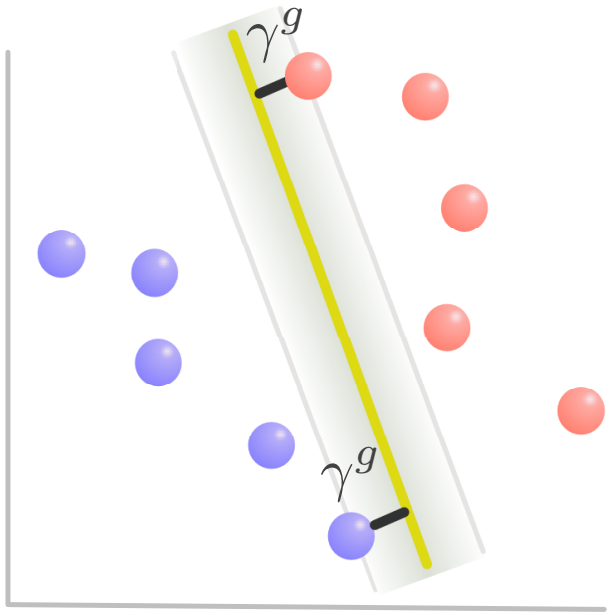
- A training set S can have different geometric margins depending on how hyperplane is defined

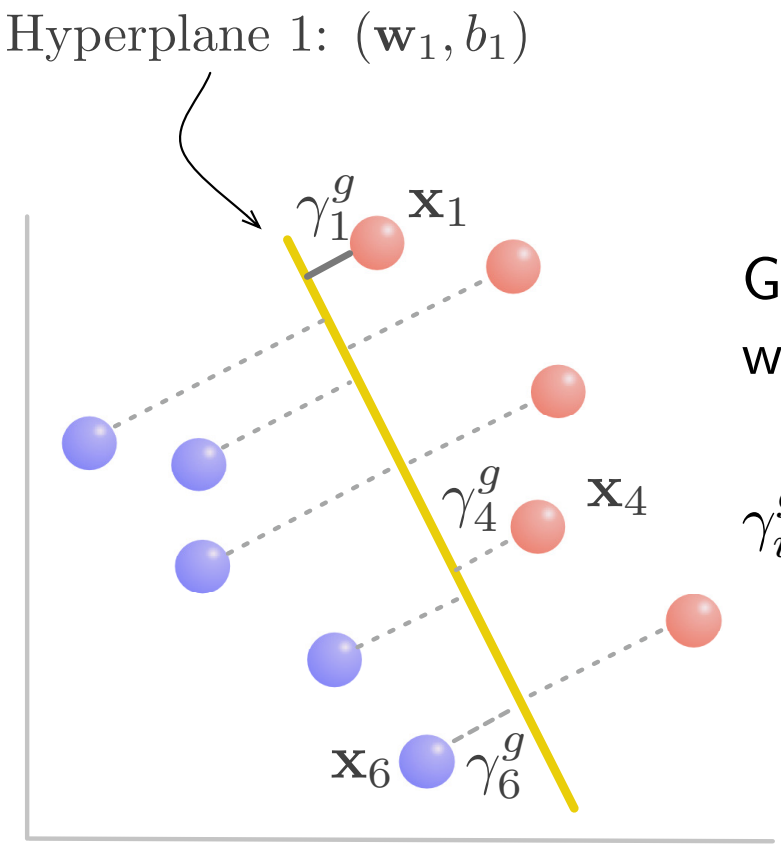


- The optimal hyperplane for a given S is one that gives maximum γ^g over all possible hyperplanes

- The optimal hyperplane for a given S is one that gives maximum γ^g over all possible hyperplanes
- This maximum γ^g is called the margin of S





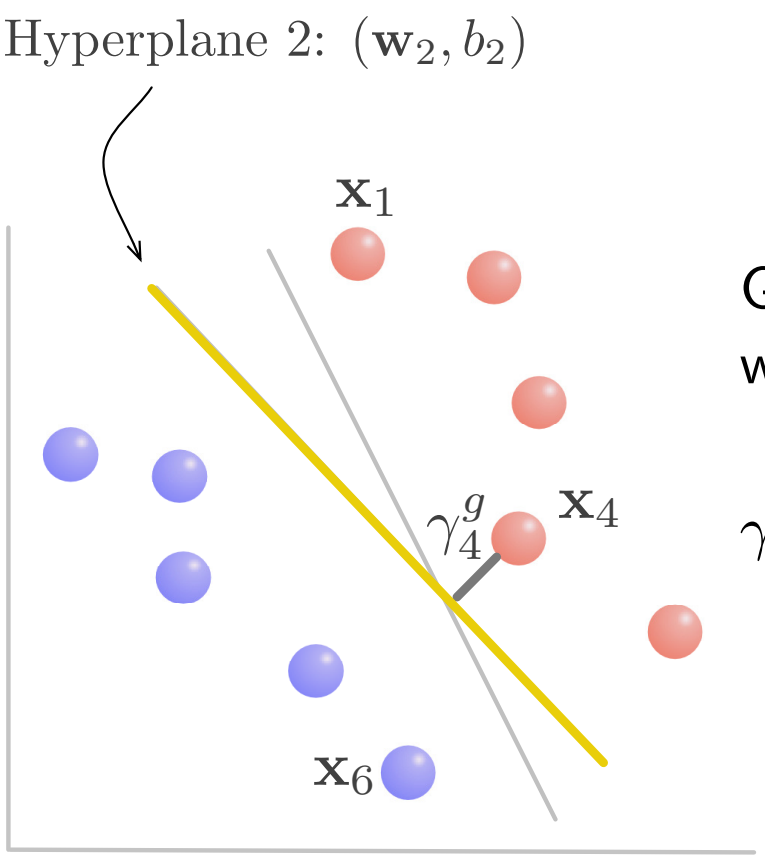


Geometric margin of an example i with respect to **hyerplane 1** is

$$\gamma_i^g = d_i \left(\left\langle \frac{1}{\|\mathbf{w}_1\|} \mathbf{w}_1 \cdot \mathbf{x}_i \right\rangle + \frac{1}{\|\mathbf{w}_1\|} b_1 \right)$$

Geometric margin of training set with respect to hyperplane 1 is

$$\min\{\gamma_1^g, \ldots, \gamma_{10}^g\} = \gamma_1^g \equiv \gamma_{(\mathbf{w}_1, b_1)}^g$$

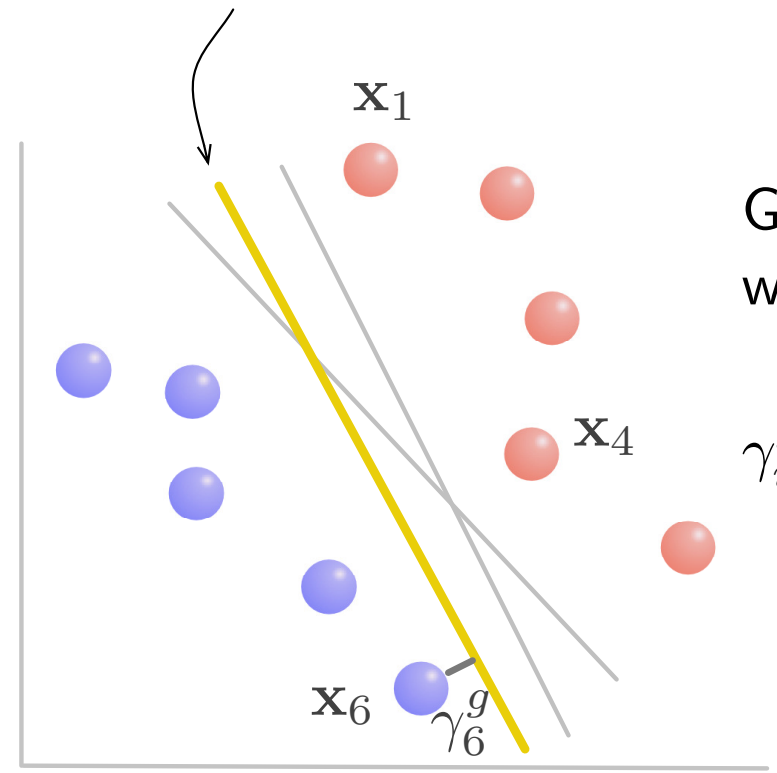


Geometric margin of an example i with respect to **hyerplane 2** is

$$\gamma_i^g = d_i \left(\left\langle \frac{1}{\|\mathbf{w}_2\|} \mathbf{w}_2 \cdot \mathbf{x}_i \right\rangle + \frac{1}{\|\mathbf{w}_2\|} b_2 \right)$$

Geometric margin of training set with respect to hyperplane 2 is

$$\min\{\gamma_1^g, \ldots, \gamma_{10}^g\} = \gamma_4^g \equiv \gamma_{(\mathbf{w}_2, b_2)}^g$$



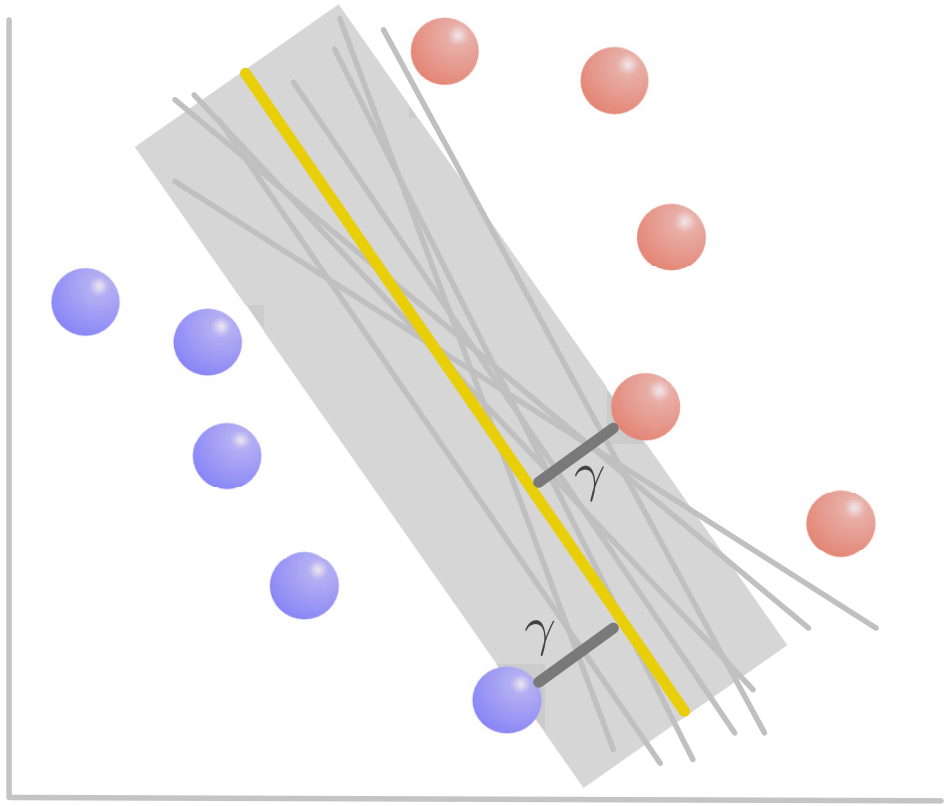
Geometric margin of an example i with respect to **hyerplane 3** is

$$\gamma_i^g = d_i \left(\left\langle \frac{1}{\|\mathbf{w}_3\|} \mathbf{w}_3 \cdot \mathbf{x}_i \right\rangle + \frac{1}{\|\mathbf{w}_3\|} b_3 \right)$$

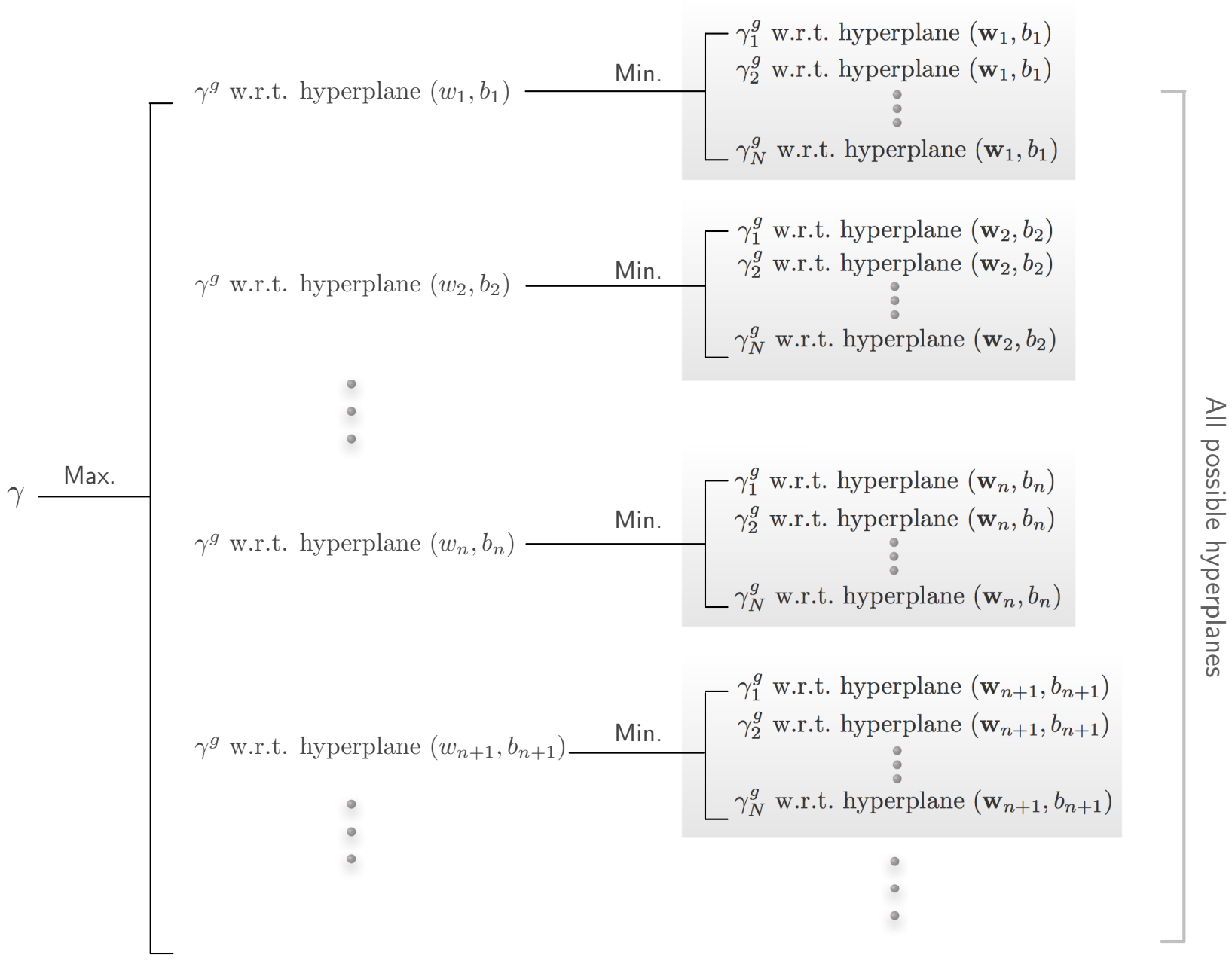
Geometric margin of training set with respect to hyperplane 3 is

$$\min\{\gamma_1^g, \ldots, \gamma_{10}^g\} = \gamma_6^g \equiv \gamma_{(\mathbf{w}_3, b_3)}^g$$

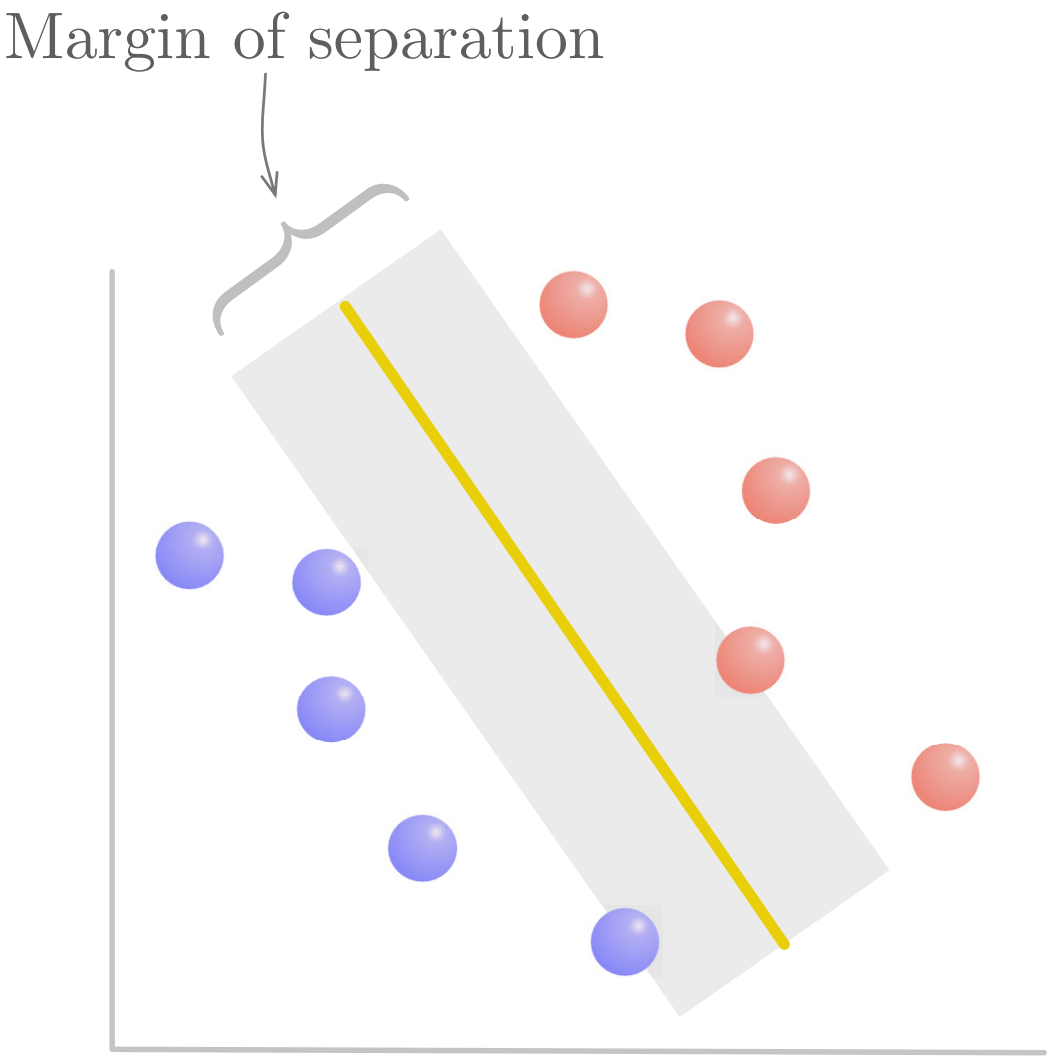
$$\gamma = \max \underbrace{\left\{ \gamma^g_{(\mathbf{w}_1,b_1)}, \gamma^g_{(\mathbf{w}_2,b_2)}, \gamma^g_{(\mathbf{w}_3,b_3)}, \cdots \right\}}_{\text{All possible hyperplanes}}$$



Margin of training set (with N examples): Summary

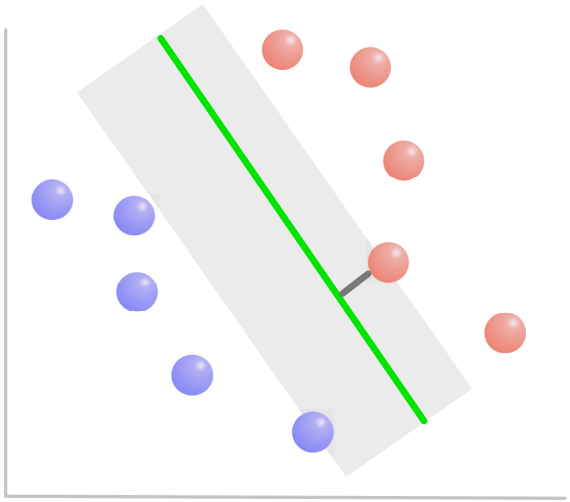


Optimal hyperplane divides margin of separation evenly

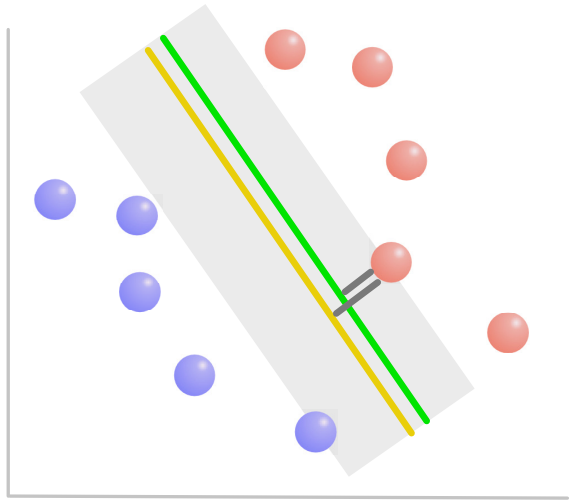


Optimal hyperplane divides margin of separation evenly: Rationale

$$\gamma = \max \underbrace{\left\{ \gamma^g_{(\mathbf{w}_1, b_1)}, \gamma^g_{(\mathbf{w}_2, b_2)}, \gamma^g_{(\mathbf{w}_3, b_3)}, \dots \right\}}_{\text{All possible hyperplanes}}$$



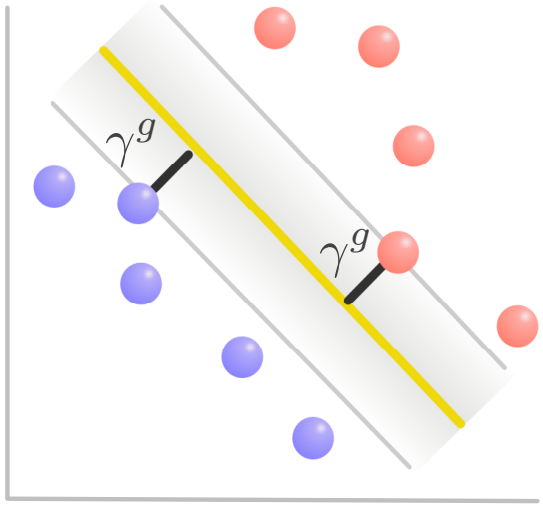
If hyperplane is optimal (with the margin as shown) but does not evenly divide margin of separation

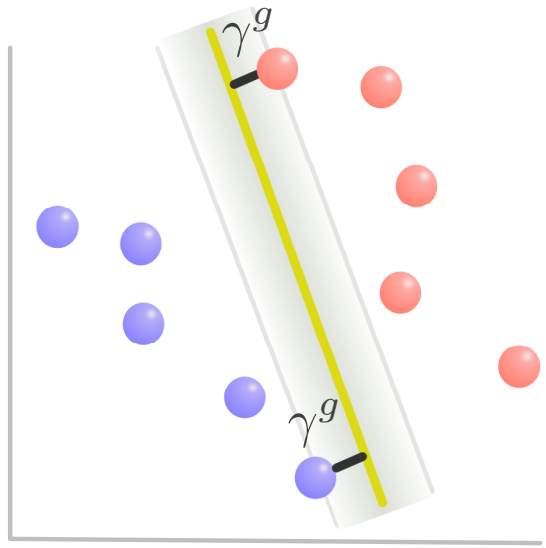


Then we can always move this hyperplane to get a larger margin, which means the original hyperplane is not optimal

Two key questions:

- Why is it important to find the margin of S ?
- How to find the margin of S ?





Two key questions:

- Why is it important to find the margin of S ?
- How to find the margin of S ?

Larger margin leads to lower probability of misclassification

Theorem 4.18 (in book by Cristianini and Shawe-Taylor): Consider thresholding real-valued linear functions $\mathcal{L}$ with unit weight vectors on an inner product space X and fix $\gamma \in \mathbb{R}^+$. For any probability distribution $\mathcal{D}$ on $X \times \{-1, 1\}$ with support in a ball of radius R around the origin, with probability $1 - \delta$ over l random examples S, any hypothesis $f \in \mathcal{L}$ that has margin $m_s(f) \geq \gamma$ on S has error no more than

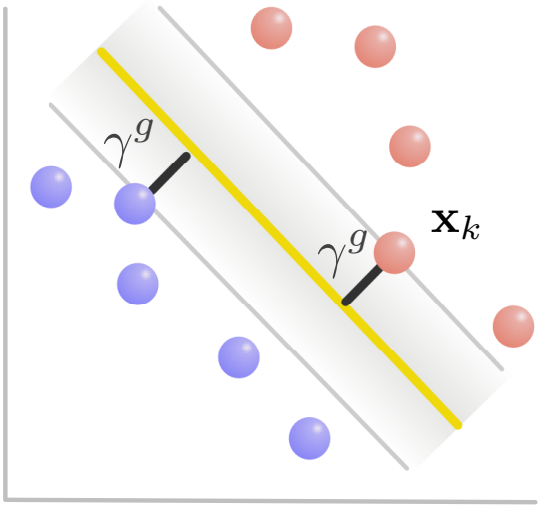
$$\mathbf{err}_{\mathcal{D}}(f) \leq \varepsilon(l, \mathcal{L}, \delta, \gamma) = \frac{2}{l}\left(\frac{64R^2}{\boldsymbol{\gamma}^2} \log \frac{el\boldsymbol{\gamma}}{4R} \log \frac{128lR^2}{\boldsymbol{\gamma}^2} + \log \frac{4}{\delta}\right)$$

provided $l > 2/\varepsilon$ and $64R^2/\gamma^2 < l$.

(We will just take it as fact and will not study detailed proof in this module.)

Two key questions:

- Why is it important to find the margin of S ?
- How to find the margin of S ?



Recall the relationship: $\gamma_i^g = \dfrac{\gamma_i^f}{\|\mathbf{w}\|}$

For a given $\mathbf{w}$, the example k that yields $\gamma_k^f = \gamma^f$ also yields $\gamma^g = \gamma_k^g$. So we can write

$$\gamma^g = \frac{\gamma^f}{\|\mathbf{w}\|}$$

We can maximize γ^g by fixing γ^f then minimizing $\|\mathbf{w}\|$

$\gamma^g = \frac{\gamma^f}{\|\mathbf{w}\|}$: We can maximize γ^g by fixing γ^f then minimizing $\|\mathbf{w}\|$

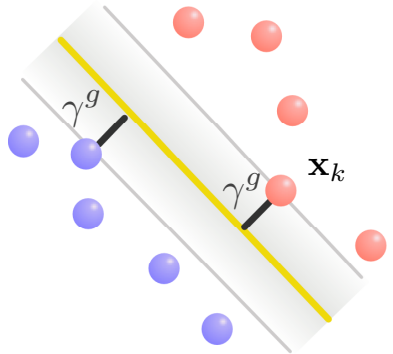
Since a hyperplane is invariant under scaling by a constant c, for any hyperplane $(\mathbf{w}, b)$, we can find c such that

$$\gamma^f \equiv \underbrace{\gamma_k^f = d_k \left(c\,\mathbf{w}^T\mathbf{x}_k + c\,b\right)}_{\text{functional margin of example } k} = 1$$

That is, we fix functional margin γ^f of training set to be 1.

Problem: Find c such that $\gamma_k^f = 1$, with

$$\mathbf{x}_k = \begin{bmatrix} 1 \\ 4 \end{bmatrix}, \quad d_k = +1\,, \quad \mathbf{w} = \begin{bmatrix} 5 \\ 3 \end{bmatrix}, \quad b = 6$$

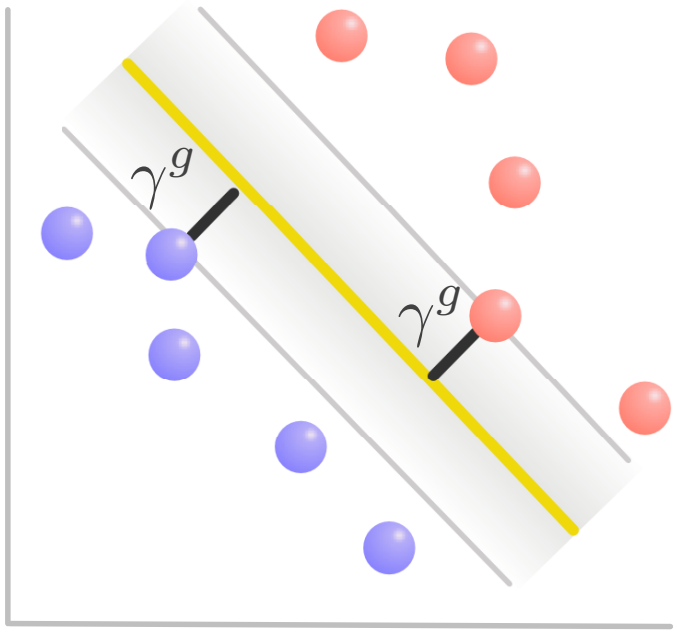


$$d_k \left(c\,\mathbf{w}^T\mathbf{x}_k + c\,b\right) = 1 \cdot c \left(\begin{bmatrix} 5 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 4 \end{bmatrix} + 6 \right) = 23\,c = 1 \;\Rightarrow\; c = \frac{1}{23}$$

$\gamma^g = \frac{\gamma^f}{\|\mathbf{w}\|}$: We can maximize γ^g by fixing γ^f then minimizing $\|\mathbf{w}\|$

With γ^f fixed at 1, for any example $\mathbf{x}_i$, we have the condition

$$\gamma_i^f = \gamma_i^g \|\mathbf{w}\| = d_i \left(\mathbf{w}^T \mathbf{x}_i + b\right) \geq 1$$



$\gamma^g = \frac{\gamma^f}{\|\mathbf{w}\|}$: We can maximize γ^g by fixing γ^f then minimizing $\|\mathbf{w}\|$

With γ^f fixed at 1, for any example $\mathbf{x}_i$, we have the condition

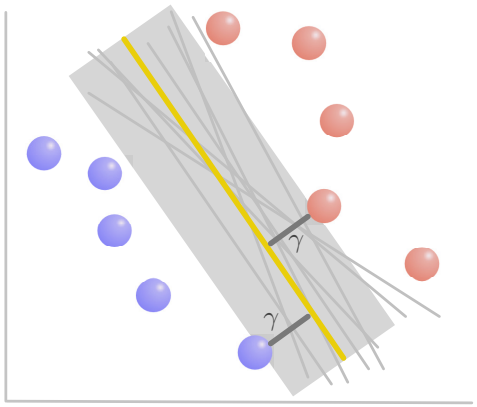
$$\gamma_i^f = \gamma_i^g \|\mathbf{w}\| = d_i \left(\mathbf{w}^T \mathbf{x}_i + b\right) \geq 1$$

The problem of minimizing $\|\mathbf{w}\|$ can now be expressed as a

Constrained optimization problem

$$\text{Minimizing:} \quad f(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 = \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

$$\text{Subject to:} \quad d_i \left(\mathbf{w}^T \mathbf{x}_i + b\right) \geq 1$$

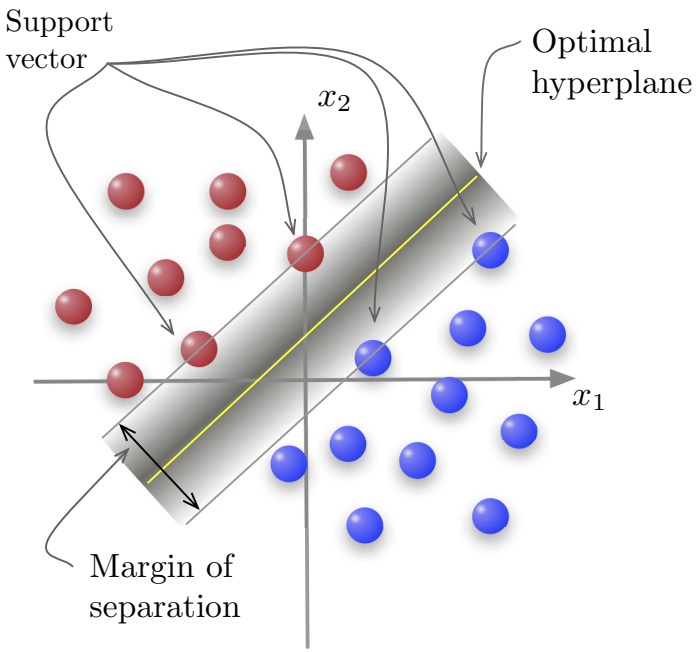


$$\gamma = \max \underbrace{\left\{\gamma^g_{(\mathbf{w}_1,b_1)}, \gamma^g_{(\mathbf{w}_2,b_2)}, \cdots\right\}}_{\text{All possible hyperplanes}}$$

Solving this problem yields the optimal hyperplane $(\mathbf{w}_\circ, b_\circ)$

For training data $\mathbf{x}_i$

$$\begin{cases} g(\mathbf{x}_i) = \mathbf{w}_\circ^T \mathbf{x}_i + b_\circ \geq +1 & \text{for } d_i = +1 \\ g(\mathbf{x}_i) = \mathbf{w}_\circ^T \mathbf{x}_i + b_\circ \leq -1 & \text{for } d_i = -1 \end{cases}$$



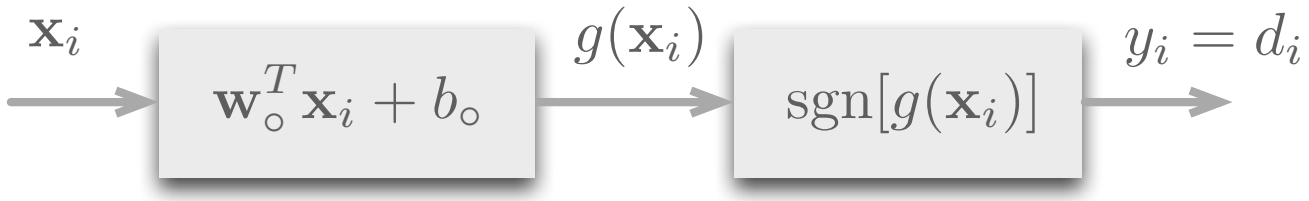
Discriminant function

$$g(\mathbf{x}) = \mathbf{w}_\circ^T \mathbf{x} + b_\circ$$

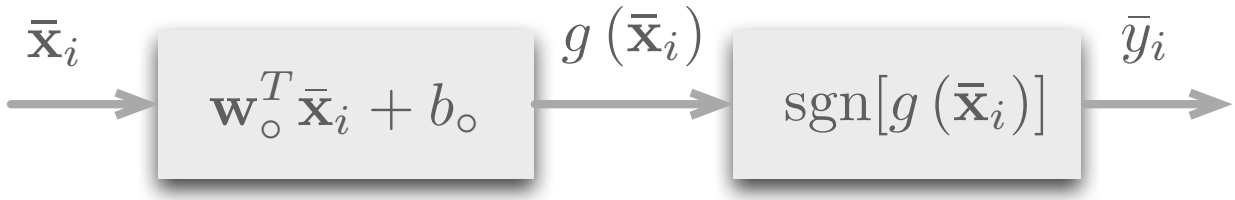
Support vector: $\mathbf{x}_i$ that satisfies

$$g(\mathbf{x}_i) = \pm 1$$

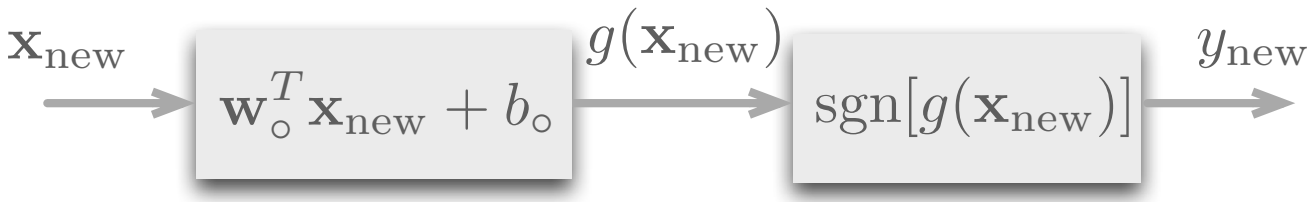
Construction: For a given training set $S = \{(\mathbf{x}_1, d_1), \ldots, (\mathbf{x}_N, d_N)\}$, find optimal hyperplane $(\mathbf{w}_\circ, b_\circ)$ such that, for all $i \in \{1, 2, \ldots, N\}$,



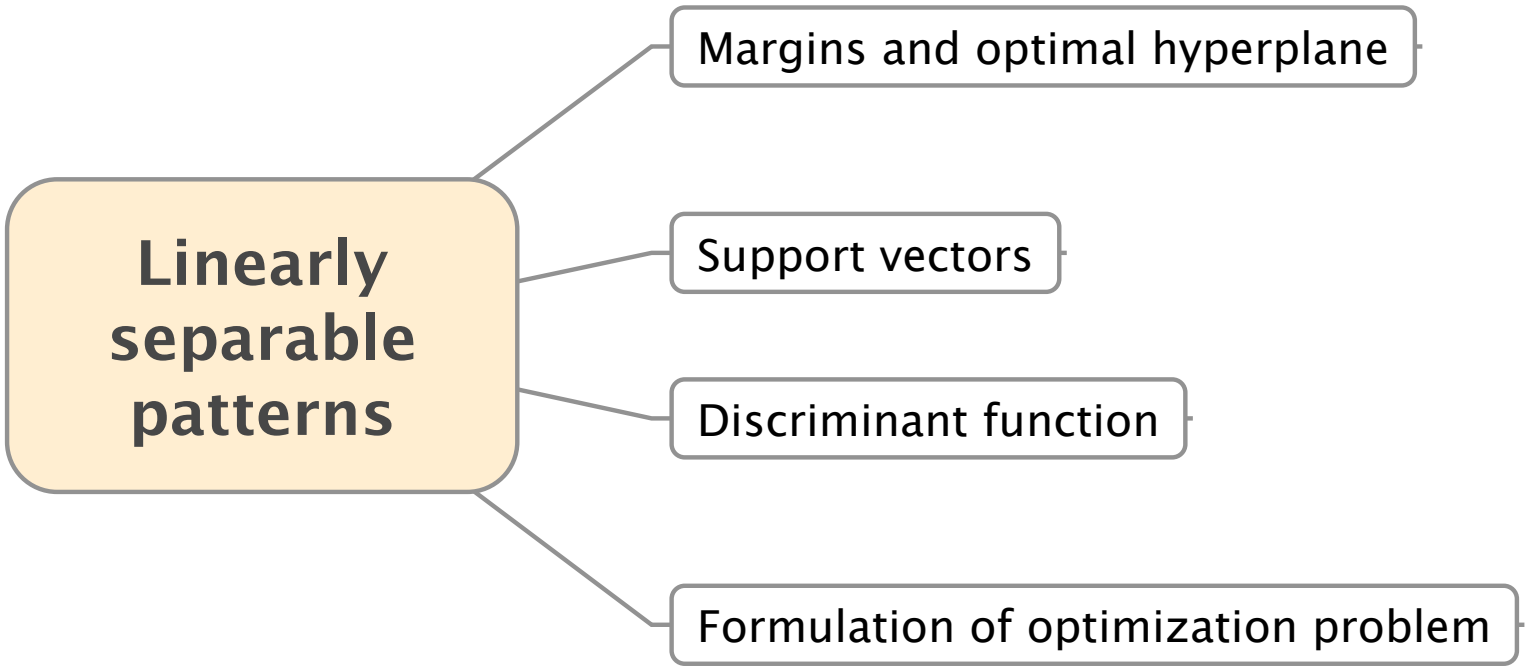
Testing: For a given test set $\bar{S} = \{(\bar{\mathbf{x}}_1, \bar{d}_1), \ldots, (\bar{\mathbf{x}}_{\bar{N}}, \bar{d}_{\bar{N}})\}$, compute output $\bar{y}_i$ of SVM (with $\mathbf{w}_\circ$ and $b_\circ$) for all $i \in \{1, 2, \ldots, \bar{N}\}$, and compare it against the known $\bar{d}_i$ to evaluate performance of SVM



Application: Given a SVM with hyperplane $(\mathbf{w}_\circ, b_\circ)$, classify a data point $\mathbf{x}_{\text{new}}$ that is not in $\Sigma = S \cup \bar{S}$:



SVM for linearly separable patterns: Summary



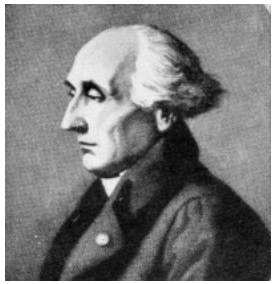
Primal problem

$$\begin{aligned} \text{Given data set :} \quad & S = \{(\mathbf{x}_i, d_i)\}, \ i = 1, 2, \ldots, N \\ \text{Find :} \quad & \mathbf{w} \text{ and } b \\ \text{Minimizing :} \quad & f(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} \\ \text{Subject to :} \quad & d_i\left(\mathbf{w}^T\mathbf{x}_i + b\right) \geq 1 \end{aligned}$$

Well known algorithms exist for solving this problem

- R., Fletcher, *Practical Methods of Optimization*. 2nd ed., John Wiley & Sons, NY, 1987
- Arnold Neumaier, Complete search in continuous global optimization and constraint satisfaction. *Acta Numerica*, Vol. 13, 271-369, 2004

Alternative formulation using method of Lagrange multipliers



Joseph-Louis Lagrange
French mathematician
1736–1813

Basic class of constrained optimization problems

$$\begin{aligned} \text{Minimize}: \quad & f(\mathbf{w}), \ \mathbf{w} \in \mathbb{R}^n, \ f(\cdot), h(\cdot) \in \mathbb{R} \\ \text{Subject to}: \quad & h_i(\mathbf{w}) = 0, \ i = 1, \ldots, m \end{aligned}$$

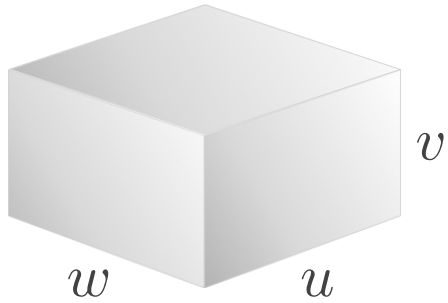
- f is called the objective function and h_i the constraints
- Optimal value of f is called value of optimization problem
- $\mathbf{w}$ corresponding to optimal f is called optimal solution

If f is to be maximized instead, such a maximization problem can be expressed as a minimization problem by the transformation

$$\max_{\mathbf{w}} f(\mathbf{w}) = -\min_{\mathbf{w}} [-f(\mathbf{w})]$$

$$\begin{aligned} \text{Maximize}: \quad & f(\mathbf{w}) \\ \text{Subject to}: \quad & h_i(\mathbf{w}) = 0 \end{aligned} \quad \equiv \quad \begin{aligned} \text{Minimize}: \quad & -f(\mathbf{w}) \\ \text{Subject to}: \quad & h_i(\mathbf{w}) = 0 \end{aligned}$$

Problem: Consider a box whose surface area is C. Formulate the optimization problem for maximizing the volume of the box.



Solution:

$$\begin{aligned} \text{Surface area:} \quad & 2wu + 2uv + 2vw = C \\ \text{Volume:} \quad & wuv \end{aligned}$$

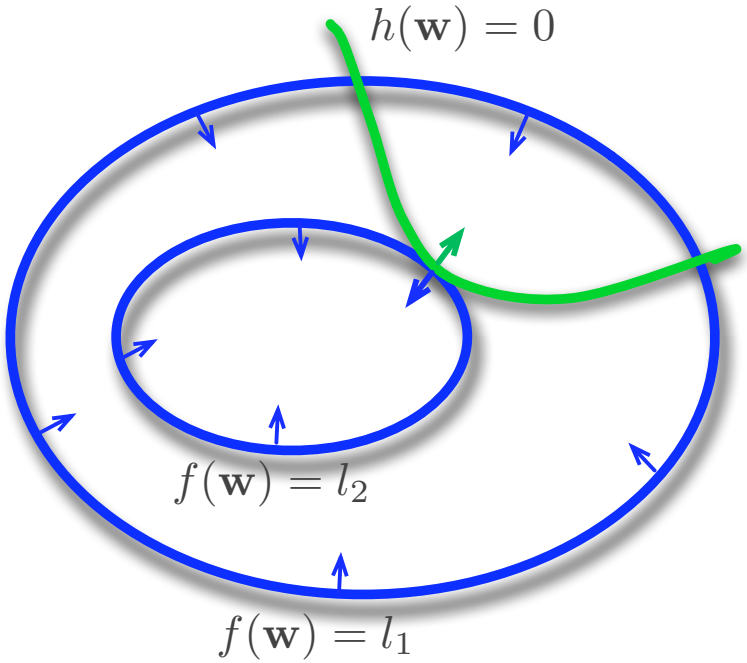
Maximizing wuv is equivalent to minimizing $-wuv$.

Optimization problem

$$\begin{aligned} \text{Minimize:} \quad & f(u, v, w) = -uvw \\ \text{Subject to:} \quad & h(u, v, w) = wu + uv + vw - \frac{C}{2} = 0 \end{aligned}$$

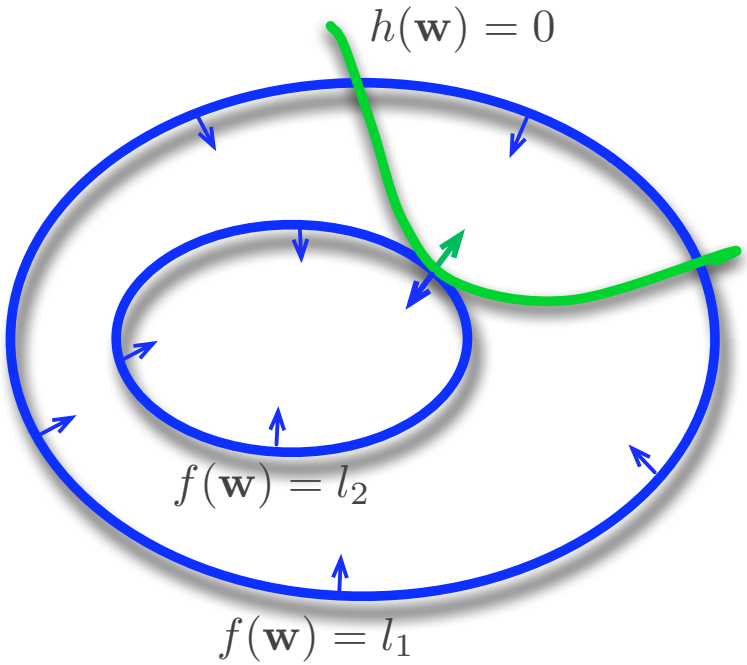
Basic class of constrained optimization problems

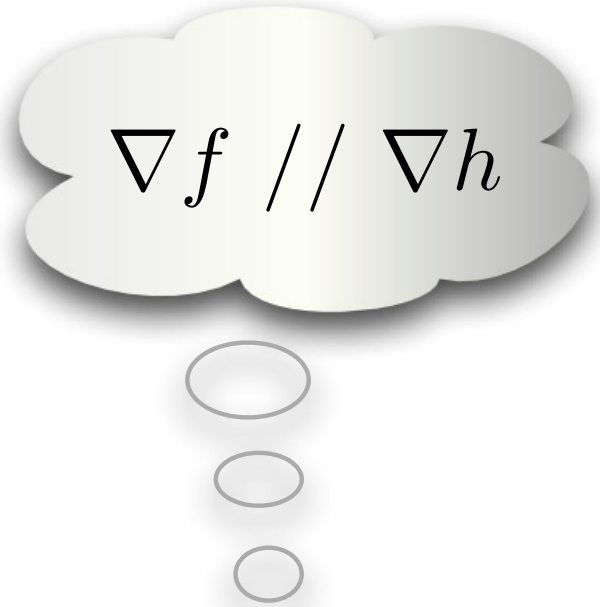
$$\begin{aligned} \text{Minimize}: \quad & f(\mathbf{w}) \\ \text{Subject to}: \quad & h(\mathbf{w}) = 0 \quad \text{(equality constraint)} \end{aligned}$$



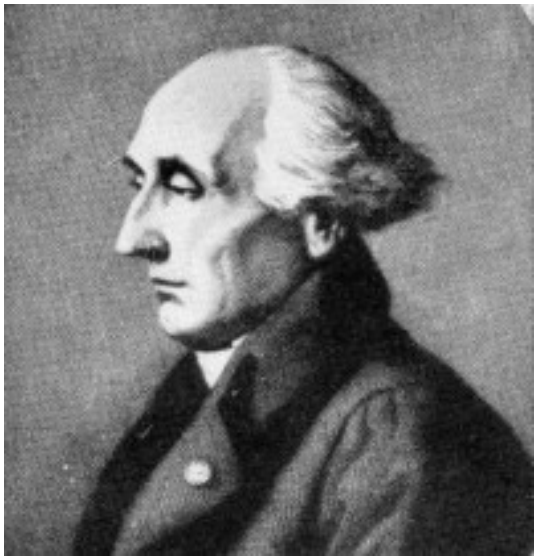
Blue arrow points to direction of greatest decrease of $f(\mathbf{w})$

- Contours of $f(\mathbf{w})$ correspond to fixed l_n, i.e., l_1, l_2, ...
- $h(\mathbf{w}) = 0$ intersects many f contours. Each intersection is a solution $\mathbf{w}$ satisfying h
- Find the solution $\mathbf{w}$ that satisfies h while gives minimum of f

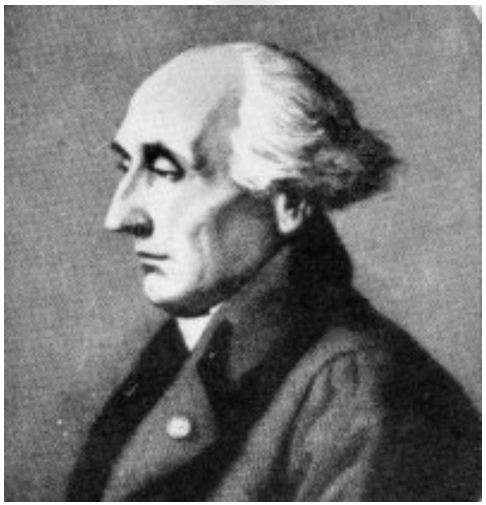




- Solution occurs at point where f and h touch tangentially
- At solution point, gradients of f and h are parallel.



$\nabla f \;//\; \nabla h$



$$\begin{aligned} \text{Minimize :} \quad & f(\mathbf{w}) \\ \text{Subject to :} \quad & h_i(\mathbf{w}) = 0\,,\ i = 1, \ldots, m \end{aligned}$$

$$\nabla f = \frac{\partial f(\mathbf{w})}{\partial \mathbf{w}}$$

$$\nabla h_i = \frac{\partial h_i(\mathbf{w})}{\partial \mathbf{w}}$$

The fact that ∇f and ∇h are parallel means there exist constants β_i such that

$$\left[\frac{\partial f\,(\mathbf{w})}{\partial \mathbf{w}}\right]_{\mathbf{w}_\circ} + \sum_{i=1}^{m} \left[\beta_i \left(\frac{\partial h_i\,(\mathbf{w})}{\partial \mathbf{w}}\right)\right]_{\mathbf{w}_\circ} = \mathbf{0}$$

where $\mathbf{w}_\circ$ is an optimal solution

The constants β_i are called Lagrange multipliers

Define Lagrangian function:

$$L(\mathbf{w}, \boldsymbol{\beta}) = f(\mathbf{w}) + \sum_{i=1}^{m} \beta_i \, h_i(\mathbf{w})$$

Then the fact that $\nabla f // \nabla h$ can be expressed compactly as

$$\left. \frac{\partial L(\mathbf{w}, \boldsymbol{\beta})}{\partial \mathbf{w}} \right|_{\mathbf{w}_\circ} = \mathbf{0}$$

Lagrange Theorem

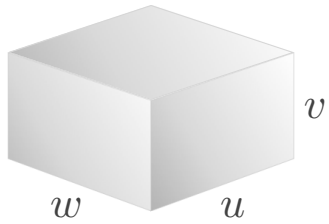
The necessary conditions for the existence of an optimal solution $\mathbf{w} = \mathbf{w}_\circ$ corresponding to a minimum of $f(\mathbf{w})$ subject to $h_i(\mathbf{w}) = 0, i = 1, \ldots, m$, are

$$\begin{aligned} \frac{\partial L(\mathbf{w}, \boldsymbol{\beta})}{\partial \mathbf{w}} &= \mathbf{0} \\ \frac{\partial L(\mathbf{w}, \boldsymbol{\beta})}{\partial \beta_i} &= 0 \end{aligned}$$

Theorem can be used to solve problems with equality constraints

Minimize: $f(u,v,w) = -uvw$

Subject to: $h(u,v,w) = wu + uv + vw - \frac{C}{2} = 0$



$$L(u,v,w) = -uvw + \beta\left(wu + uv + vw - \frac{C}{2}\right)$$

Lagrange's Theorem yields

$$\begin{aligned}
\frac{\partial L}{\partial u} = -vw + \beta(v+w) &= 0 \\
\frac{\partial L}{\partial v} = -wu + \beta(w+u) &= 0 \\
\frac{\partial L}{\partial w} = -uv + \beta(u+v) &= 0 \\
wu + uv + vw &= \frac{C}{2}
\end{aligned}$$

The solution is

$$u = v = w = \sqrt{\frac{C}{6}}$$

and the maximum volume is

$$\begin{aligned}
&\max(wuv) \\
&= -\min(-wuv) \\
&= -\left(-\left(\frac{C}{6}\right)^{\frac{3}{2}}\right) = \left(\frac{C}{6}\right)^{\frac{3}{2}}
\end{aligned}$$

Lagrange Theorem can be generalized to deal with problems having both equality and inequality constraints

Primal problem (Standard form)

$$\begin{aligned} \text{Minimize:} \quad & f(\mathbf{w}) \\ \text{Subject to:} \quad & q_i(\mathbf{w}) \leq 0, \quad i = 1, \ldots, k \\ & h_i(\mathbf{w}) = 0, \quad i = 1, \ldots, m \end{aligned}$$

Constraints in the form $q_i(\mathbf{w}) \geq 0$

can be re-written as $q_i'(\mathbf{w}) = -q_i(\mathbf{w}) \leq 0$

Lagrange Theorem can be generalized to deal with problems having both equality and inequality constraints

Primal problem

$$\begin{aligned} \text{Minimize:} \quad & f(\mathbf{w}) \\ \text{Subject to:} \quad & q_i(\mathbf{w}) \leq 0, \quad i = 1, \ldots, k \\ & h_i(\mathbf{w}) = 0, \quad i = 1, \ldots, m \end{aligned}$$

Generalized Lagrangian function

$$L(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = f(\mathbf{w}) + \sum_{i=1}^{k} \alpha_i \, q_i(\mathbf{w}) + \sum_{i=1}^{m} \beta_i \, h_i(\mathbf{w})$$

where α_i and β_i are the Lagrange multipliers

Solution characterized by Karush-Kuhn-Tucker conditions

Kuhn-Tucker Theorem

Consider the primal problem with the Lagrangian as defined earlier. The sufficient and necessary condition for a point $\mathbf{w} = \mathbf{w}_\circ$ to be an optimal solution is the existence of $\boldsymbol{\alpha} = \boldsymbol{\alpha}_\circ$ and $\boldsymbol{\beta} = \boldsymbol{\beta}_\circ$ such that

$$\begin{aligned}
\frac{\partial L\left(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}\right)}{\partial \mathbf{w}} &= \mathbf{0} \\
\frac{\partial L\left(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}\right)}{\partial \beta_i} &= 0 \\
\alpha_i \, q_i\left(\mathbf{w}\right) &= 0\,, \quad i = 1, \ldots, k \\
q_i\left(\mathbf{w}\right) &\leq 0\,, \quad i = 1, \ldots, k \\
\alpha_i &\geq 0\,, \quad i = 1, \ldots, k
\end{aligned}$$

These equations are the KKT conditions

The KKT conditions enable us to transform the primal problem into an alternative, simpler form called the dual problem

Primal problem

$$\begin{aligned} \text{Minimize:} \quad & f(\mathbf{w}) \\ \text{Subject to:} \quad & q_i(\mathbf{w}) \leq 0 \\ & h_i(\mathbf{w}) = 0 \end{aligned}$$

Dual problem

$$\begin{aligned} \text{Maximize:} \quad & L(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \\ \text{Subject to:} \quad & \frac{\partial L(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})}{\partial \mathbf{w}} = 0 \\ & \boldsymbol{\alpha} \geq 0 \end{aligned}$$

The primal and dual problems are equivalent because they have the same *value of the optimization problem*, i.e.,

$$f(\mathbf{w}_\circ) = L(\mathbf{w}_\circ, \boldsymbol{\alpha}_\circ, \boldsymbol{\beta}_\circ)$$

Proof in *Practical Methods of Optimization* by Fletcher, Wiley, 1987

Finding optimal hyperplane (primal problem)

$$\begin{aligned} \text{Given data set}: &\quad S = \{(\mathbf{x}_i, d_i)\} \\ \text{Find}: &\quad \mathbf{w} \text{ and } b \\ \text{Minimizing}: &\quad f(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} \\ \text{Subject to}: &\quad d_i\left(\mathbf{w}^T\mathbf{x}_i + b\right) \geq 1 \end{aligned}$$

In primal problem

$$\begin{aligned} \text{Known parameters}: &\quad \mathbf{x}_i,\ d_i \\ \text{Unknown variables}: &\quad \mathbf{w},\ b \end{aligned}$$

Apply KKT conditions to reduce unknowns to just α_i to form dual problem

Write constraints $d_i \left(\mathbf{w}^T \mathbf{x}_i + b\right) \geq 1$ in standard form as

$$q_i = -d_i \left(\mathbf{w}^T \mathbf{x}_i + b\right) + 1 \leq 0$$

Now the Lagrangian function is

$$\begin{aligned} L(\mathbf{w}, b, \boldsymbol{\alpha}) &= \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \alpha_i \left(d_i \left(\mathbf{w}^T \mathbf{x}_i + b\right) - 1\right) \\ &= \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \alpha_i \, d_i \, \mathbf{w}^T \mathbf{x}_i - b \sum_{i=1}^{N} \alpha_i \, d_i + \sum_{i=1}^{N} \alpha_i \end{aligned}$$

The KKT conditions are

$$\begin{aligned} &\frac{\partial L(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial \mathbf{w}} = \mathbf{0} \\ &\frac{\partial L(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial b} = 0 \\ &d_i \left(\mathbf{w}^T \mathbf{x}_i + b\right) \geq 1 \\ &\alpha_i \left(d_i \left(\mathbf{w}^T \mathbf{x}_i + b\right) - 1\right) = 0 \\ &\alpha_i \geq 0 \end{aligned}$$

Consider the first KKT condition: $\dfrac{\partial L(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial \mathbf{w}} = \mathbf{0}$

$$\begin{aligned}
\frac{\partial L(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial \mathbf{w}} &= \frac{\partial}{\partial \mathbf{w}} \left(\frac{\mathbf{w}^T \mathbf{w}}{2} - \sum_{i=1}^{N} \alpha_i \, d_i \, \mathbf{w}^T \mathbf{x}_i - b \sum_{i=1}^{N} \alpha_i \, d_i + \sum_{i=1}^{N} \alpha_i \right) \\
&= \frac{1}{2} \frac{\partial \left(\mathbf{w}^T \mathbf{w} \right)}{\partial \mathbf{w}} - \sum_{i=1}^{N} \alpha_i d_i \left(\frac{\partial \left(\mathbf{w}^T \mathbf{x}_i \right)}{\partial \mathbf{w}} \right) \\
&= \frac{2\mathbf{w}}{2} - \sum_{i=1}^{N} \alpha_i d_i \left(\frac{\partial \left(\mathbf{x}_i^T \mathbf{w} \right)}{\partial \mathbf{w}} \right) \quad \left(\text{Since } \mathbf{u}^T \mathbf{v} = \mathbf{v}^T \mathbf{u} \right) \\
&= \mathbf{w} - \sum_{i=1}^{N} \alpha_i d_i \mathbf{x}_i \quad \left(\text{Since } \frac{\partial \left(\mathbf{u}^T \mathbf{v} \right)}{\partial \mathbf{v}} = \mathbf{u} \right) \\
&= \mathbf{0}
\end{aligned}$$

Therefore

▸ View

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

Consider the second KKT condition: $\dfrac{\partial L(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial b} = 0$

$$\begin{aligned} & \frac{\partial L(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial b} \\ = \quad & \frac{\partial}{\partial b}\left(\frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N}\alpha_i\, d_i\, \mathbf{w}^T\mathbf{x}_i - b\sum_{i=1}^{N}\alpha_i\, d_i + \sum_{i=1}^{N}\alpha_i\right) \\ = \quad & \sum_{i=1}^{N}\alpha_i\, d_i = 0 \end{aligned}$$

Therefore

$$\sum_{i=1}^{N}\alpha_i\, d_i = 0$$

Thus from KKT conditions we have

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i \, d_i \, \mathbf{x}_i \qquad \text{and} \qquad \sum_{i=1}^{N} \alpha_i \, d_i = 0$$

Recall that

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \alpha_i \, d_i \, \mathbf{w}^T\mathbf{x}_i - b\sum_{i=1}^{N} \alpha_i \, d_i + \sum_{i=1}^{N} \alpha_i$$

So

$$\frac{1}{2}\mathbf{w}^T\mathbf{w} = \frac{1}{2}\left[\sum_{i=1}^{N} \alpha_i \, d_i \, \mathbf{x}_i^T\right]\left[\sum_{j=1}^{N} \alpha_j \, d_j \, \mathbf{x}_j\right] = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \, \alpha_j \, d_i \, d_j \, \mathbf{x}_i^T\mathbf{x}_j$$

$$\sum_{i=1}^{N} \alpha_i \, d_i \, \mathbf{w}^T\mathbf{x}_i = \sum_{i=1}^{N} \alpha_i \, d_i \left[\sum_{j=1}^{N} \alpha_j \, d_j \, \mathbf{x}_j^T\right]\mathbf{x}_i = \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \, \alpha_j \, d_i \, d_j \, \mathbf{x}_i^T\mathbf{x}_j$$

Transforming primal problem into dual problem by KKT conditions

$$\frac{1}{2}\mathbf{w}^T\mathbf{w} = \frac{1}{2}\left[\sum_{i=1}^{N}\alpha_i\, d_i\, \mathbf{x}_i^T\right]\left[\sum_{j=1}^{N}\alpha_j\, d_j\, \mathbf{x}_j\right] = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\,\alpha_j\, d_i\, d_j\, \mathbf{x}_i^T\mathbf{x}_j$$

$$\sum_{i=1}^{N}\alpha_i\, d_i\, \mathbf{w}^T\mathbf{x}_i = \sum_{i=1}^{N}\alpha_i\, d_i\left[\sum_{j=1}^{N}\alpha_j\, d_j\, \mathbf{x}_j^T\right]\mathbf{x}_i = \sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\,\alpha_j\, d_i\, d_j\, \mathbf{x}_i^T\mathbf{x}_j$$

$$\sum_{i=1}^{N}\alpha_i\, d_i = 0$$

$$\begin{aligned}
L(\mathbf{w}, b, \boldsymbol{\alpha}) &= \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N}\alpha_i\left(d_i\left(\mathbf{w}^T\mathbf{x}_i + b\right) - 1\right) \\
&= \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N}\alpha_i\, d_i\, \mathbf{w}^T\mathbf{x}_i - b\sum_{i=1}^{N}\alpha_i\, d_i + \sum_{i=1}^{N}\alpha_i \\
&= \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\,\alpha_j\, d_i\, d_j\, \mathbf{x}_i^T\mathbf{x}_j \\
&\equiv Q(\boldsymbol{\alpha})
\end{aligned}$$

Finding optimal hyperplane (dual problem)

$$\text{Given :} \quad S = \{(\mathbf{x}_i, d_i)\}$$

$$\text{Find :} \quad \text{Lagrange multipliers } \{\alpha_i\}$$

$$\text{Maximizing :} \quad Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \, \alpha_j \, d_i \, d_j \, \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{Subject to :} \quad (1) \quad \sum_{i=1}^{N} \alpha_i \, d_i = 0$$

$$(2) \quad \alpha_i \geq 0$$

- α_i are the only unknowns
- Algorithms and software available for solving this problem
- Optimal solution denoted by $\alpha_{\circ,i}$
- $\mathbf{x}_i^T \mathbf{x}_j$ is called a linear kernel (more about kernels later)

Suppose that a support vector machine is to be constructed using the training set below:

i	$\mathbf{x}_i$	d_i
1	$[\ 1 \quad -1\ ]^T$	-1
2	$[\ 2 \quad 1\ ]^T$	1
3	$[\ 3 \quad 1\ ]^T$	1

(i) Determine the Lagrangian function L

(ii) Show the explicit form of the dual problem

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \alpha_i \left(d_i \left(\mathbf{w}^T\mathbf{x}_i + b\right) - 1\right)$$

From the given data, we have $N = 3$, and $\mathbf{w} = [w_1, w_2]^T$. Now

$$\begin{aligned}
L(\mathbf{w}, b, \boldsymbol{\alpha}) &= \frac{1}{2}\begin{bmatrix} w_1 & w_2 \end{bmatrix}\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \\
&\quad - \alpha_1(-1)\begin{bmatrix} w_1 & w_2 \end{bmatrix}\begin{bmatrix} 1 \\ -1 \end{bmatrix} - \alpha_1(-1)b + \alpha_1 \\
&\quad - \alpha_2(1)\begin{bmatrix} w_1 & w_2 \end{bmatrix}\begin{bmatrix} 2 \\ 1 \end{bmatrix} - \alpha_2(1)b + \alpha_2 \\
&\quad - \alpha_3(1)\begin{bmatrix} w_1 & w_2 \end{bmatrix}\begin{bmatrix} 3 \\ 1 \end{bmatrix} - \alpha_3(1)b + \alpha_3 \\
&= \frac{1}{2}\left(w_1^2 + w_2^2\right) + \alpha_1\left(w_1 - w_2 + 1\right) - \alpha_2\left(2w_1 + w_2 - 1\right) \\
&\quad -\alpha_3\left(3w_1 + w_2 - 1\right) + \left(\alpha_1 - \alpha_2 - \alpha_3\right)b
\end{aligned}$$

$$Q(\boldsymbol{\alpha}) = \sum_{i=1}^{3} \alpha_i - \frac{1}{2} \sum_{i=1}^{3} \sum_{j=1}^{3} \alpha_i \, d_i \, \alpha_j \, d_j \, \mathbf{x}_i^T \mathbf{x}_j$$

$$\begin{aligned}
= \quad & \alpha_1 + \alpha_2 + \alpha_3 - \frac{1}{2}\alpha_1 d_1 \alpha_1 d_1 \mathbf{x}_1^T \mathbf{x}_1 - \frac{1}{2}\alpha_1 d_1 \alpha_2 d_2 \mathbf{x}_1^T \mathbf{x}_2 - \frac{1}{2}\alpha_1 d_1 \alpha_3 d_3 \mathbf{x}_1^T \mathbf{x}_3 \\
& - \frac{1}{2}\alpha_2 d_2 \alpha_1 d_1 \mathbf{x}_2^T \mathbf{x}_1 - \frac{1}{2}\alpha_2 d_2 \alpha_2 d_2 \mathbf{x}_2^T \mathbf{x}_2 - \frac{1}{2}\alpha_2 d_2 \alpha_3 d_3 \mathbf{x}_2^T \mathbf{x}_3 \\
& - \frac{1}{2}\alpha_3 d_3 \alpha_1 d_1 \mathbf{x}_3^T \mathbf{x}_1 - \frac{1}{2}\alpha_3 d_3 \alpha_2 d_2 \mathbf{x}_3^T \mathbf{x}_2 - \frac{1}{2}\alpha_3 d_3 \alpha_3 d_3 \mathbf{x}_3^T \mathbf{x}_3 \\
= \quad & \alpha_1 + \alpha_2 + \alpha_3 \\
& - \frac{1}{2}\alpha_1(-1)\alpha_1(-1) \begin{bmatrix} 1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} - \frac{1}{2}\alpha_1(-1)\alpha_2(1) \begin{bmatrix} 1 & -1 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \end{bmatrix} \\
& - \frac{1}{2}\alpha_1(-1)\alpha_3(1) \begin{bmatrix} 1 & -1 \end{bmatrix} \begin{bmatrix} 3 \\ 1 \end{bmatrix} - \frac{1}{2}\alpha_2(1)\alpha_1(-1) \begin{bmatrix} 2 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} \\
& - \frac{1}{2}\alpha_2(1)\alpha_2(1) \begin{bmatrix} 2 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \end{bmatrix} - \frac{1}{2}\alpha_2(1)\alpha_3(1) \begin{bmatrix} 2 & 1 \end{bmatrix} \begin{bmatrix} 3 \\ 1 \end{bmatrix} \\
& - \frac{1}{2}\alpha_3(1)\alpha_1(-1) \begin{bmatrix} 3 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} - \frac{1}{2}\alpha_3(1)\alpha_2(1) \begin{bmatrix} 3 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \end{bmatrix} \\
& - \frac{1}{2}\alpha_3(1)\alpha_3(1) \begin{bmatrix} 3 & 1 \end{bmatrix} \begin{bmatrix} 3 \\ 1 \end{bmatrix} \\
= \quad & \alpha_1 + \alpha_2 + \alpha_3 - \frac{1}{2}\left(2\alpha_1^2 + 5\alpha_2^2 + 10\alpha_3^2 - 2\alpha_1\alpha_2 - 4\alpha_1\alpha_3 + 14\alpha_2\alpha_3\right)
\end{aligned}$$

Explicit form of dual problem

Maximize

$$\begin{aligned} Q(\alpha) &= \alpha_1 + \alpha_2 + \alpha_3 \\ &\quad -\frac{1}{2}\left(2\alpha_1^2 + 5\alpha_2^2 + 10\alpha_3^2 - 2\alpha_1\alpha_2 - 4\alpha_1\alpha_3 + 14\alpha_2\alpha_3\right) \end{aligned}$$

Subject to

$$\begin{aligned} \sum_{i=1}^{3} \alpha_i\, d_i &= \alpha_1(-1) + \alpha_2(1) + \alpha_3(1) = -\alpha_1 + \alpha_2 + \alpha_3 = 0 \\ \alpha_1 &\geq 0 \\ \alpha_2 &\geq 0 \\ \alpha_3 &\geq 0 \end{aligned}$$

Recall that for support vector $\mathbf{x}_i$: $g(\mathbf{x}_i) = \mathbf{w}_\circ^T \mathbf{x}_i + b_\circ = \pm 1$

Constraint on $\mathbf{w}_\circ$ and $b_\circ$: $d_i \left(\mathbf{w}_\circ^T \mathbf{x}_i + b_\circ\right) \geq 1$

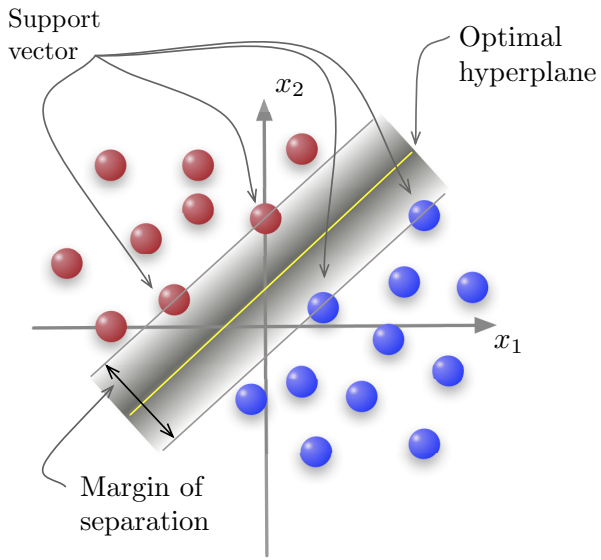
From KKT condition: $\alpha_{\circ,i} \left(d_i \left(\mathbf{w}_\circ^T \mathbf{x}_i + b_\circ\right) - 1\right) = 0$

- For data point $\mathbf{x}_i$ that is not a support vector

$$d_i \left(\mathbf{w}_\circ^T \mathbf{x}_i + b_\circ\right) > 1 \quad \Rightarrow \quad \alpha_{\circ,i} = 0$$

- For data point $\mathbf{x}_i$ that is a support vector

$$d_i \left(\mathbf{w}_\circ^T \mathbf{x}_i + b_\circ\right) = 1 \quad \Rightarrow \quad \alpha_{\circ,i} \neq 0$$



Furthermore, for a support vector $\mathbf{x}_i$

$$d_i \left(\mathbf{w}_\circ^T \mathbf{x}_i + b_\circ\right) = 1 \quad \Rightarrow \quad b_\circ = \frac{1}{d_i} - \mathbf{w}_\circ^T \mathbf{x}_i$$

	Primal	Dual
Find :	$\mathbf{w}\,, b$	α_i
Minimizing :	$f(\mathbf{w})$	$\|\mathbf{w}\|$
Maximizing :	$-$	$Q(\boldsymbol{\alpha})$
Subject to :	$d_i\left(\mathbf{w}^T\mathbf{x}_i + b\right) \geq 1$	$\sum_{i=1}^{N} \alpha_i d_i = 0$
		$\alpha_i \geq 0$

$$f\left(\mathbf{w}\right) = \frac{1}{2}\mathbf{w}^T\mathbf{w} \qquad Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\,\alpha_j\,d_i\,d_j\,\mathbf{x}_i^T\mathbf{x}_j$$

After $\alpha_{\circ,i}$ is obtained, we can calculate $\mathbf{w}_\circ$ and $b_\circ$ as follows:

$$\mathbf{w}_\circ = \sum_{i=1}^{N}\alpha_{\circ,i}\,d_i\,\mathbf{x}_i\,, \quad b_\circ = \frac{1}{d^{(s)}} - \mathbf{w}_\circ^T\mathbf{x}^{(s)}$$

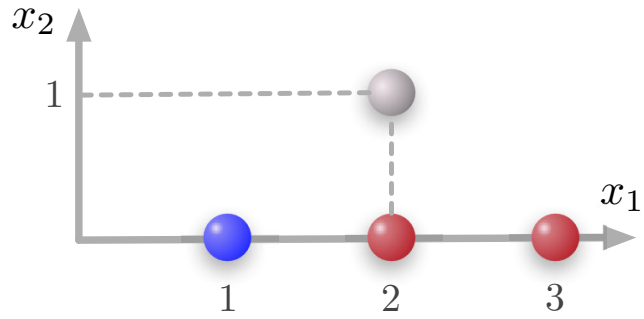
where $\mathbf{x}^{(s)}$ is a support vector with label $d^{(s)}$

Problem: Consider the training set and a support vector machine having the Lagrange multiplier values as listed in the table below:

i	$\alpha_{\circ,i}$	$\mathbf{x}_i$	d_i
1	2	$[1\ 0]^T$	-1
2	2	$[2\ 0]^T$	$+1$
3	0	$[3\ 0]^T$	$+1$

Compute the output of this support vector machine when the input is

$$\begin{bmatrix} 2 \\ 1 \end{bmatrix}$$



Note that $\mathbf{x}_1$ and $\mathbf{x}_2$ are support vectors but $\mathbf{x}_3$ is not. (Why?)

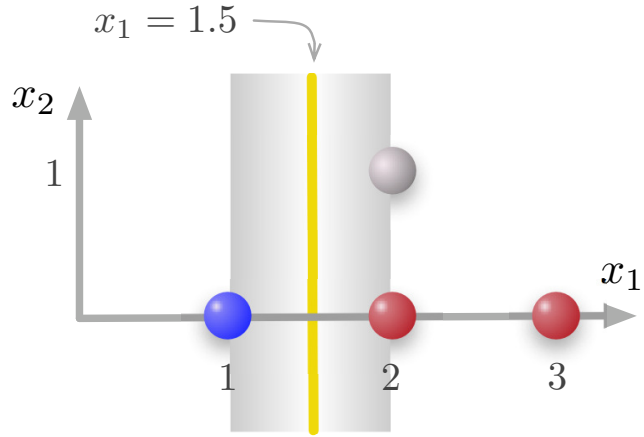
Solution: Given input $\mathbf{x}$, output of SVM is

$$\text{sgn}[g(\mathbf{x})]$$

where

$$g(\mathbf{x}) = \mathbf{w}_\circ^T \mathbf{x} + b_\circ$$

is the discriminant function. Now



$$\begin{aligned}
\mathbf{w}_\circ &= \sum_{i=1}^{3} \alpha_{\circ,i} d_i \mathbf{x}_i = \alpha_{\circ,1} d_1 \mathbf{x}_1 + \alpha_{\circ,2} d_2 \mathbf{x}_2 + \alpha_{\circ,3} d_3 \mathbf{x}_3 \\
&= (2)(-1) \begin{bmatrix} 1 \\ 0 \end{bmatrix} + (2)(1) \begin{bmatrix} 2 \\ 0 \end{bmatrix} + (0)(1) \begin{bmatrix} 3 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \end{bmatrix}
\end{aligned}$$

To calculate $b_\circ$, we pick the support vector $\mathbf{x}_1^{(s)}$ which has $d_1 = -1$. So

$$b_\circ = \frac{1}{d_1} - \mathbf{w}_\circ^T \mathbf{x}_1^{(s)} = \frac{1}{-1} - \begin{bmatrix} 2 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = -3$$

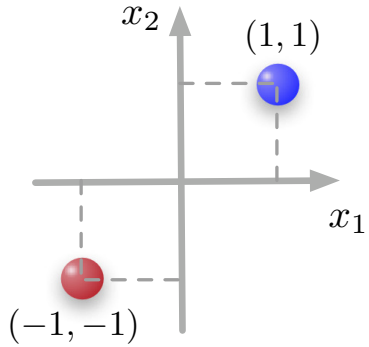
Thus, for the input $\mathbf{x} = [2 \;\; 1]^T$, the output of the support vector machine is

$$\text{sgn}\left[g(\mathbf{x})\right] = \text{sgn}\left[\mathbf{w}_\circ^T \mathbf{x} + b_\circ\right] = \text{sgn}\left[\begin{bmatrix} 2 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \end{bmatrix} - 3\right] = \text{sgn}[4 - 3] = 1$$

Given the training set below, find $\mathbf{w}_\circ$ and $b_\circ$.

$$\mathbf{x}_1 = \begin{bmatrix} -1 \\ -1 \end{bmatrix} \qquad \mathbf{x}_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$d_1 = -1 \qquad d_2 = +1$$



$$\begin{aligned} Q(\boldsymbol{\alpha}) &= \sum_{i=1}^{2} \alpha_i - \frac{1}{2}\left(\sum_{i=1}^{2}\sum_{j=1}^{2} \alpha_i \alpha_j d_i d_j \mathbf{x}_i^T \mathbf{x}_j\right) \\ &= \alpha_1 + \alpha_2 - \frac{1}{2}\left(\alpha_1\alpha_1(-1)(-1)[-1\ -1]\begin{bmatrix} -1 \\ -1 \end{bmatrix} + \right. \\ &\qquad \left. \alpha_1\alpha_2(-1)(1)[-1\ -1]\begin{bmatrix} 1 \\ 1 \end{bmatrix} + \cdots\right) \\ &= \alpha_1 + \alpha_2 - \frac{1}{2}\left(2\alpha_1^2 + 4\alpha_1\alpha_2 + 2\alpha_2^2\right) \\ &= \alpha_1 + \alpha_2 - (\alpha_1 + \alpha_2)^2 \end{aligned}$$

Constrained optimization problem

$$\text{Maximizing :} \quad Q(\boldsymbol{\alpha}) = \alpha_1 + \alpha_2 - (\alpha_1 + \alpha_2)^2$$

$$\text{Subject to :} \quad \sum_{i=1}^{2} \alpha_i d_i = -\alpha_1 + \alpha_2 = 0$$

$$\alpha_1 \geq 0\,, \;\; \alpha_2 \geq 0$$

Solution from *Mathematica* with *Global Optimization* Package

```
In[1]=<<GO52A‘          (* Load Global Optimization package *)
In[2]=GlobalSearch[     (* Algorithm that finds minimum *)
-(a1+a2-(a1+*a2)^2),    (* -Q to be minimized *)
{-a1,-a2},              (* a1 >= 0, a2 >= 0 *)
{-a1+a2},               (* -a1 + a2 = 0 *)
{{a1,0,1},{a2,0,1}},    (* Range of variable *)
0.00001,                (* Tolerance *)
Starts->1]              (* Number of starting points for search *)
Out[2]={{{a1->0.25,a2->0.25},-0.25}}  (* Solution *)
```

$(* \ldots *)$ indicates comment

$$\begin{aligned} \text{Maximizing}: \quad & Q(\boldsymbol{\alpha}) = \alpha_1 + \alpha_2 - (\alpha_1 + \alpha_2)^2 \\ \text{Subject to}: \quad & -\alpha_1 + \alpha_2 = 0 \\ & \alpha_1 \geq 0\,,\ \alpha_2 \geq 0 \end{aligned}$$

For optimal solution, we have

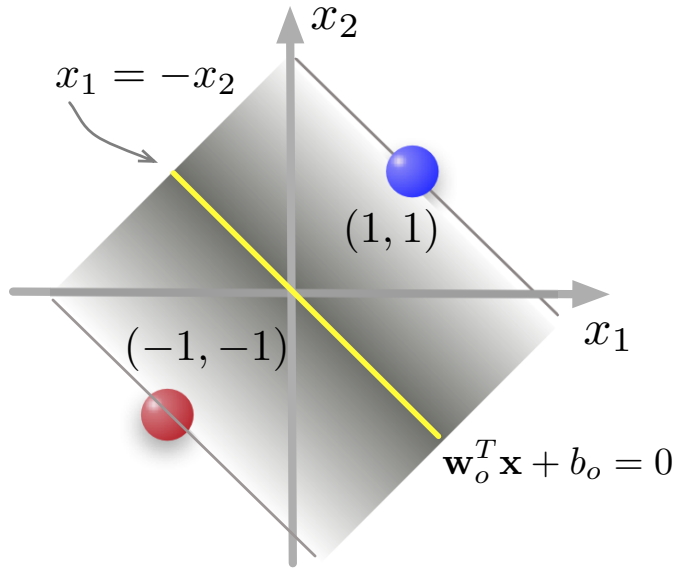
$$\frac{\partial Q(\boldsymbol{\alpha})}{\partial \alpha_1} = \frac{\partial Q(\boldsymbol{\alpha})}{\partial \alpha_2} = 0$$

$$\left.\begin{aligned} \frac{\partial Q(\boldsymbol{\alpha})}{\partial \alpha_1} &= 1 - 2\,(\alpha_1 + \alpha_2) = 0 \\ \frac{\partial Q(\boldsymbol{\alpha})}{\partial \alpha_2} &= 1 - 2\,(\alpha_1 + \alpha_2) = 0 \end{aligned}\right\} \quad \alpha_1 + \alpha_2 = \frac{1}{2} \quad \boxed{1}$$

$$\left.\sum_{i=1}^{2} \alpha_i d_i = 0\right\} \quad -\alpha_1 + \alpha_2 = 0 \quad \boxed{2}$$

Solving 1 and 2 yields

$$\alpha_{\circ,1} = \alpha_{\circ,2} = \frac{1}{4}$$



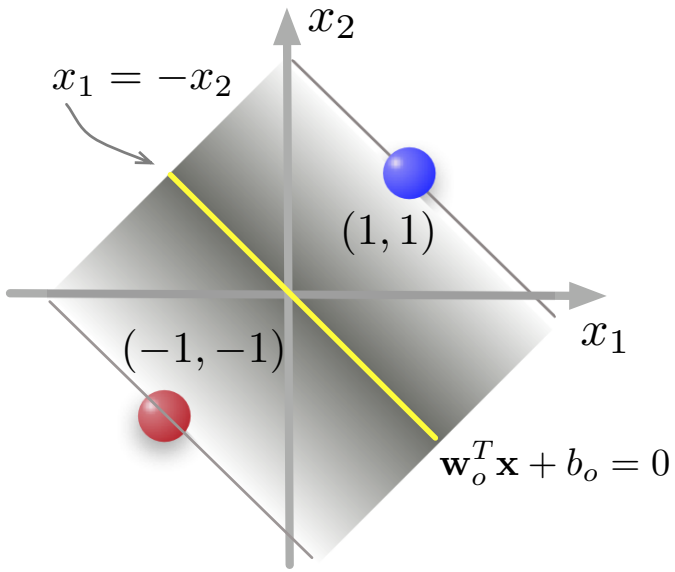
$$\begin{aligned}\mathbf{w}_\circ &= \sum_{i=1}^{N} \alpha_{\circ,i} d_i \mathbf{x}_i = \alpha_{\circ,1} d_1 \mathbf{x}_1 + \alpha_{\circ,2} d_2 \mathbf{x}_2 \\ &= \frac{1}{4}(-1)\begin{bmatrix} -1 \\ -1 \end{bmatrix} + \frac{1}{4}(1)\begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}\end{aligned}$$

To find $b_\circ$, we can take support vector $\mathbf{x}_2$ with $d_2 = 1$. So

$$b_\circ = \frac{1}{d^{(s)}} - \mathbf{w}_\circ^T \mathbf{x}^{(s)} = \frac{1}{d_2} - \mathbf{w}_\circ^T \mathbf{x}_2 = 1 - [0.5\ 0.5]\begin{bmatrix} 1 \\ 1 \end{bmatrix} = 0$$

Classify a new data point $\mathbf{x}_{\text{new}}$ as $+1$ or -1 based on

$$\begin{aligned} d_{\text{new}} &= \text{sgn}\left[g(\mathbf{x}_{\text{new}})\right] \\ &= \text{sgn}\left[\mathbf{w}_\circ^T \mathbf{x}_{\text{new}} + b_\circ\right] \end{aligned}$$



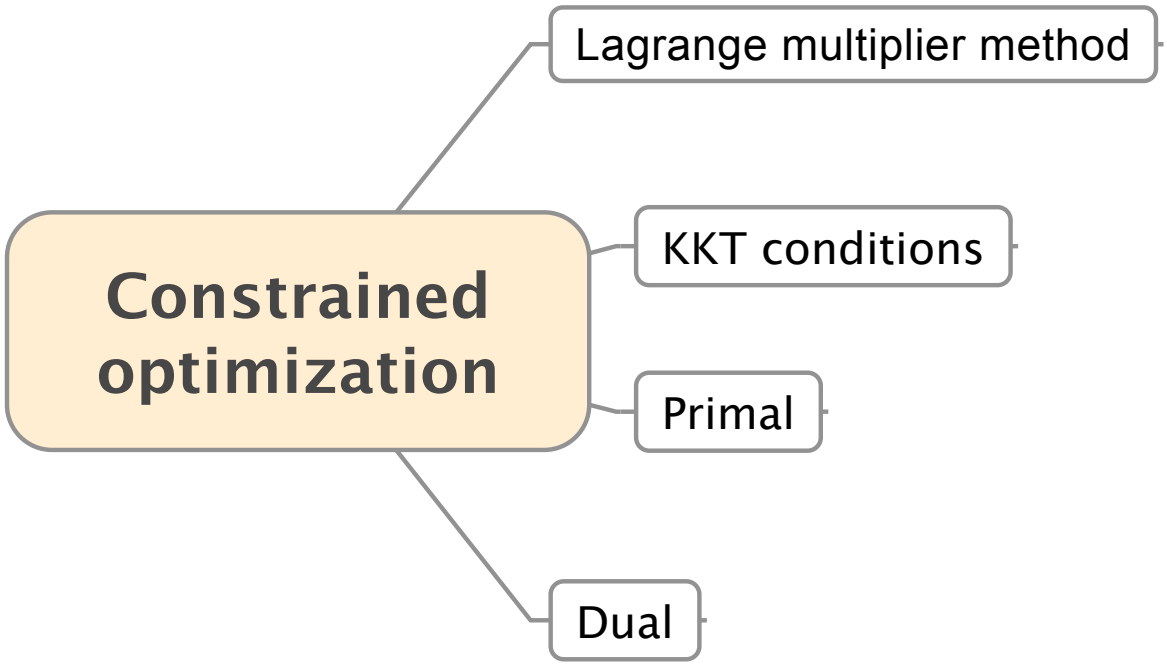
Example: Given a SVM with

$$\mathbf{w}_\circ = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}, \; b_\circ = 0$$

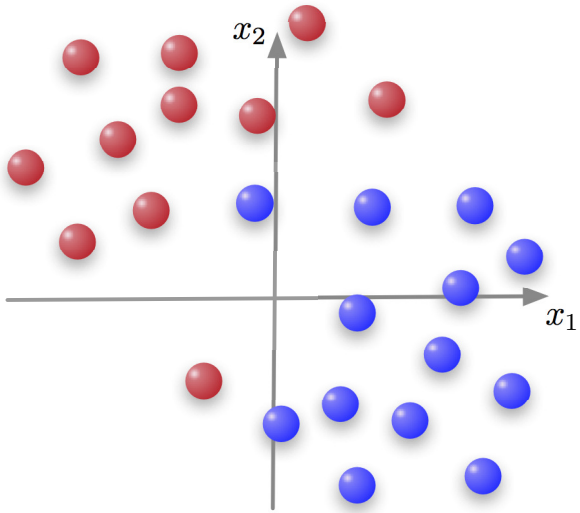
To classify $\mathbf{x}_{\text{new}} = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$

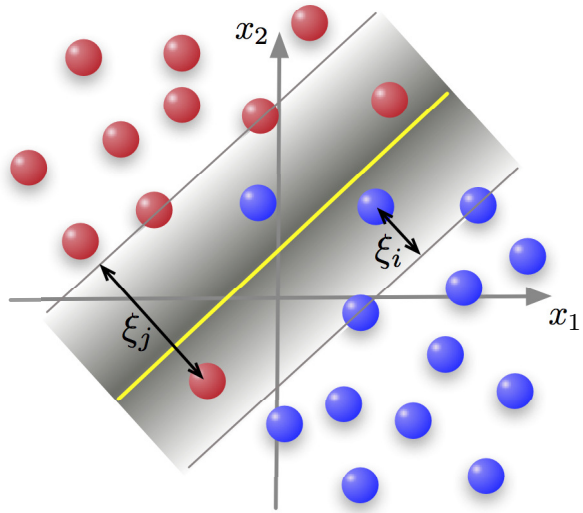
$$\begin{aligned} d_{\text{new}} &= \text{sgn}\left[g(\mathbf{x}_{\text{new}})\right] \\ &= \text{sgn}\left[\mathbf{w}_\circ^T \mathbf{x}_{\text{new}} + b_\circ\right] \\ &= \text{sgn}\left[[0.5 \; 0.5] \begin{bmatrix} 2 \\ 2 \end{bmatrix}\right] \\ &= +1 \end{aligned}$$

Constrained optimization by Lagrange multiplier method: Summary

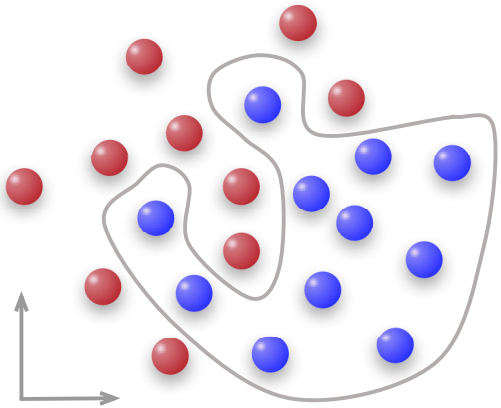


1. Find optimal hyperplane to minimize classification error

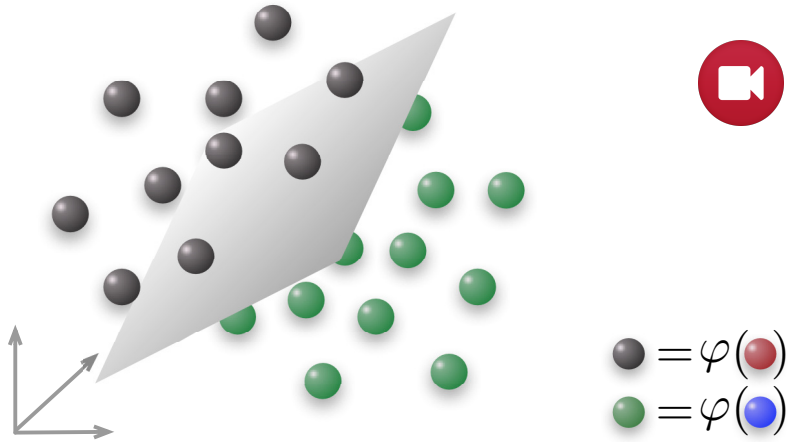




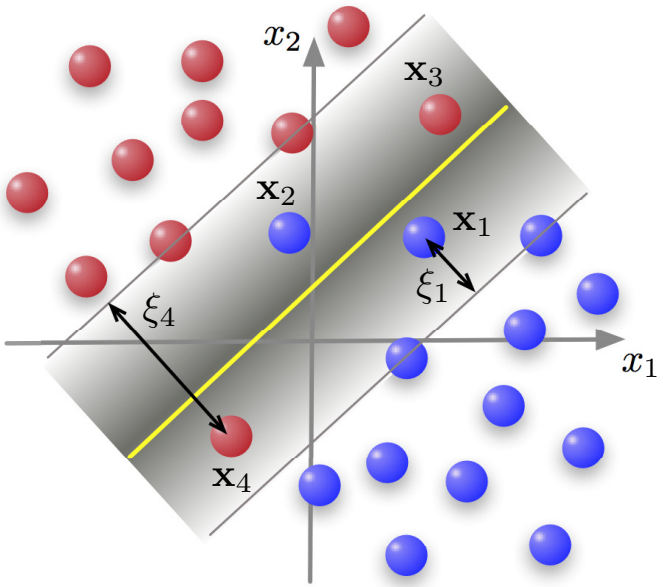
2. Transform data into higher dimension space for separation



Input space



Feature space



Nonnegative slack variables

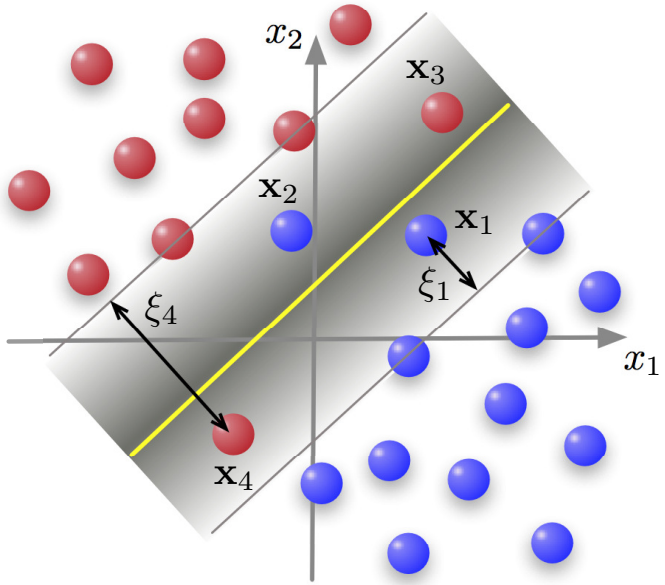
$$\xi_i\,, \quad i = 1, \ldots, N$$

$\xi_i = 0$ Data point outside region of separation (including those on the border)

$0 \leq \xi_i \leq 1$ Data point in region of separation and on correct side of hyperplane (e.g., $\mathbf{x}_1$ and $\mathbf{x}_3$)

$\xi_i > 1$ Data point in region of separation but on wrong side of hyperplane (e.g., $\mathbf{x}_2$ and $\mathbf{x}_4$)

Back



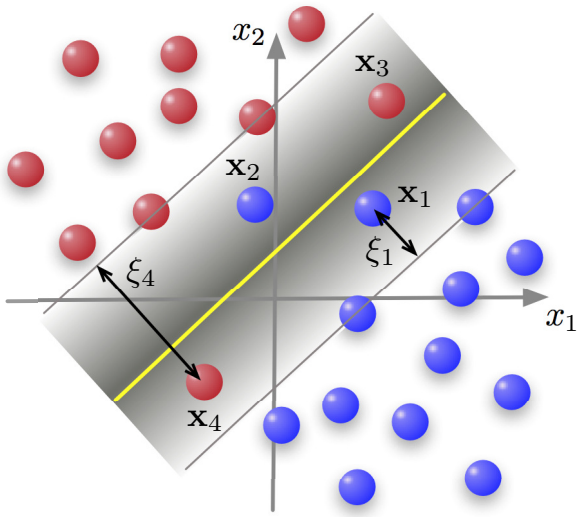
Optimal hyperplane must also minimize error penalty

$$\sum_{i=1}^{N} \xi_i$$

New function to be minimized

$$f(\mathbf{w}, \boldsymbol{\xi}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N} \xi_i$$

- Value of $C > 0$ reflects cost of violating constraints
  - A large C generally leads to smaller margin but also fewer misclassification of training data
  - A small C generally leads to larger margin but more misclassification of training data
- As a design parameter, value of C is set by user



For an example $\mathbf{x}_i$, we have

$$d_i \left(\mathbf{w}^T \mathbf{x}_i + b\right) \geq 1 - \xi_i$$

If $\xi_i = 0$, then constraint is the same as that in basic formulation (which is also known as hard margin classification problem.)

Optimal hyperplane with soft margin: Primal problem

$$\text{Minimize:} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}\xi_i$$

$$\text{Subject to:} \quad d_i \left(\mathbf{w}^T \mathbf{x}_i + b\right) - 1 + \xi_i \geq 0$$

$$\xi_i \geq 0$$

ξ_i is a variable to be optimized

Let α_i and β_i be the Lagrange multipliers. Then

$$
\begin{aligned}
& L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \\
= \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i\left(d_i\left(\mathbf{w}^T\mathbf{x}_i + b\right) - 1 + \xi_i\right) - \sum_{i=1}^{N}\beta_i\xi_i \\
= \quad & \frac{\mathbf{w}^T\mathbf{w}}{2} + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i d_i \mathbf{w}^T\mathbf{x}_i - b\sum_{i=1}^{N}\alpha_i d_i + \sum_{i=1}^{N}\alpha_i - \sum_{i=1}^{N}\alpha_i\xi_i - \sum_{i=1}^{N}\beta_i\xi_i
\end{aligned}
$$

The KKT conditions are

$$\frac{\partial L}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{N}\alpha_i\, d_i\, \mathbf{x}_i = \mathbf{0}$$

$$\frac{\partial L}{\partial b} = -\sum_{i=1}^{N}\alpha_i\, d_i = 0$$

$$\frac{\partial L}{\partial \xi_i} = C - \alpha_i - \beta_i = 0$$

$$d_i\left(\mathbf{w}^T\mathbf{x}_i + b\right) - 1 + \xi_i \geq 0$$

$$\alpha_i\left(d_i\left(\mathbf{w}^T\mathbf{x}_i + b\right) - 1 + \xi_i\right) = 0$$

$$\beta_i\xi_i = 0$$

$$\alpha_i \geq 0$$

$$\beta_i \geq 0$$

$$\xi_i \geq 0$$

Back

Since $\mathbf{w} = \sum_{i=1}^{N} \alpha_i \, d_i \, \mathbf{x}_i$. We have

$$\begin{aligned} \mathbf{w}^T \mathbf{w} &= \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \, d_i \, \alpha_j \, d_j \, \mathbf{x}_i^T \mathbf{x}_j \\ \sum_{i=1}^{N} \alpha_i \, d_i \, \mathbf{w}^T \mathbf{x}_i &= \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \, d_i \, \alpha_j \, d_j \, \mathbf{x}_i^T \mathbf{x}_j \end{aligned}$$

Moreover, from the KKT conditions we also have

$$C = \alpha_i + \beta_i$$

Hence,

$$\begin{aligned} & L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \\ = \; & \frac{\mathbf{w}^T \mathbf{w}}{2} + C \sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \alpha_i d_i \mathbf{w}^T \mathbf{x}_i - b \sum_{i=1}^{N} \alpha_i d_i + \sum_{i=1}^{N} \alpha_i - \sum_{i=1}^{N} \alpha_i \xi_i - \sum_{i=1}^{N} \beta_i \xi_i \\ = \; & -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i d_i \alpha_j \, d_j \mathbf{x}_i^T \mathbf{x}_j + \sum_{i=1}^{N} \underbrace{(\alpha_i + \beta_i)}_{C} \xi_i + \sum_{i=1}^{N} \alpha_i - \sum_{i=1}^{N} \alpha_i \xi_i - \sum_{i=1}^{N} \beta_i \xi_i \\ = \; & -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \, d_i \, \alpha_j \, d_j \, \mathbf{x}_i^T \mathbf{x}_j + \sum_{i=1}^{N} \alpha_i \equiv Q(\boldsymbol{\alpha}) \end{aligned}$$

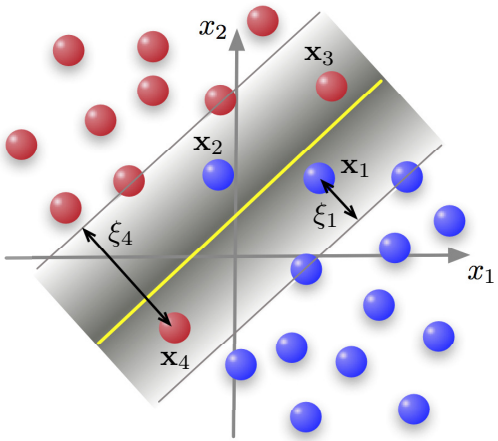
Dual problem (with soft margin)

$$\text{Find :} \quad \alpha_i$$

$$\text{Maximize :} \quad Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{Subject to :} \quad \sum_{i=1}^{N} \alpha_i d_i = 0 \quad \text{and} \quad 0 \leq \alpha_i \leq C$$

Note that $Q(\boldsymbol{\alpha})$ is same as that for the dual problem without soft margin. The effect of the error penalty $\sum_{i=1}^{N} \xi_i$ on the optimization problem is to set an upper bound for the Lagrange multiplier α_i. This can be seen by the fact that, from the KKT conditions, $\alpha_i = C - \beta_i$ with $\beta_i \geq 0$; that is, $0 \leq \alpha_i \leq C$.



From KKT conditions:

View

$$\begin{aligned} d_i\left(\mathbf{w}^T\mathbf{x}_i+b\right)-1+\xi_i &\geq 0 \\ \beta_i\xi_i &= 0 \\ \alpha_i+\beta_i &= C \end{aligned}$$

If the example $\mathbf{x}_i$ is a support vector (i.e., with $0<\alpha_i\leq C$), then

$$d_i\left(\mathbf{w}^T\mathbf{x}_i+b\right)-1+\xi_i=0 \quad\Rightarrow\quad d_i\left(\mathbf{w}^T\mathbf{x}_i+b\right)=1-\xi_i$$

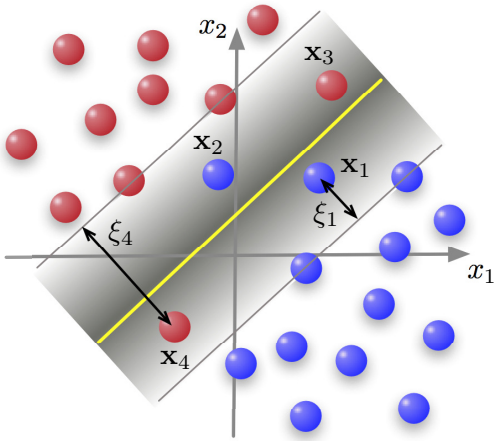
Case 1: Support vector $\mathbf{x}_i$ with $\alpha_i < C$

$$\beta_i = C-\alpha_i > 0$$

- To satisfy $\beta_i\xi_i=0$, we must have $\xi_i=0$
- Support vector is located **on the border** of margin of separation

Case 2: Support vector $\mathbf{x}_i$ with $\alpha_i = C$

(next slide)



From KKT conditions:

$$\begin{aligned} d_i\left(\mathbf{w}^T\mathbf{x}_i+b\right)-1+\xi_i &\geq 0 \\ \beta_i\xi_i &= 0 \\ \alpha_i+\beta_i &= C \end{aligned}$$

If the example $\mathbf{x}_i$ is a support vector (i.e., with $0<\alpha_i\leq C$), then

$$d_i\left(\mathbf{w}^T\mathbf{x}_i+b\right)-1+\xi_i=0 \quad \Rightarrow \quad d_i\left(\mathbf{w}^T\mathbf{x}_i+b\right)=1-\xi_i$$

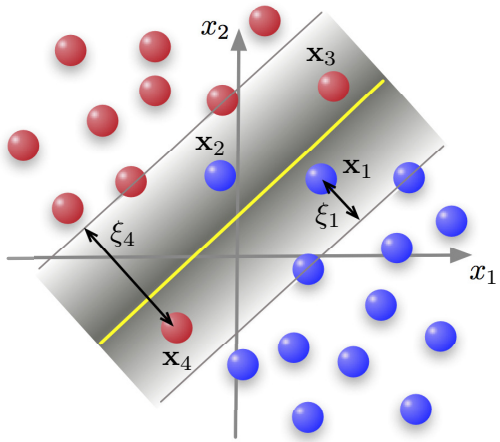
Case 2: Support vector with $\alpha_i = C$

$$\beta_i = C-\alpha_i = 0$$

- Condition $\beta_i\xi_i=0$ is still satisfied when $\xi_i\neq 0$
- Support vector is located **inside** margin of separation. ▸ View

If example $\mathbf{x}_i$ is a support vector (i.e., with $0 < \alpha_i \leq C$), then

$$d_i \left(\mathbf{w}^T \mathbf{x}_i + b\right) - 1 + \xi_i = 0 \quad \Rightarrow \quad d_i \left(\mathbf{w}^T \mathbf{x}_i + b\right) = 1 - \xi_i$$



Case 2: Support vector with $\alpha_i = C$

- $\xi_i \neq 0$
- SV $\mathbf{x}_i$ is **inside** margin of separation

SV $\mathbf{x}_i$ is classified correctly if

$$\text{sgn}\left[d_i \left(\mathbf{w}^T \mathbf{x}_i + b\right)\right] \equiv \text{sgn}\left[1 - \xi_i\right] = 1$$

- If $\xi_i \leq 1$, $\mathbf{x}_i$ is classified **correctly**; e.g., $\mathbf{x}_1$ and $\mathbf{x}_3$ in figure
- If $\xi_i > 1$, $\mathbf{x}_i$ is classified **incorrectly**; e.g., $\mathbf{x}_2$ and $\mathbf{x}_4$ in figure

	Primal	Dual
Find :	$\mathbf{w}\,, b$	α_i
Minimizing :	$f\left(\mathbf{w}, \boldsymbol{\xi}\right)$	$\|\mathbf{w}\|$
Maximizing :	$-$	$Q(\boldsymbol{\alpha})$
Subject to :	$d_i\left(\mathbf{w}^T\mathbf{x}_i + b\right) \geq 1 - \xi_i$	$\sum_{i=1}^{N} \alpha_i d_i = 0$
	$\xi_i \geq 0$	$0 \leq \alpha_i \leq C$

$$f\left(\mathbf{w}, \boldsymbol{\xi}\right) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}\xi_i \qquad Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\,\alpha_j\,d_i\,d_j\,\mathbf{x}_i^T\mathbf{x}_j$$

After $\alpha_{\circ,i}$ is obtained, we can calculate $\mathbf{w}_\circ$ as follows:

$$\mathbf{w}_\circ = \sum_{i=1}^{N}\alpha_{\circ,i}\,d_i\,\mathbf{x}_i$$

After $\mathbf{w}_\circ$ is obtained, we can calculate $b_\circ$ as follows:

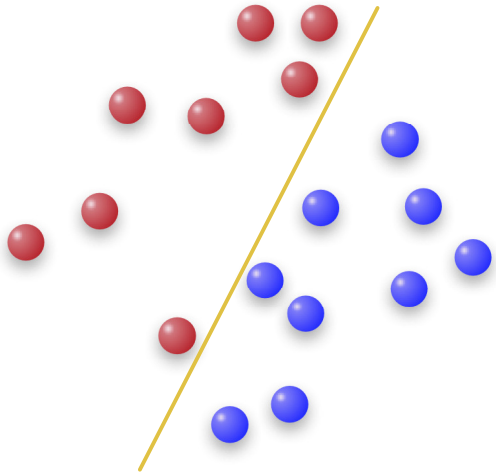
1. For each example $\mathbf{x}_i$ with $0 < \alpha_i \leq C$,

$$b_{\circ,i} = \frac{1}{d_i} - \mathbf{w}_\circ^T \mathbf{x}_i$$

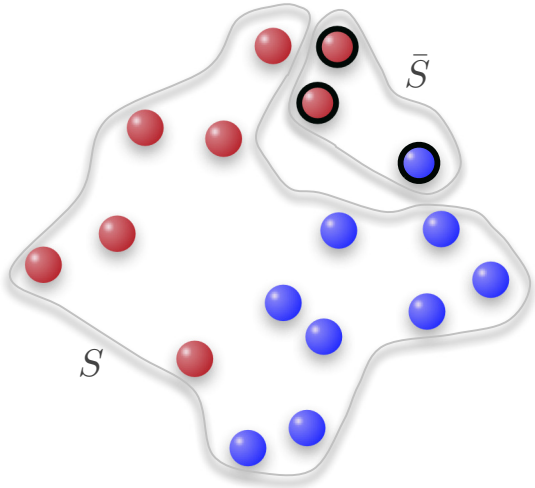
2. Take $b_\circ$ as the average of all such $b_{\circ,i}$

$$b_\circ = \frac{\sum_{i=1}^m b_{\circ,i}}{m}$$

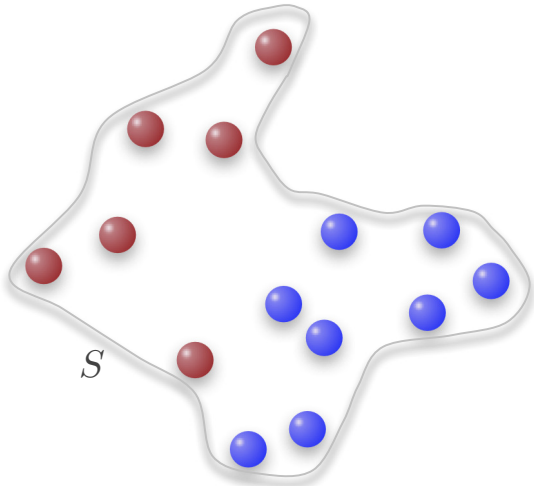
where m is the total number of $\mathbf{x}_i$ with $0 < \alpha_i \leq C$.



Data set Σ (assumed separable here for simple illustration)



Divide Σ into training set S and test set $\bar{S}$

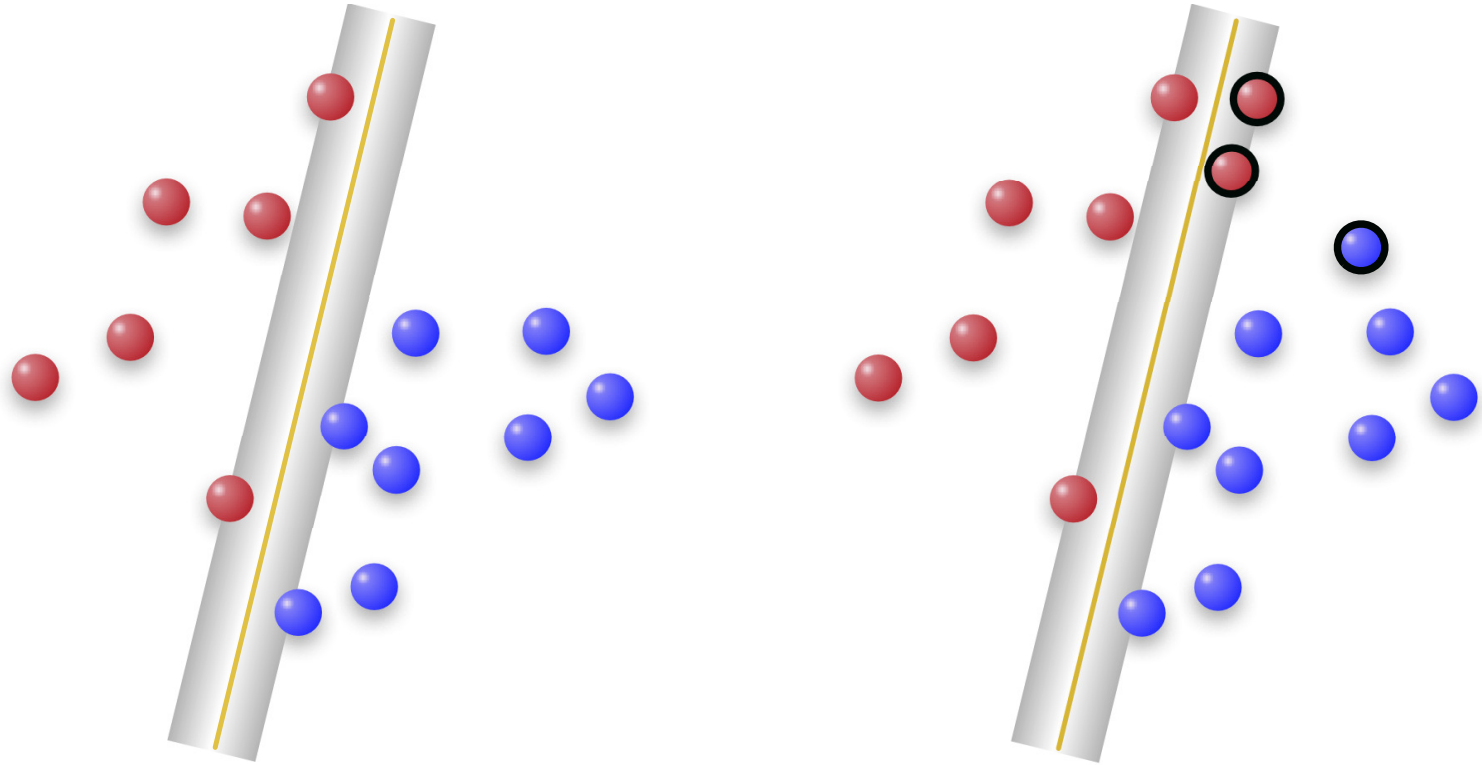


Use S to find $(\mathbf{w}_\circ, b_\circ)$ for SVM

Performance of SVM should be evaluated based on the number of misclassifications for Σ, not just S.

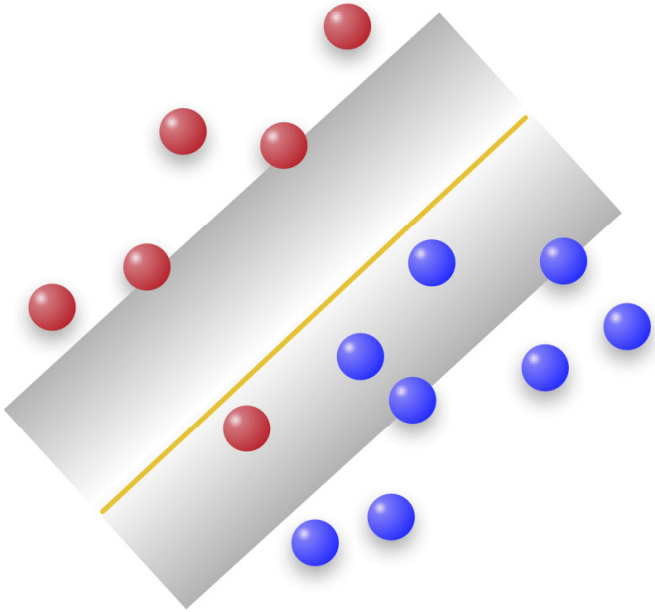
In other words, a SVM that classifies S perfectly may not be the desired solution to the classification problem.

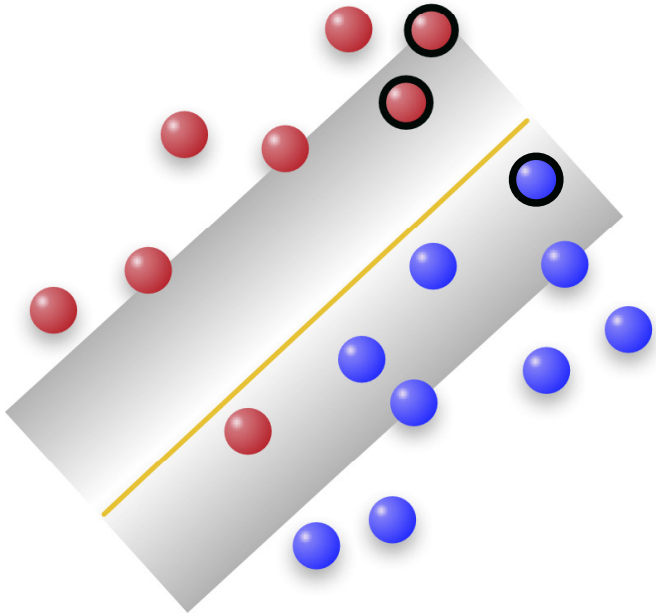
Why?



Optimal hyperplane with hard margin classifies all examples in training set S correctly, but misclassifies 2 examples in the test set $\bar{S}$.

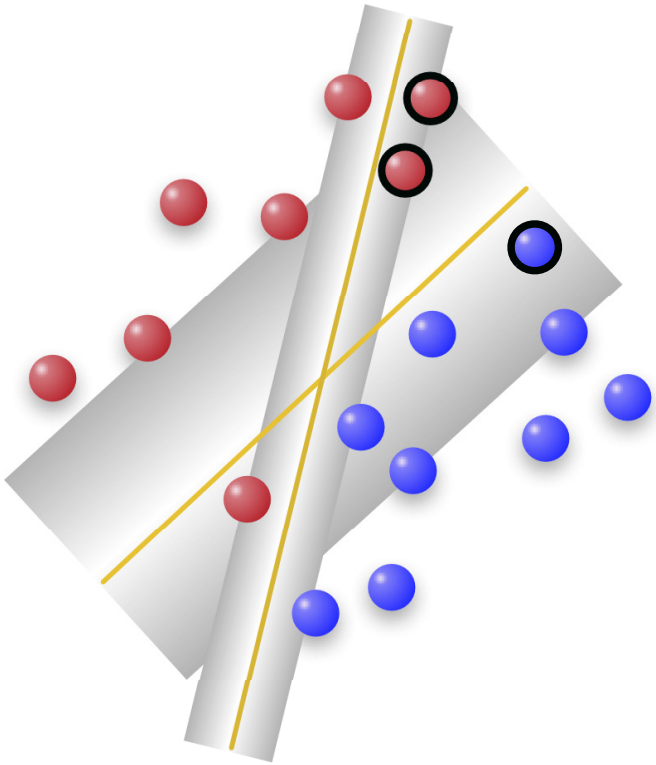
Hence, for the data set Σ, the number of misclassifications is 2.





An optimal hyperplane with soft margin misclassifies 1 example in training set S, but it classifies correctly all examples in test set $\bar{S}$.

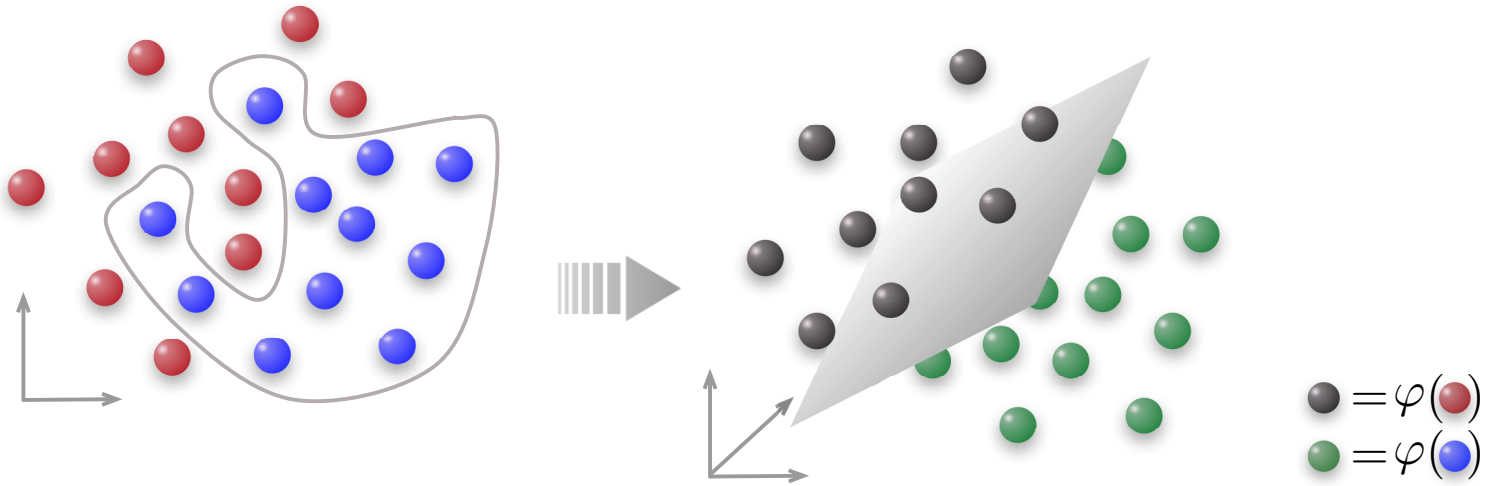
Hence, for the data set Σ, the number of misclassifications is 1.



Summary of misclassifications:

Type of SVM	S	$\bar{S}$	Σ
With hard margin	0	2	2
With soft margin	1	0	1

A SVM that classifies training set S perfectly may not be the desired solution to the classification problem.



Low-dimension space High-dimension space

Cover's Theorem

Probability that classes are linearly separable increases when data points in input space are nonlinearly mapped to a higher dimensional feature space.

Consider Newton's law of gravitation

$$f(m_1, m_2, r) = c\left(\frac{m_1 m_2}{r^2}\right) \qquad \text{(For simplicity, take } r = 1\text{)}$$

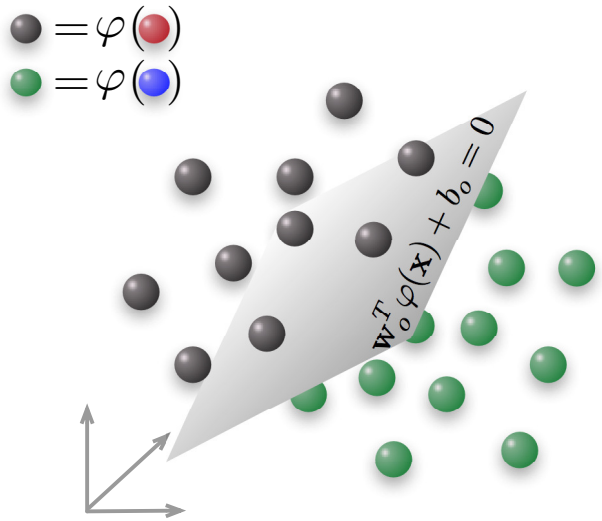
Map $f(m_1, m_2)$ into a new function $g(u, v)$ using nonlinear $\boldsymbol{\varphi}$

$$\mathbf{x} = \begin{bmatrix} m_1 \\ m_2 \end{bmatrix} \mapsto \boldsymbol{\varphi}(\mathbf{x}) = \begin{bmatrix} \varphi_1(\mathbf{x}) \\ \varphi_2(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} \ln m_1 \\ \ln m_2 \end{bmatrix} = \begin{bmatrix} u \\ v \end{bmatrix}$$

Solution: Applying the natural log on f yields

$$\begin{aligned} \ln f(m_1, m_2) &= \ln\left(c\left(\frac{m_1 m_2}{1^2}\right)\right) \\ &= \ln c + \ln m_1 + \ln m_2 - 2\ln 1 \\ &= u + v + c' \qquad \text{(where } c' = \ln c\text{)} \\ &= g(u, v) \end{aligned}$$

- $g(u, v)$ is linear with respect to the new coordinates (u, v).



Optimal hyperplane in feature space

$$g(\mathbf{x}) = \mathbf{w}_\circ^T \boldsymbol{\varphi}(\mathbf{x}) + b_\circ = 0$$

For training data $\mathbf{x}_i$

$$\begin{cases} g(\mathbf{x}_i) = \mathbf{w}_\circ^T \boldsymbol{\varphi}(\mathbf{x}_i) + b_\circ \geq +1 & \text{for } d_i = +1 \\ g(\mathbf{x}_i) = \mathbf{w}_\circ^T \boldsymbol{\varphi}(\mathbf{x}_i) + b_\circ \leq -1 & \text{for } d_i = -1 \end{cases}$$

or, in a compact form:

$$d_i \, g(\mathbf{x}_i) = d_i \left(\mathbf{w}_\circ^T \boldsymbol{\varphi}(\mathbf{x}_i) + b_\circ \right) \geq 1$$

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \alpha_i \left(d_i \left(\mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}_i) + b\right) - 1\right)$$

$$= \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \alpha_i\, d_i\, \mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}_i) - b\sum_{i=1}^{N} \alpha_i\, d_i + \sum_{i=1}^{N} \alpha_i$$

From KKT conditions: $\mathbf{w} = \sum_{i=1}^{N} \alpha_i d_i \boldsymbol{\varphi}(\mathbf{x}_i)$ and $\sum_{i=1}^{N} \alpha_i\, d_i = 0$

So $\mathbf{w}^T\mathbf{w} = \sum_{i=1}^{N} \alpha_i d_i \mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}_i) = \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j \boldsymbol{\varphi}^T(\mathbf{x}_i)\boldsymbol{\varphi}(\mathbf{x}_j)$

Let $Q(\boldsymbol{\alpha}) \equiv L(\mathbf{w}, b, \boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j\, d_i d_j \underbrace{\boldsymbol{\varphi}^T(\mathbf{x}_i)\boldsymbol{\varphi}(\mathbf{x}_j)}_{K(\mathbf{x}_i, \mathbf{x}_j)}$

Kernel: $\underbrace{K(\mathbf{x}_i, \mathbf{x}_j) = K(\mathbf{x}_j, \mathbf{x}_i)}_{\text{symmetric}} = \boldsymbol{\varphi}^T(\mathbf{x}_i)\boldsymbol{\varphi}(\mathbf{x}_j) = \boldsymbol{\varphi}^T(\mathbf{x}_j)\boldsymbol{\varphi}(\mathbf{x}_i)$

Given the mapping

$$\mathbf{x} = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T \mapsto \boldsymbol{\varphi}(\mathbf{x}) = \begin{bmatrix} 1 & x_1^2 & \sqrt{2}x_1x_2 & x_2^2 & \sqrt{2}x_1 & \sqrt{2}x_2 \end{bmatrix}^T$$

(i) Determine the kernel $K(\mathbf{x}, \mathbf{y})$

(ii) Calculate the value of the kernel if $\mathbf{x} = [1\ 2]^T$ and $\mathbf{y} = [3\ 4]^T$

Solution: (i) The kernel defined by this mapping is

$$\begin{aligned} K(\mathbf{x}, \mathbf{y}) &= \boldsymbol{\varphi}^T(\mathbf{x})\,\boldsymbol{\varphi}(\mathbf{y}) \\ &= \begin{bmatrix} 1 & x_1^2 & \sqrt{2}x_1x_2 & x_2^2 & \sqrt{2}x_1 & \sqrt{2}x_2 \end{bmatrix} \begin{bmatrix} 1 \\ y_1^2 \\ \sqrt{2}y_1y_2 \\ y_2^2 \\ \sqrt{2}y_1 \\ \sqrt{2}y_2 \end{bmatrix} \\ &= 1 + x_1^2y_1^2 + 2x_1x_2y_1y_2 + x_2^2y_2^2 + 2x_1y_1 + 2x_2y_2 \\ &= \left(1 + \mathbf{x}^T\mathbf{y}\right)^2 \end{aligned}$$

(ii) With $\mathbf{x} = [1\ 2]^T$ and $\mathbf{y} = [3\ 4]^T$, the value of the kernel is

$$\begin{aligned} K(\mathbf{x}, \mathbf{y}) &= 1 + x_1^2y_1^2 + 2x_1x_2y_1y_2 + x_2^2y_2^2 + 2x_1y_1 + 2x_2y_2 \\ &= 1 + 1^2 \cdot 3^2 + 2 \cdot 1 \cdot 2 \cdot 3 \cdot 4 + 2^2 \cdot 4^2 + 2 \cdot 1 \cdot 3 + 2 \cdot 2 \cdot 4 \\ &= 1 + 9 + 48 + 64 + 6 + 16 = 144 \end{aligned}$$

Dual problem with soft margin

$$\text{Find}: \quad \alpha_i$$

$$\text{Maximize}: \quad Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{Subject to}: \quad \sum_{i=1}^{N} \alpha_i d_i = 0\,, \; 0 \leq \alpha_i \leq C$$

Dual problem with soft margin and transformation

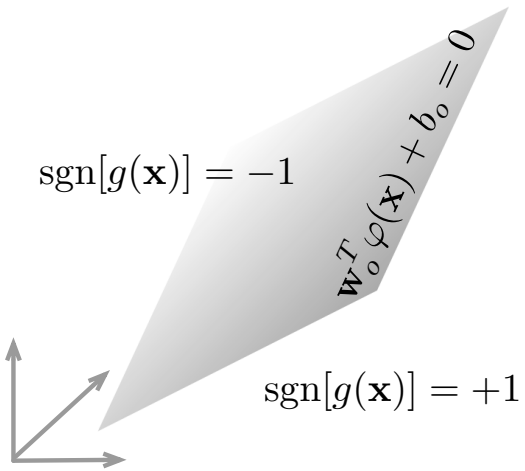
$$\text{Find}: \quad \alpha_i$$

$$\text{Maximize}: \quad Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j \boldsymbol{\varphi}^T(\mathbf{x}_i) \boldsymbol{\varphi}(\mathbf{x}_j)$$

$$\text{Subject to}: \quad \sum_{i=1}^{N} \alpha_i d_i = 0\,, \; 0 \leq \alpha_i \leq C$$

Given optimal value $\alpha_{\circ,i}$

$$\begin{cases} \mathbf{w}_\circ = \sum_{i=1}^{N} \alpha_{\circ,i}\, d_i \boldsymbol{\varphi}(\mathbf{x}_i) \\ b_\circ = \frac{1}{d^{(s)}} - \mathbf{w}_\circ^T \boldsymbol{\varphi}(\mathbf{x}^{(s)}) \\ (\mathbf{x}^{(s)} \text{ is a SV with label } d^{(s)}) \end{cases}$$



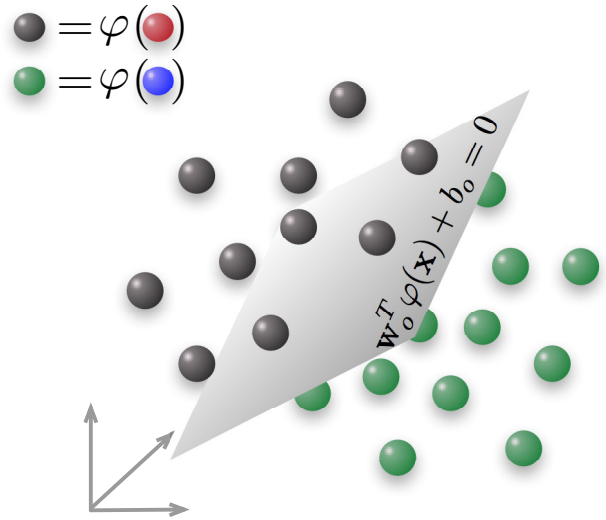
Discriminant function

$$g(\mathbf{x}) = \mathbf{w}_\circ^T \boldsymbol{\varphi}(\mathbf{x}) + b_\circ = \sum_{i=1}^{N} \alpha_{\circ,i} d_i \underbrace{\boldsymbol{\varphi}^T(\mathbf{x}_i)\boldsymbol{\varphi}(\mathbf{x})}_{K(\mathbf{x}_i,\mathbf{x})} + b_\circ$$

To classify a new data point $\mathbf{x}_{\text{new}}$

$$d_{\text{new}} = \text{sgn}\left[g(\mathbf{x}_{\text{new}})\right]$$

Main issue: Need $\boldsymbol{\varphi}(\cdot)$



Solution requires φ but finding explicit form of φ is difficult

Kernel trick:

Find expression for $K(\cdot,\cdot)$ directly

For instance, find

$$K(\mathbf{x},\mathbf{y}) = \left(1+\mathbf{x}^T\mathbf{y}\right)^2 = \boldsymbol{\varphi}^T(\mathbf{x})\,\boldsymbol{\varphi}(\mathbf{y})$$

without first knowing the mapping

$$\mathbf{x} = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T \mapsto \boldsymbol{\varphi}(\mathbf{x}) = \begin{bmatrix} 1 & x_1^2 & \sqrt{2}x_1x_2 & x_2^2 & \sqrt{2}x_1 & \sqrt{2}x_2 \end{bmatrix}^T$$

Kernel trick Find an expression for $K(\cdot,\cdot)$ directly without knowing $\varphi(\cdot)$

Procedure Choose an expression for $K(\cdot,\cdot)$. If this expression satisfies the Mercer's Condition, then it can be used as a kernel

Mercer's condition

For training set $S = \{(\mathbf{x}_i, d_i)\}$, $i = 1, 2, \ldots, N$, the Gram matrix

$$\mathbf{K} = \begin{bmatrix} K(\mathbf{x}_1, \mathbf{x}_1) & \ldots & K(\mathbf{x}_1, \mathbf{x}_N) \\ \vdots & \ddots & \vdots \\ K(\mathbf{x}_N, \mathbf{x}_1) & \ldots & K(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix} \in R^{N \times N}$$

is positive semi-definite (i.e., its eigenvalues are nonnegative) ▸ View

Given kernel $K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i^T \mathbf{x}_j)^2$ and training set below. Determine $\mathbf{K}$.

i	1	2	3	4
$\mathbf{x}_i$	$[-1,-1]^T$	$[-1,+1]^T$	$[+1,-1]^T$	$[+1,+1]^T$

$$
\begin{aligned}
K(\mathbf{x}_1, \mathbf{x}_1) &= \left(1 + \mathbf{x}_1^T \mathbf{x}_1\right)^2 = \left(1 + \begin{bmatrix} -1 & -1 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \end{bmatrix}\right)^2 = 9 \\
K(\mathbf{x}_1, \mathbf{x}_2) &= \left(1 + \mathbf{x}_1^T \mathbf{x}_2\right)^2 = \left(1 + \begin{bmatrix} -1 & -1 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \end{bmatrix}\right)^2 = 1 = K(\mathbf{x}_2, \mathbf{x}_1) \\
K(\mathbf{x}_1, \mathbf{x}_3) &= \left(1 + \mathbf{x}_1^T \mathbf{x}_3\right)^2 = \left(1 + \begin{bmatrix} -1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix}\right)^2 = 1 = K(\mathbf{x}_3, \mathbf{x}_1) \\
K(\mathbf{x}_1, \mathbf{x}_4) &= \left(1 + \mathbf{x}_1^T \mathbf{x}_4\right)^2 = \left(1 + \begin{bmatrix} -1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right)^2 = 1 = K(\mathbf{x}_4, \mathbf{x}_1) \\
\vdots & \quad \vdots
\end{aligned}
$$

Therefore, the Gram matrix is

$$
\mathbf{K} = \begin{bmatrix} K(\mathbf{x}_1, \mathbf{x}_1) & \dots & K(\mathbf{x}_1, \mathbf{x}_4) \\ \vdots & \ddots & \vdots \\ K(\mathbf{x}_4, \mathbf{x}_1) & \dots & K(\mathbf{x}_4, \mathbf{x}_4) \end{bmatrix} = \begin{bmatrix} 9 & 1 & 1 & 1 \\ 1 & 9 & 1 & 1 \\ 1 & 1 & 9 & 1 \\ 1 & 1 & 1 & 9 \end{bmatrix}
$$

Problem: Consider the set of data in the table below and the kernel candidate $K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\mathbf{x}_i^T \mathbf{x}_j - 1)$.

i	1	2	3	4
$\mathbf{x}_i$	$[-1,-1]^T$	$[-1,+1]^T$	$[+1,-1]^T$	$[+1,+1]^T$

(i) Set up the equation that can be solved to obtain the eigenvalues of the Gram matrix associated with this kernel candidate.

(ii) Determine whether the kernel candidate satisfies Mercer's condition. (Note: You might need to use some software, such as MATLAB, to find the eigenvalues.)

Solution: (i) We first set up the Gram matrix.

$$K(\mathbf{x}_1, \mathbf{x}_1) = \tanh\left(\mathbf{x}_1^T \mathbf{x}_1 - 1\right) = \tanh\left(\begin{bmatrix} -1 & -1 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \end{bmatrix} - 1\right) = 0.7616$$

$$K(\mathbf{x}_1, \mathbf{x}_2) = \tanh\left(\mathbf{x}_1^T \mathbf{x}_2 - 1\right) = \tanh\left(\begin{bmatrix} -1 & -1 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \end{bmatrix} - 1\right) = -0.7616$$

$$K(\mathbf{x}_1, \mathbf{x}_3) = \tanh\left(\mathbf{x}_1^T \mathbf{x}_3 - 1\right) = \tanh\left(\begin{bmatrix} -1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} - 1\right) = -0.7616$$

$$K(\mathbf{x}_1, \mathbf{x}_4) = \tanh\left(\mathbf{x}_1^T \mathbf{x}_4 - 1\right) = \tanh\left(\begin{bmatrix} -1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} - 1\right) = -0.9951$$

$$\vdots \qquad \vdots \qquad \vdots \qquad \vdots$$

The Gram matrix is

$$\mathbf{K} = \begin{bmatrix} K(\mathbf{x}_1, \mathbf{x}_1) & \dots & K(\mathbf{x}_1, \mathbf{x}_4) \\ \vdots & \ddots & \vdots \\ K(\mathbf{x}_4, \mathbf{x}_1) & \dots & K(\mathbf{x}_4, \mathbf{x}_4) \end{bmatrix}$$

$$= \begin{bmatrix} 0.7616 & -0.7616 & -0.7616 & -0.9951 \\ -0.7616 & 0.7616 & -0.9951 & -0.7616 \\ -0.7616 & -0.9951 & 0.7616 & -0.7616 \\ -0.9951 & -0.7616 & -0.7616 & 0.7616 \end{bmatrix}$$

The eigenvalues of $\mathbf{K}$ are determined by solving the equation

$$
\begin{aligned}
& |\lambda \mathbf{I} - \mathbf{K}| \\
= & \left| \lambda \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} - \begin{bmatrix} 0.7616 & -0.7616 & -0.7616 & -0.9951 \\ -0.7616 & 0.7616 & -0.9951 & -0.7616 \\ -0.7616 & -0.9951 & 0.7616 & -0.7616 \\ -0.9951 & -0.7616 & -0.7616 & 0.7616 \end{bmatrix} \right| \\
= & \begin{vmatrix} \lambda - 0.7616 & 0.7616 & 0.7616 & 0.9951 \\ 0.7616 & \lambda - 0.7616 & 0.9951 & 0.7616 \\ 0.7616 & 0.9951 & \lambda - 0.7616 & 0.7616 \\ 0.9951 & 0.7616 & 0.7616 & \lambda - 0.7616 \end{vmatrix} = 0
\end{aligned}
$$

(ii) The eigenvalues of the Gram matrix $\mathbf{K}$ are found to be:

$$
\begin{aligned}
\lambda_1 &= -1.7567 < 0 \\
\lambda_2 &= 1.7567 \\
\lambda_3 &= 1.7567 \\
\lambda_4 &= 1.2897
\end{aligned}
$$

Since one of the eigenvalues is negative, this kernel candidate is not admissible.

Given a training set $S = \{(\mathbf{x}_i, d_i)\}$, $i = 1, \ldots, N$

1. Find a suitable kernel

   Choose expression then check Mercer's condition

2. Choose a value for C
3. Solve for $\alpha_{\circ,i}$
4. Determine $b_\circ$ in

$$g(\mathbf{x}) = \sum_{i=1}^{N} \alpha_{\circ,i} d_i K(\mathbf{x}, \mathbf{x}_i) + b_\circ$$

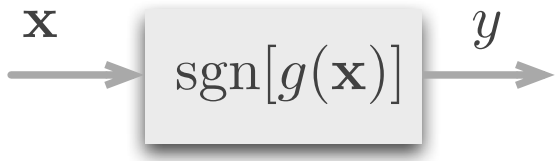
using the fact that for a support vector $\mathbf{x}^{(s)}$

$$g(\mathbf{x}^{(s)}) = \pm 1 = d^{(s)}$$

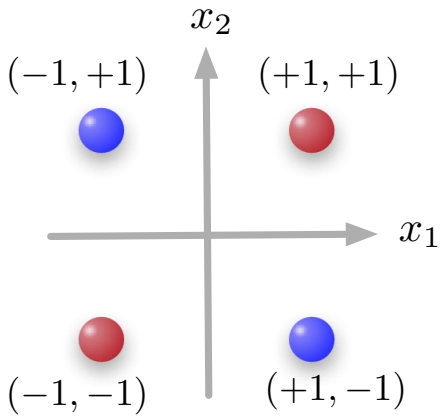
$$\text{Maximize :} \quad Q(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j K(\mathbf{x}_i, \mathbf{x}_j)$$

$$\text{Subject to :} \quad \sum_{i=1}^{N} \alpha_i d_i = 0\,, \; 0 \le \alpha_i \le C$$

Support vector machine:



Example: Exclusive OR



i	$\mathbf{x}_i$	d_i
1	$[-1,-1]^T$	-1
2	$[-1,+1]^T$	$+1$
3	$[+1,-1]^T$	$+1$
4	$[+1,+1]^T$	-1

Choose kernel: $K(\mathbf{x}_i,\mathbf{x}_j) = \left(1+\mathbf{x}_i^T\mathbf{x}_j\right)^2$

$$= 1 + x_{i,1}^2 x_{j,1}^2 + 2x_{i,1}x_{i,2}x_{j,1}x_{j,2}$$
$$+ x_{i,2}^2 x_{j,2}^2 + 2x_{i,1}x_{j,1} + 2x_{i,2}x_{j,2}$$

The Gram matrix is

$$\mathbf{K} = \begin{bmatrix} 9 & 1 & 1 & 1 \\ 1 & 9 & 1 & 1 \\ 1 & 1 & 9 & 1 \\ 1 & 1 & 1 & 9 \end{bmatrix}$$

The eigenvalues are

In *Mathematica*

```
In[1] = Eigenvalues[
{{9, 1, 1, 1}, {1, 9, 1, 1},
 {1, 1, 9, 1}, {1, 1, 1, 9}}]
Out[1]  = {12, 8, 8, 8}
```

$$
\begin{aligned}
Q(\boldsymbol{\alpha}) &= \sum_{i=1}^{4} \alpha_i - \frac{1}{2} \sum_{i=1}^{4} \sum_{j=1}^{4} \alpha_i \alpha_j d_i d_j K(\mathbf{x}_i, \mathbf{x}_j) \\
&= \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 - \\
&\quad \frac{1}{2} \Big(9\alpha_1^2 - 2\alpha_1\alpha_2 - \alpha_1\alpha_3 + 2\alpha_1\alpha_4 + \\
&\quad 9\alpha_2^2 + 2\alpha_2\alpha_3 - 2\alpha_2\alpha_4 + 9\alpha_3^2 - 2\alpha_3\alpha_4 + 9\alpha_4^2 \Big)
\end{aligned}
$$

$$
\left.
\begin{aligned}
9\alpha_1 - \alpha_2 - \alpha_3 + \alpha_4 &= 1 \\
-\alpha_1 + 9\alpha_2 + \alpha_3 - \alpha_4 &= 1 \\
-\alpha_1 + \alpha_2 + 9\alpha_3 - \alpha_4 &= 1 \\
\alpha_1 - \alpha_2 - \alpha_3 + 9\alpha_4 &= 1
\end{aligned}
\right\} \quad \frac{\partial Q}{\partial \alpha_i} = 0
$$

$$
\left.
\begin{aligned}
-\alpha_1 + \alpha_2 + \alpha_3 - \alpha_4 &= 0
\end{aligned}
\right\} \quad \text{from KKT conditions}
$$

$$
\alpha_{\mathrm{o},1} = \alpha_{\mathrm{o},2} = \alpha_{\mathrm{o},3} = \alpha_{\mathrm{o},4} = \frac{1}{8} \qquad (\text{all } \mathbf{x}_i \text{ are SVs})
$$

Note: This approach happens to work in this particular case. It is not a general method. More sophisticated numerical methods are needed to solve general constrained optimization problems

Discriminant function

General vector: $\mathbf{x} = [x_1, x_2]^T$

$$\begin{aligned} g(\mathbf{x}) &= \sum_{i=1}^{N} \alpha_{\circ,i} d_i K(\mathbf{x}, \mathbf{x}_i) + b_\circ \\ &= \alpha_{\circ,1} d_1 K(\mathbf{x}, \mathbf{x}_1) + \\ &\quad \alpha_{\circ,2} d_2 K(\mathbf{x}, \mathbf{x}_2) + \\ &\quad \alpha_{\circ,3} d_3 K(\mathbf{x}, \mathbf{x}_3) + \\ &\quad \alpha_{\circ,4} d_4 K(\mathbf{x}, \mathbf{x}_4) + b_\circ \end{aligned}$$

Training data: $\mathbf{x}_i = [x_{i,1}, x_{i,2}]^T$

i	$\mathbf{x}_i$	d_i
1	$[-1,-1]^T$	-1
2	$[-1,+1]^T$	$+1$
3	$[+1,-1]^T$	$+1$
4	$[+1,+1]^T$	-1

$$K(\mathbf{x}_i, \mathbf{x}_j) = 1 + x_{i,1}^2 x_{j,1}^2 + 2x_{i,1}x_{i,2}x_{j,1}x_{j,2} + x_{i,2}^2 x_{j,2}^2 + 2x_{i,1}x_{j,1} + 2x_{i,2}x_{j,2}$$

For example, $K(\mathbf{x}, \mathbf{x}_1)$ is calculated as follows:

$$\begin{aligned} & 1 + x_1^2 x_{1,1}^2 + 2x_1 x_2 x_{1,1} x_{1,2} + x_2^2 x_{1,2}^2 + 2x_1 x_{1,1} + 2x_2 x_{1,2} \\ = \; & 1 + x_1^2 (-1)^2 + 2x_1 x_2 (-1)(-1) + x_2^2 (-1)^2 + 2x_1 (-1) + 2x_2 (-1) \\ = \; & 1 + x_1^2 + 2x_1 x_2 + x_2^2 - 2x_1 - 2x_2 \end{aligned}$$

$$
\begin{aligned}
g(\mathbf{x}) &= \sum_{i=1}^{N} \alpha_{\circ,i} d_i K(\mathbf{x}, \mathbf{x}_i) + b_\circ \\
&= \alpha_{\circ,1} d_1 K(\mathbf{x}, \mathbf{x}_1) + \\
&\quad \alpha_{\circ,2} d_2 K(\mathbf{x}, \mathbf{x}_2) + \\
&\quad \alpha_{\circ,3} d_3 K(\mathbf{x}, \mathbf{x}_3) + \\
&\quad \alpha_{\circ,4} d_4 K(\mathbf{x}, \mathbf{x}_4) + b_\circ \\
&= \frac{1}{8}(-8 x_1 x_2) + b_\circ \\
&= -x_1 x_2 + b_\circ
\end{aligned}
$$

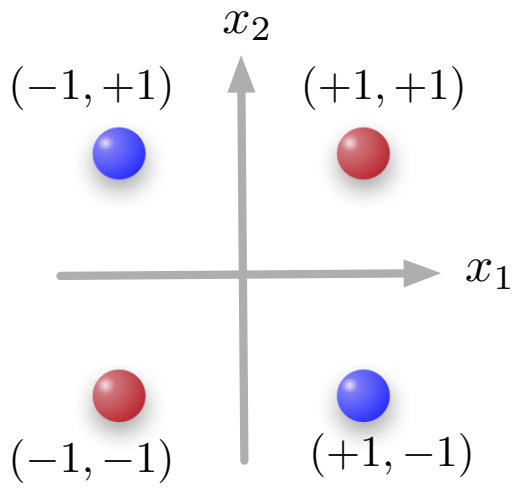
General vector: $\mathbf{x} = [x_1, x_2]^T$

Training data: $\mathbf{x}_i = [x_{i,1}, x_{i,2}]^T$

i	$\mathbf{x}_i$	d_i	$\alpha_{\circ,i}$
1	$[-1,-1]^T$	-1	$1/8$
2	$[-1,+1]^T$	$+1$	$1/8$
3	$[+1,-1]^T$	$+1$	$1/8$
4	$[+1,+1]^T$	-1	$1/8$

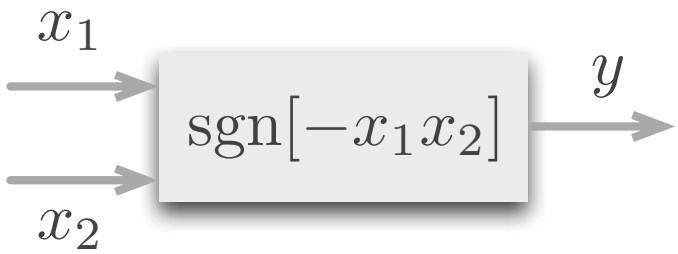
To find $b_\circ$, pick a support vector, say, $\mathbf{x}_2 = [x_{2,1}\ x_{2,2}]^T$, then

$$
\begin{aligned}
g(\mathbf{x}_2) &= -x_{2,1} x_{2,2} + b_0 = 1 \\
b_\circ &= 1 + x_{2,1} x_{2,2} = 1 + (-1)1 = 0
\end{aligned}
$$

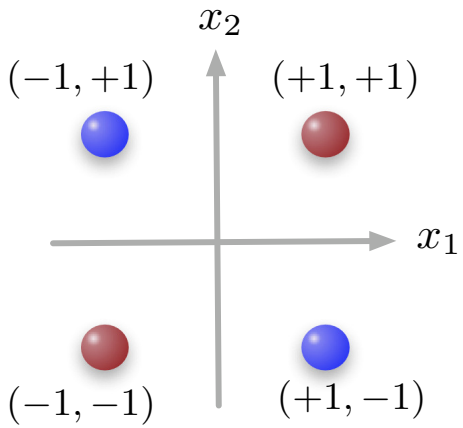


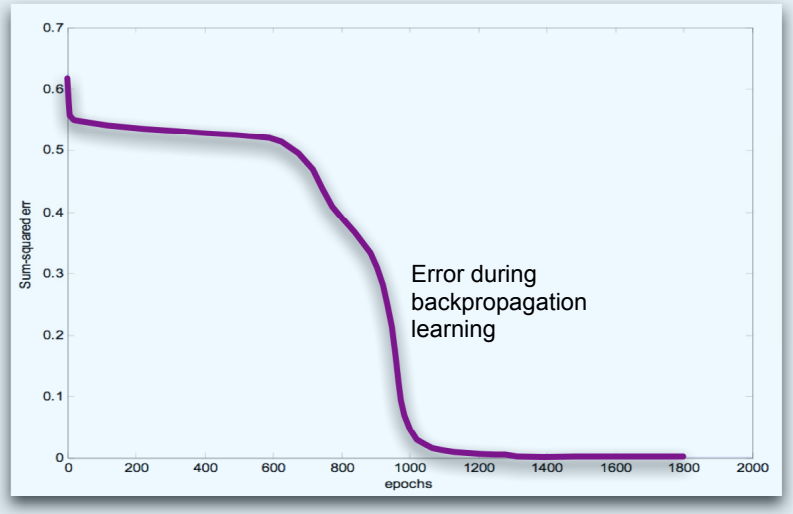
i	$\mathbf{x}_i$	d_i
1	$[-1,-1]^T$	-1
2	$[-1,+1]^T$	$+1$
3	$[+1,-1]^T$	$+1$
4	$[+1,+1]^T$	-1

Solution in the form of a SVM:

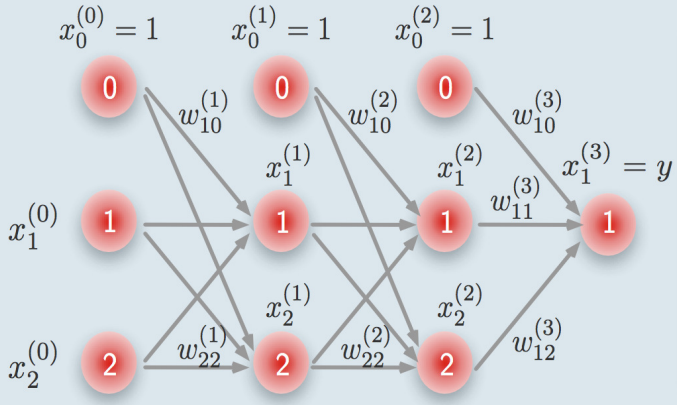


Advantage of SVM





MLP solution

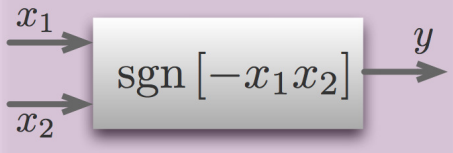


$$\boldsymbol{w}^{(1)} = \begin{matrix}\text{bias} & & \\ \end{matrix}\begin{bmatrix} -1.87 & 5.17 & 5.10 \\ -4.66 & 3.06 & 3.04 \end{bmatrix}$$

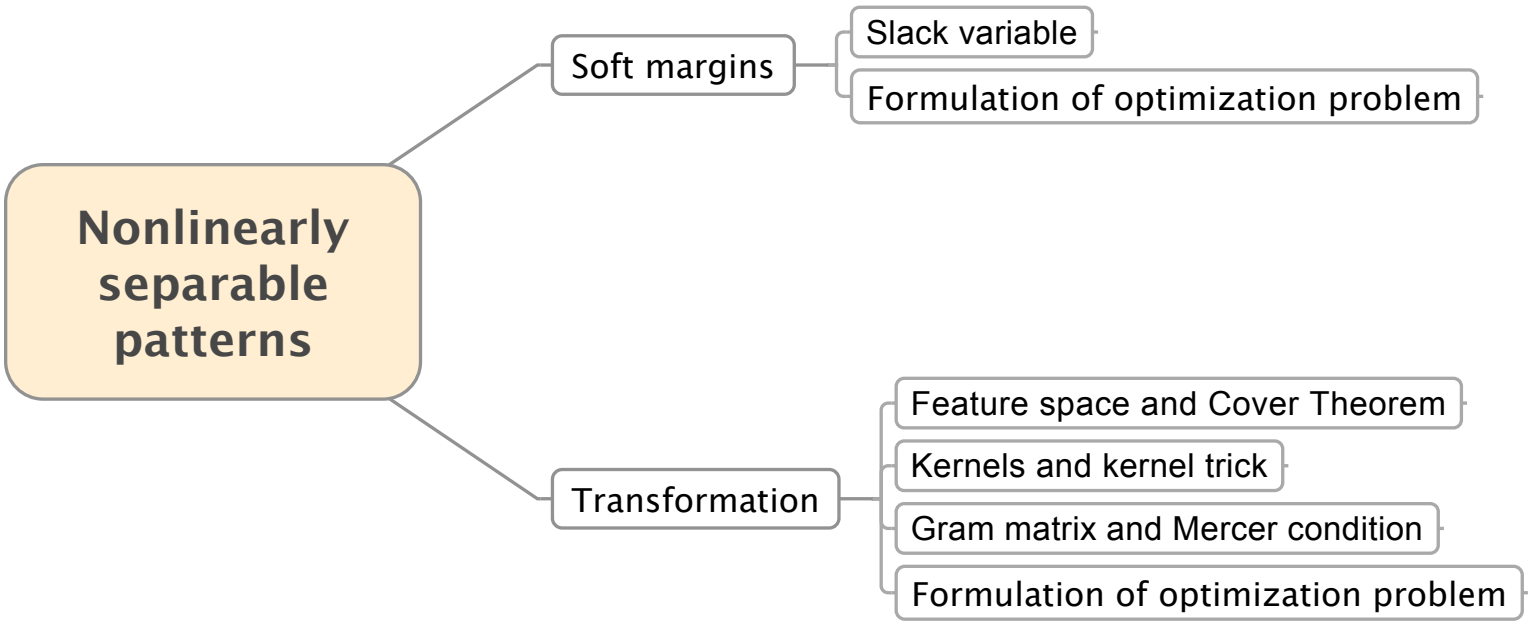
$$\boldsymbol{w}^{(2)} = \begin{matrix}\text{bias} & & \\ \end{matrix}\begin{bmatrix} -1.06 & 3.69 & -5.29 \\ 2.53 & -4.67 & 3.54 \end{bmatrix}$$

$$\boldsymbol{w}^{(3)} = \begin{matrix}\text{bias} \\ \end{matrix}[-0.36,\ 6.49,\ -6.5147]$$

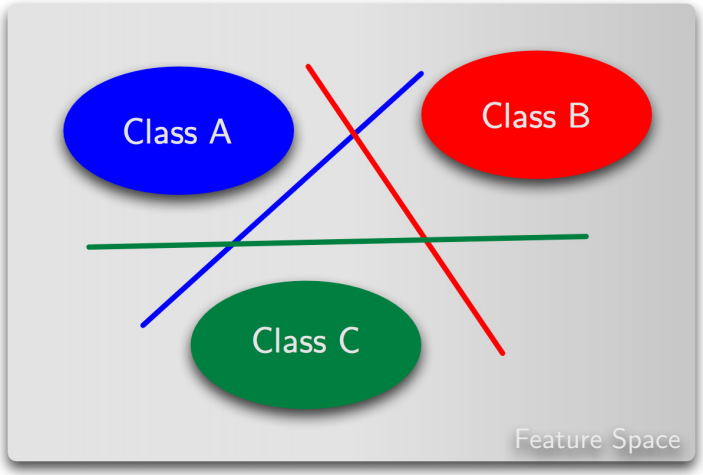
SVM solution



SVM for nonlinearly separable patterns: Summary

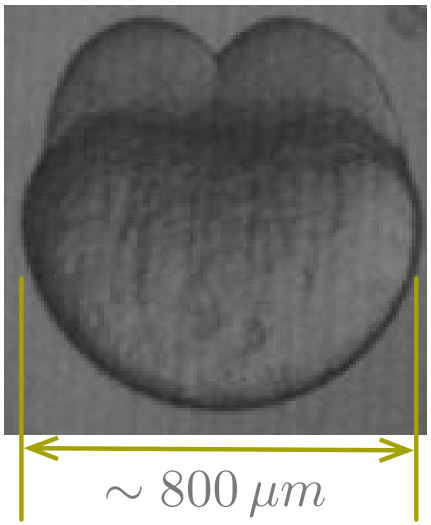


SVM for multi-class classification: One-versus-the-rest

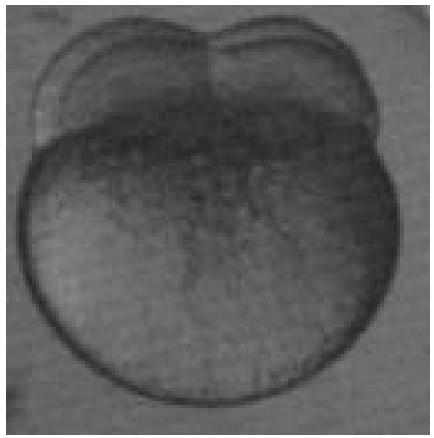


- K data classes: C_1, C_2, ..., C_K
- Construct K SVMs, with the k^{th} SVM trained with data from C_k as positive examples and data from remaining $K-1$ classes as negative examples

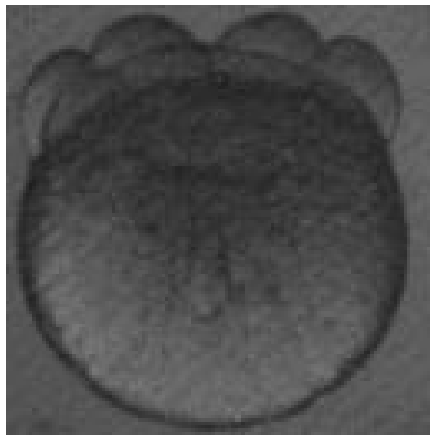
Two-cell embryo



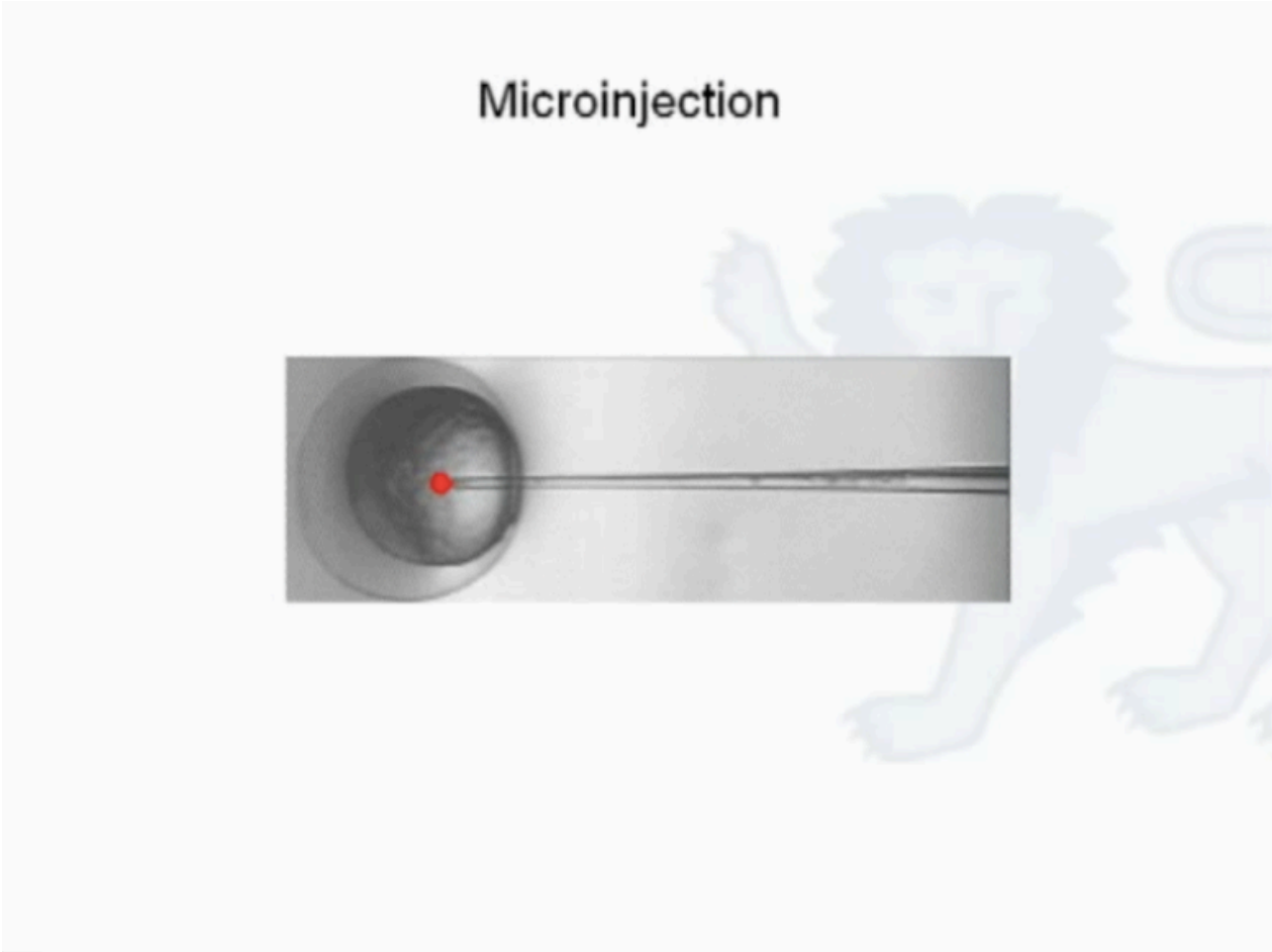
Four-cell embryo



Eight-cell embryo

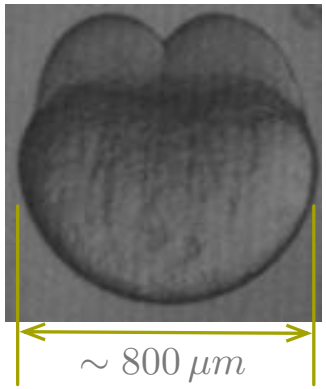


A Micromanipulation System for Automatic Batch Microinjection

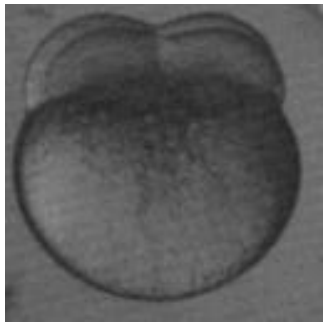


Zhe LU, Peter CHEN, Joo Hoo NAM, Ruowen GE, and Wei LIN
2007 IEEE International Conference on Robotics and Automation, 10-14 April 2007, Roma, Italy

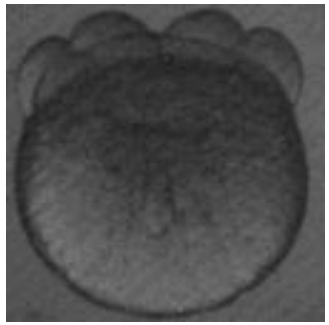
Two-cell embryo



Four-cell embryo



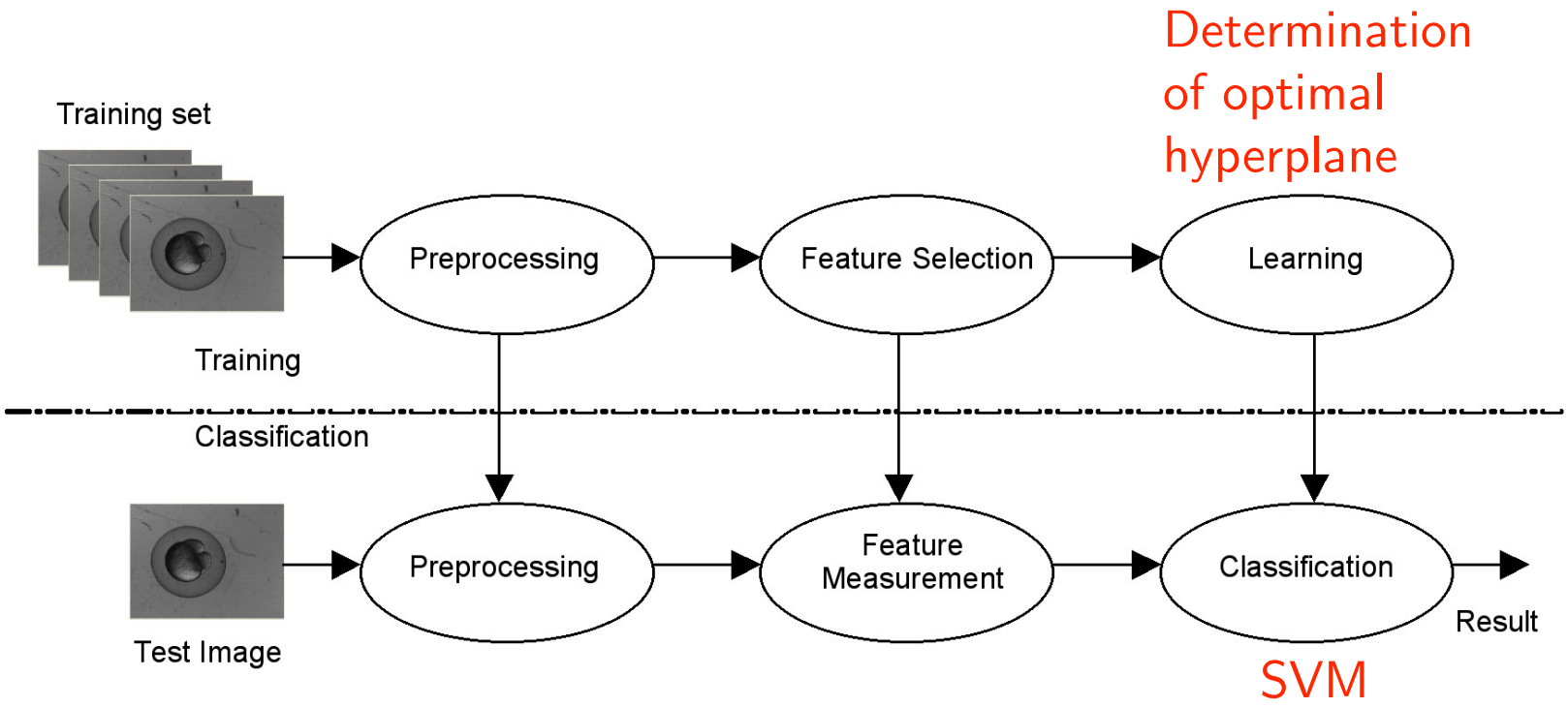
Eight-cell embryo

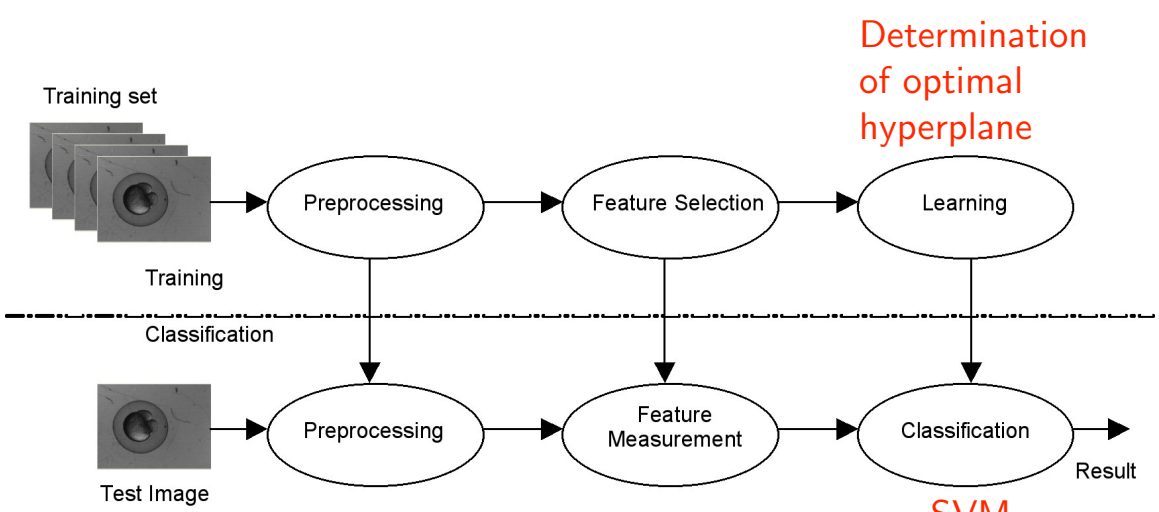


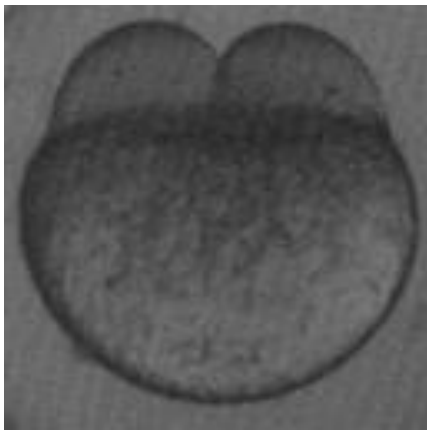
- Injecting DNA material into embryos to study development
- Injection done at the right development stage of embryo
  (Too late for injection if embryo has > 8 cells)
- Depending on stage, DNA used may be different

✓ Important to classify stage of embryos before injection

SVM for classification of zebrafish embryos: Learning and classification







128×128 pixels
with 8-bit gray level

$$\begin{bmatrix} 0.06294 & 0.3865 & -0.4545 & -0.3429 \\ 0.03054 & -0.3097 & -0.3181 & 0.4041 \\ -0.2543 & 0.3271 & -0.5241 & -0.8429 \\ 0.2073 & -0.2562 & -0.07045 & -0.3694 \\ 0.7951 & -0.1101 & 0.2174 & 0.6135 \\ 0.0488 \end{bmatrix}$$

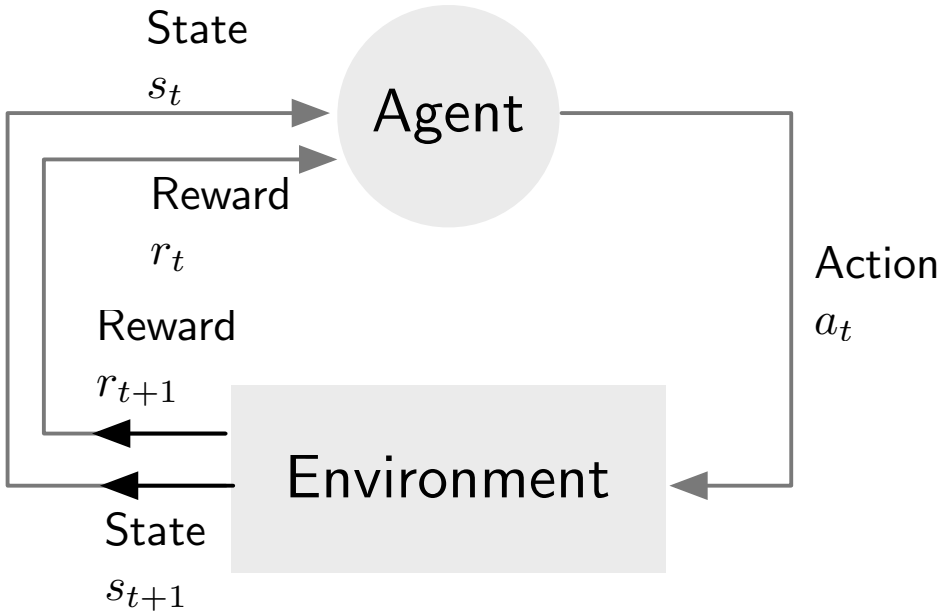
Vector of 21 normalized eigenbases

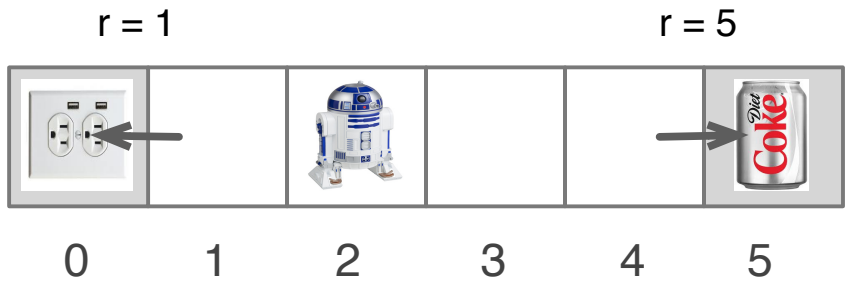
- SVM with quadratic kernel and $C = 1.25$
- A totla of 61 images:

2-cell embryo	25
4-cell embryo	21
8-cell embryo	15
Training set	40
Validation set	21

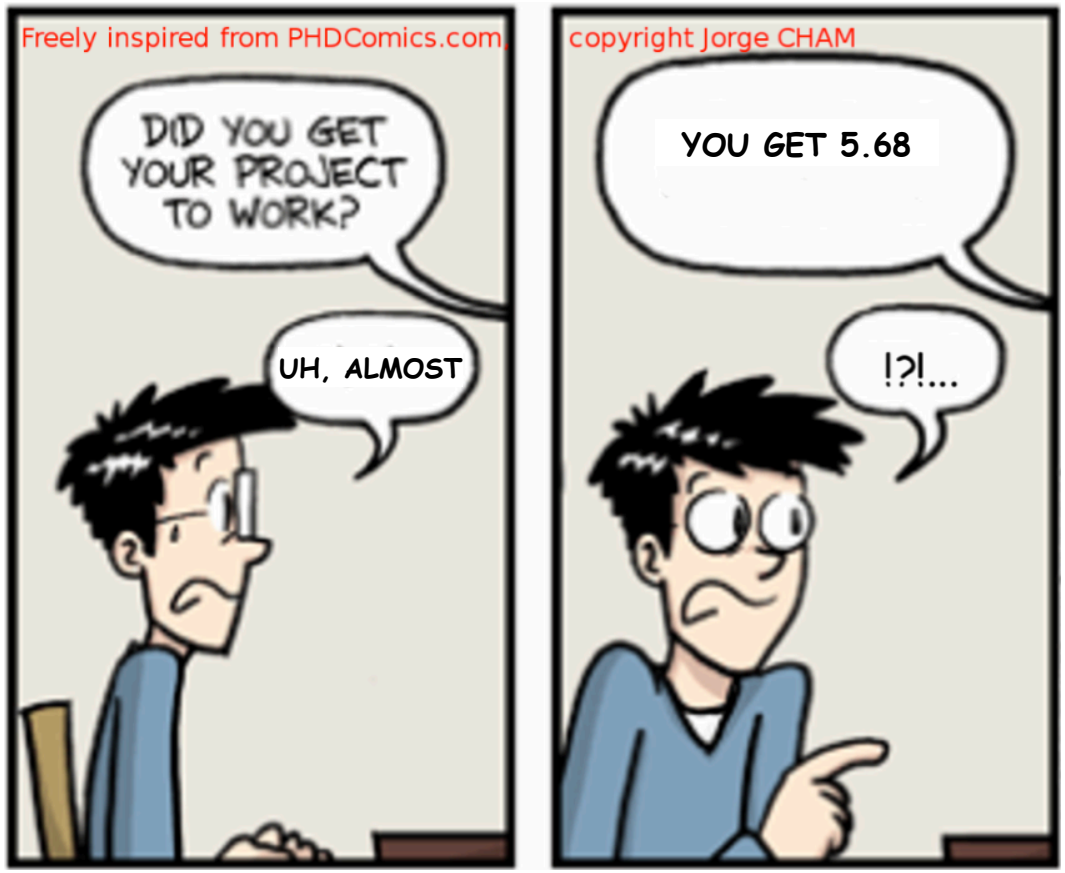
- Correct classification: 71%

Agent learns to maximize reward when completing task



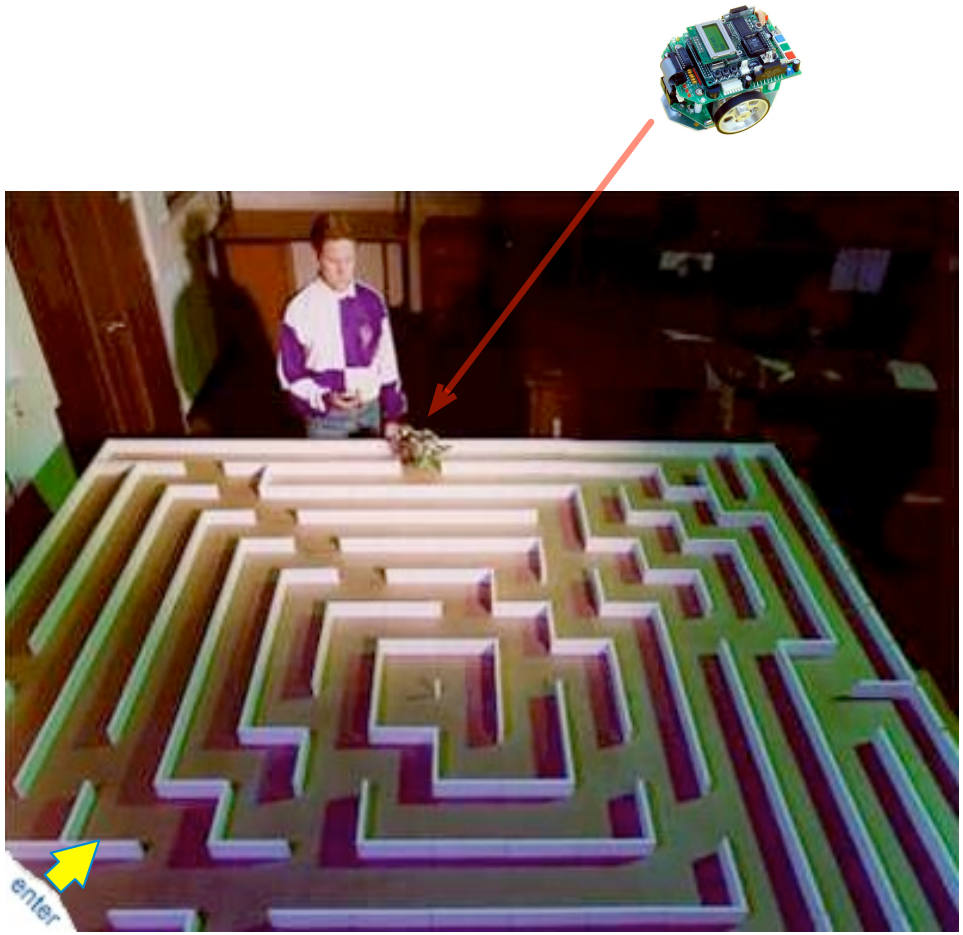


Reinforcement learning is unsupervised learning



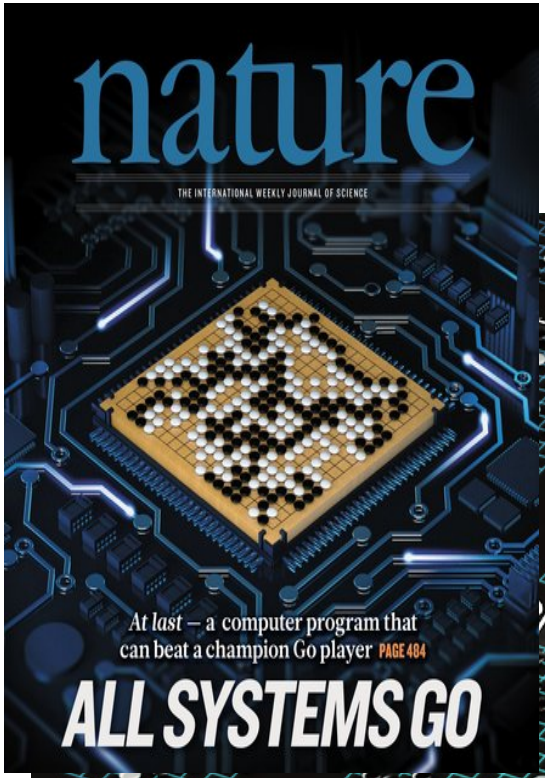
Adapted from slides by Olivier Sigaud

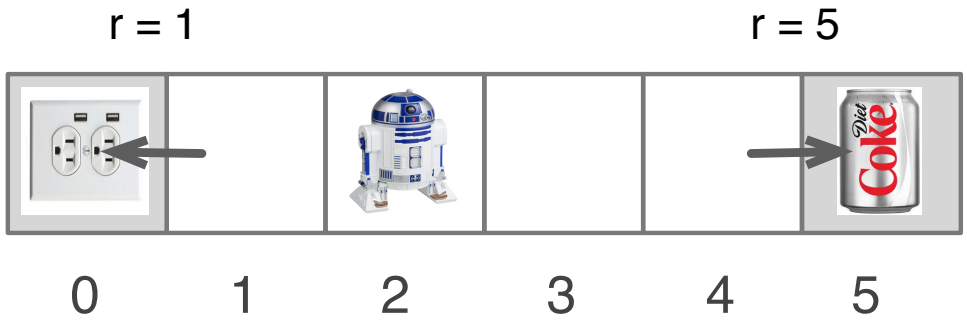
- Is a score of 5.68 good or bad, how good or how bad?
- The score does not tell you how to get the project to work
- You have to learn how yourself by maximizing your scores



- Robot has no knowledge about maze before learning starts
- Robot explores maze and receives reward if it reaches goal
- Robot learns the "optimal" route to reach goal

AlphaGo used RL to beat world No. 1 ranked player in 3-game match in 2017





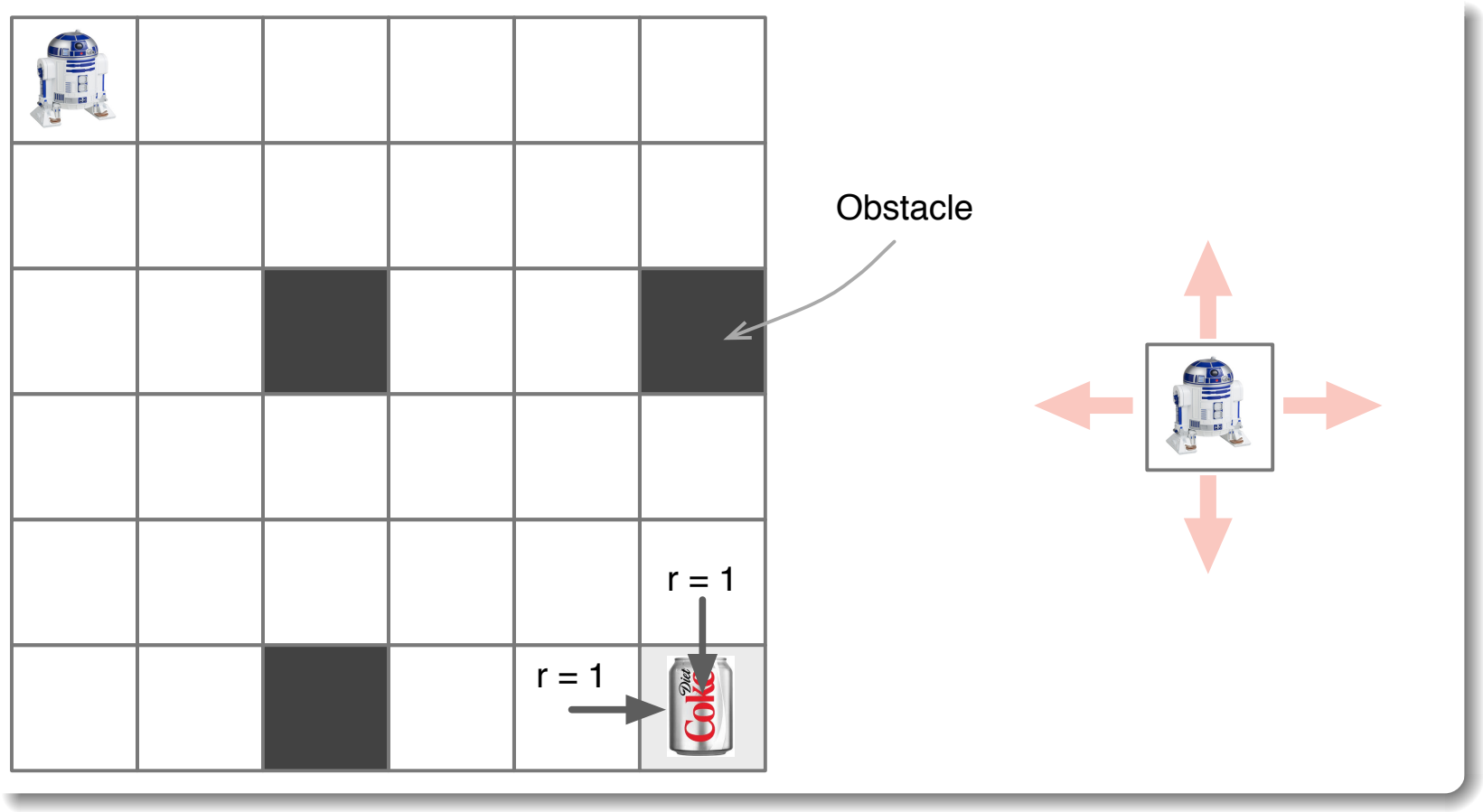
- Robot can move left or right
- Robot stops when reaching state 0 or 5 and gets reward
- Future rewards are discounted by 50% per step after 1^{st}
- What should robot do to maximize reward?

Find out maximum reward starting from a given state

At state	Move left	Move right
1	**1**	$5 \cdot (1-0.5)^3 = 0.625$
2	$1 \cdot (1-0.5)^1 = 0.5$	$5 \cdot (1-0.5)^2 = \mathbf{1.25}$
3	$1 \cdot (1-0.5)^2 = 0.625$	$5 \cdot (1-0.5)^1 = \mathbf{2.5}$
4	$1 \cdot (1-0.5)^3 = 0.125$	

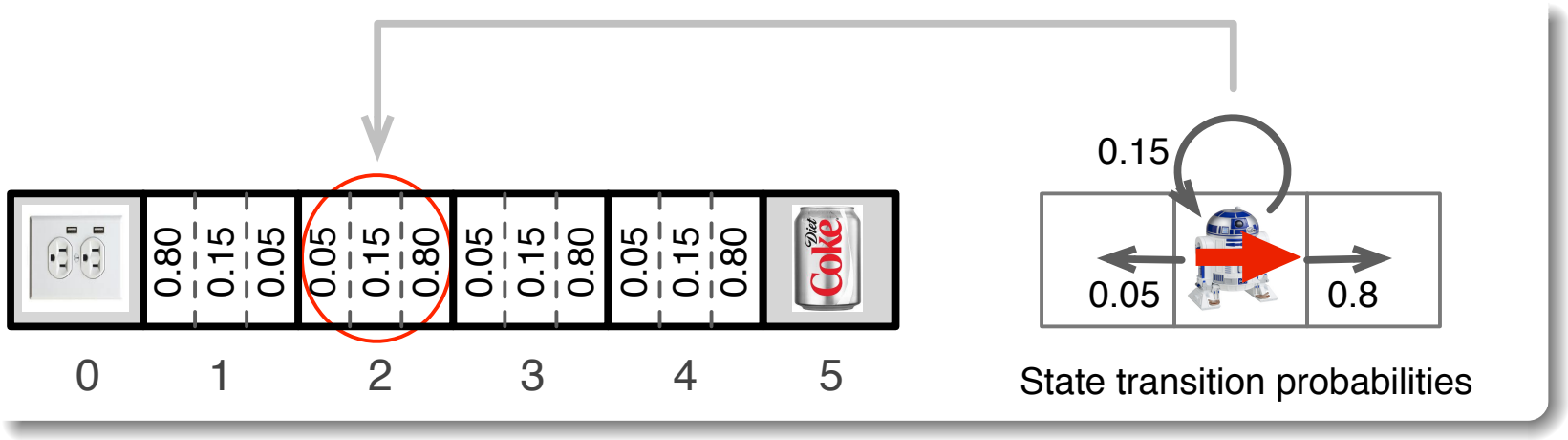
Actions corresponding to values in blue are the "best" actions

How does robot maximize reward in this case?

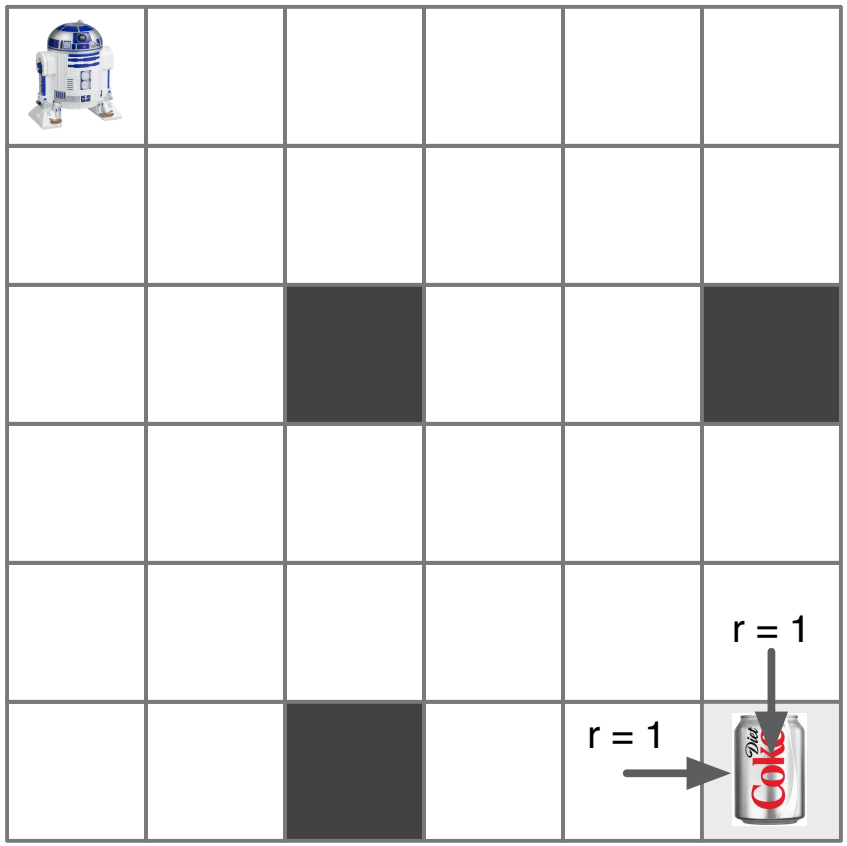


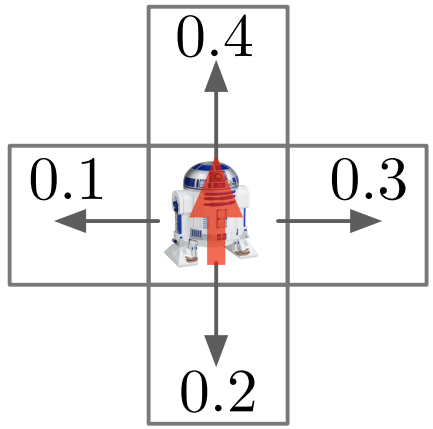
For complex situation, we need formal method

What if robot behavior is uncertain?



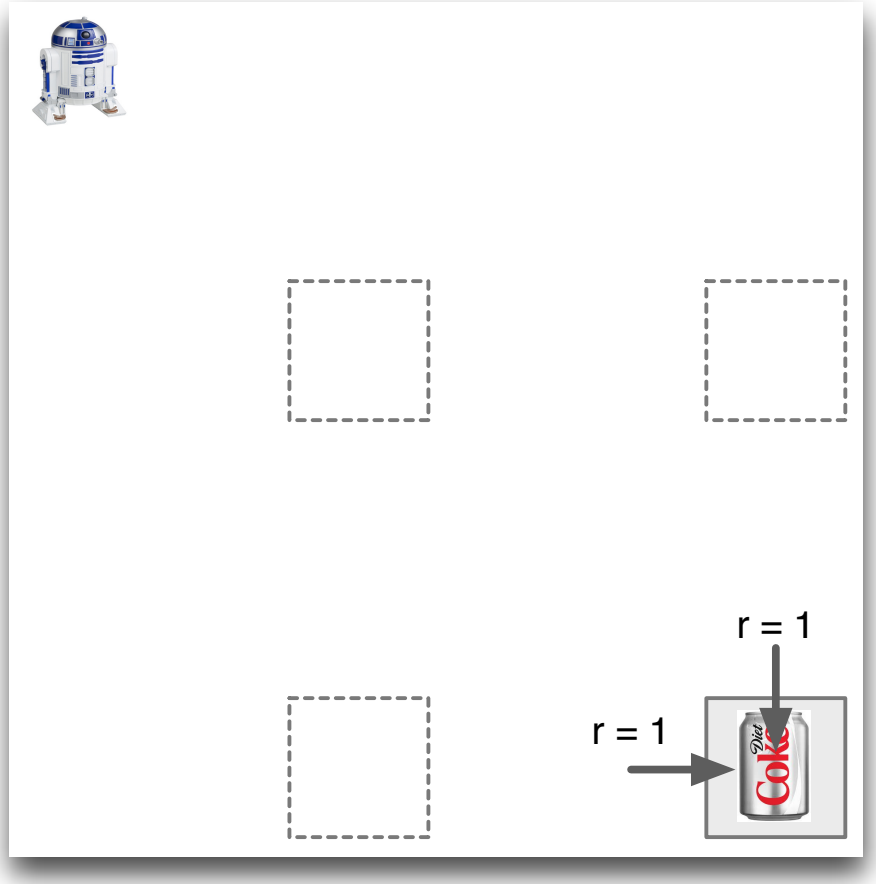
What if situation is both complex and uncertain?



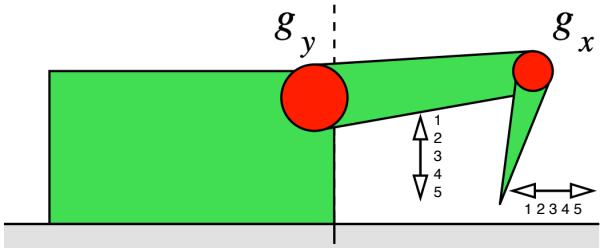


State transition probabilities

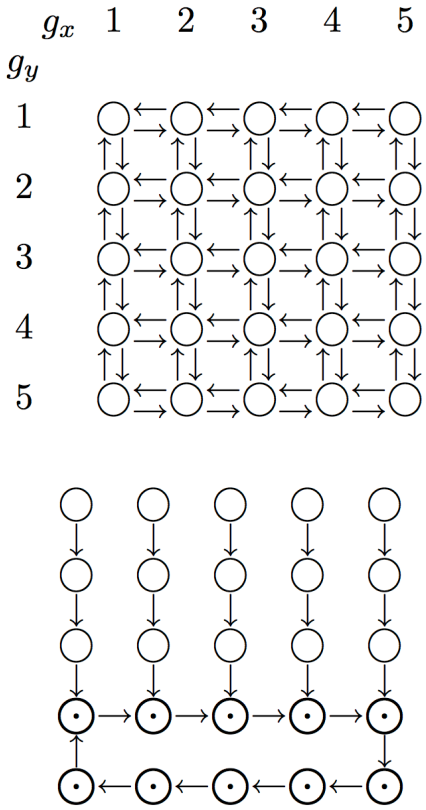
What if situation is complex and unknown (e.g., robot does not know where obstacles are)?



We need systematic approach to solve such problems



- 2 joints for actuating upper and lower arms
- Motion of each arm is discretized into 5 states
- Reward = distance of forward motion/step time



Tokic, M., Ertel, W., and Fessler, J., The crawler, a class room demonstrator for reinforcement learning. Proc. of the 22nd International Florida Artificial Intelligence Research Society Conference, 160-165, 2009

Another application example: A Robot Gatekeeper

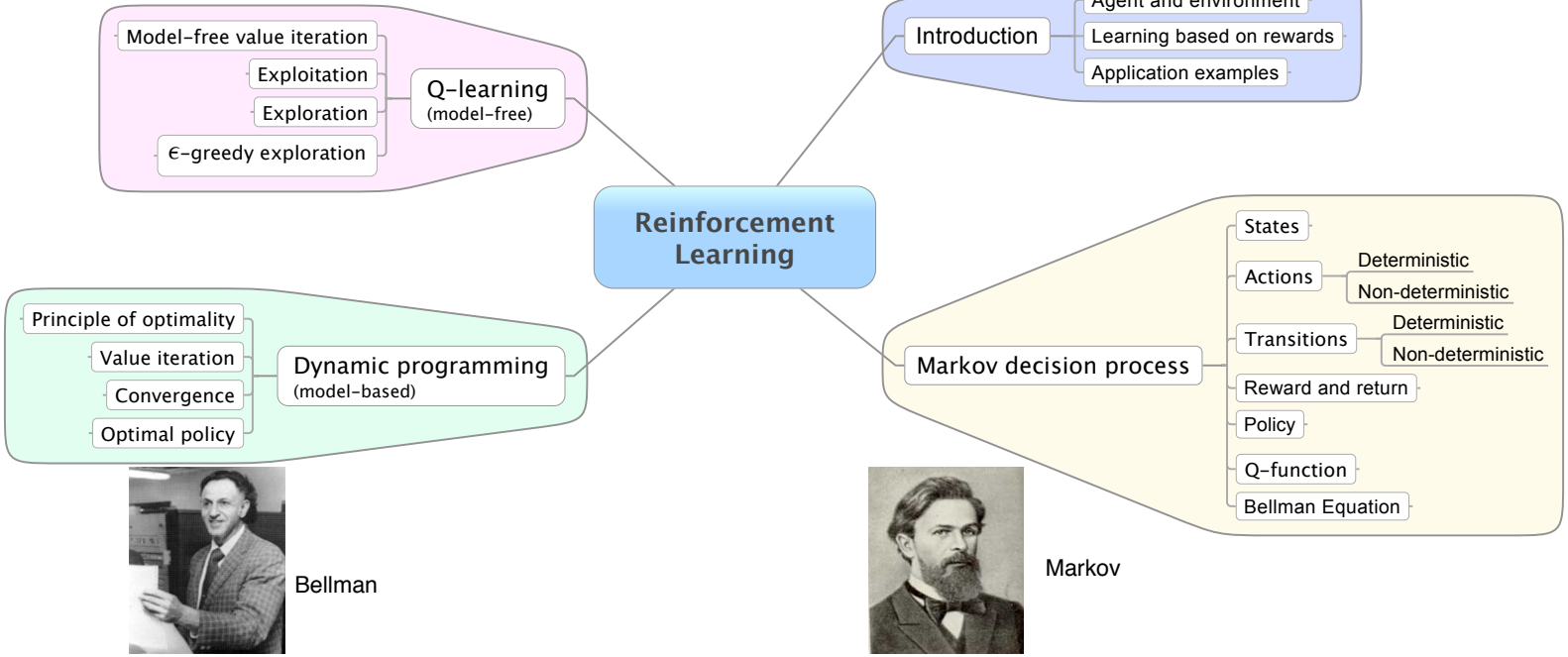
Adam, S., Busoniu, L. and Babuska, R., Experience replay for real-time reinforcement learning control
IEEE Trans. on Sys., Man, and Cybern., Part C: App. and Rev., vol. 42, no. 2, 201–212, 2012

- Unsupervised: Agent does not know the desired output
- Uncertain or unknown system behavior
- Reward received after each action is the only feedback
- Objective: Maximize total reward when executing a task

Reinforcement learning: Outline

Watkins

Sutton and Barto
and many others



Bellman

Markov

- A set of states: $S = \{s_1, s_2, \ldots\}$
- A set of actions: $A = \{a_1, a_2, \ldots, \}$
- A transition function

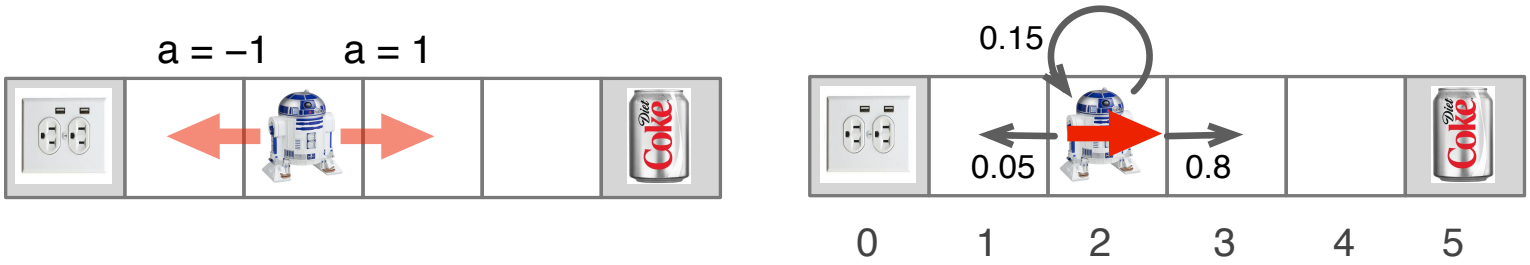
  Deterministic:

$$\bar{f}(s, a) = s', \qquad s' \in S$$

  Non-deterministic:

$$f\left(s, a, s'\right) = P\left(s_{t+1} = s' \middle| s_t = s, a_t = a\right) \equiv P_{ss'}^{a}$$

  where $P(a|b) =$ Probability of a being true under condition b



$$S = \{0, 1, 2, 3, 4, 5\}, \quad A = \{-1, 1\}$$

Suppose that due to uncertainties in robot's operation (such as slippery floor or error in robot position-tracking system, etc.), effect of taking $a = 1$ at state 2 becomes stochastic

Transition probabilities are:

$$\begin{cases} f(2,1,3) = 0.8 \\ f(2,1,2) = 0.15 \\ f(2,1,1) = 0.05 \end{cases}$$

Note that $$\sum_{i=1}^{3} f(2,1,i) = 1$$

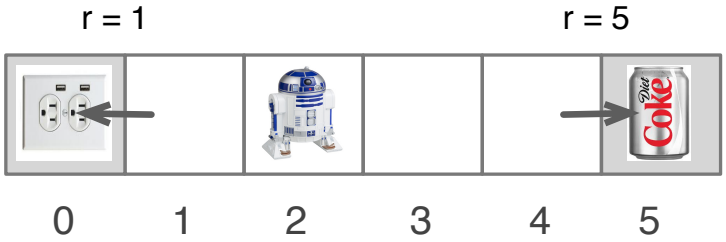
- Agent takes action a to move from state s to s'
- Agent receives reward r one time-step later (i.e., at s')

Reward for taking a_t at s_t at time step t and reaching s_{t+1}

$$r_{t+1} = \rho\left(s_t, a_t, s_{t+1}\right)$$

where $\rho : S \times A \times S \to \mathbb{R}$ is called the reward function

Example: Robot takes action 1 at state 4 at time step t



$$r_{t+1} = \rho\left(4, 1, 5\right) = 5$$

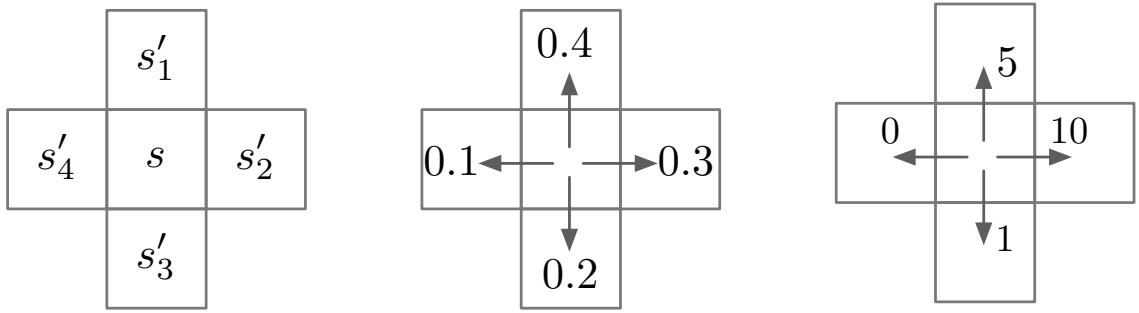
For non-deterministic transitions:

Reward for taking a at s is characterized by expected value of r_{t+1} over all possible new states:

$$\mathbb{E}\Big[r_{t+1}\Big|\, s_t = s\Big] = \sum_{s'} \Big(P^a_{ss'}\, r_{t+1}\Big|\, s_{t+1} = s'\Big) = \sum_{s'} P^a_{ss'}\, \rho(s, a, s')$$

- $P^a_{ss'}$ = probability of reaching s' after taking a at s.
- Summation over s' accounts for all possible new states

- Taking action a at state s

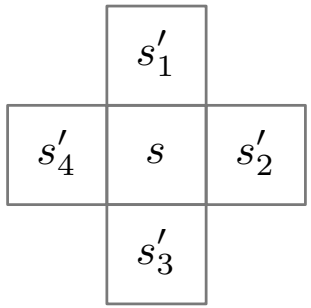


States Transition probabilities Rewards

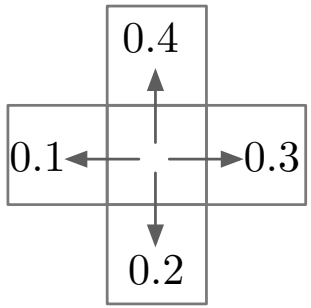
$$
\begin{aligned}
r_{t+1}|_{s_{t+1}=s'_1} = \rho\left(s, a, s'_1\right) &= 5 \\
r_{t+1}|_{s_{t+1}=s'_2} = \rho\left(s, a, s'_2\right) &= 10 \\
r_{t+1}|_{s_{t+1}=s'_3} = \rho\left(s, a, s'_3\right) &= 1 \\
r_{t+1}|_{s_{t+1}=s'_4} = \rho\left(s, a, s'_4\right) &= 0
\end{aligned}
$$

$$
\begin{aligned}
f\left(s, a, s'_1\right) &= P\left(s_{t+1} = s'_1 | \, s_t = s, a_t = a\right) \equiv P^a_{ss'_1} = 0.4 \\
f\left(s, a, s'_2\right) &= P\left(s_{t+1} = s'_2 | \, s_t = s, a_t = a\right) \equiv P^a_{ss'_2} = 0.3 \\
f\left(s, a, s'_3\right) &= P\left(s_{t+1} = s'_3 | \, s_t = s, a_t = a\right) \equiv P^a_{ss'_3} = 0.2 \\
f\left(s, a, s'_4\right) &= P\left(s_{t+1} = s'_4 | \, s_t = s, a_t = a\right) \equiv P^a_{ss'_4} = 0.1
\end{aligned}
$$

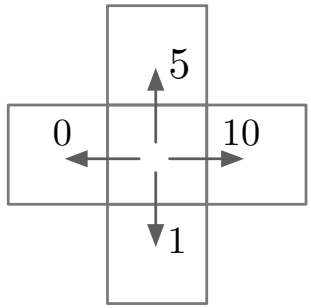
Determination of reward when transition is non-deterministic: An example



States



Transition probabilities



Rewards

$$
\begin{aligned}
\mathbb{E}\left[r_{t+1}\right] &= \sum_{s'}\left(P^a_{ss'}\ r_{t+1}\right) = \sum_{s'} P^a_{ss'}\rho(s,a,s') \\
&= P^a_{ss'_1}\,\rho\left(s,a,s'_1\right) + P^a_{ss'_2}\,\rho\left(s,a,s'_2\right) \\
&\quad + P^a_{ss'_3}\,\rho\left(s,a,s'_3\right) + P^a_{ss'_4}\,\rho\left(s,a,s'_4\right) \\
&= 0.4\times 5 + 0.3\times 10 + 0.2\times 1 + 0.1\times 0 \\
&= 5.2
\end{aligned}
$$

- Agent starts from initial state and reaches s after t time steps
- Question: What is the total reward from this point onward if agent continues to make transitions?

Return R_t

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$

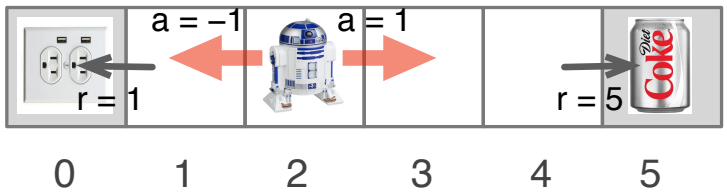
- $k =$ index of time steps after t, with $k = 0$ being 1^{st} step
- γ is called the discount rate, with $0 \leq \gamma \leq 1$

Back

Return R_t

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$

- R_t determines present value of future rewards
- Reward received k steps in future is discounted by factor of γ^{k-1}
- A small γ forces agent to focus more on immediate rewards from next few steps and heavily discount rewards from future steps
- A large γ forces agent to take into account future rewards more strongly, i.e., agent becomes more farsighted



$$\begin{aligned} S &= \{0,1,2,3,4,5\} \\ A &= \{-1,1\} \\ \gamma &= 0.5 \end{aligned}$$

$$\bar{f}(s,a) = \begin{cases} s+a & \text{if } 1 \le s \le 4 \\ s & \text{if } s = 0 \text{ or } s = 5 \end{cases}$$

$$\rho(s,a,s') = \begin{cases} 5 & \text{if } s \neq 5 \text{ and } s' = 5 \\ 1 & \text{if } s \neq 0 \text{ and } s' = 0 \\ 0 & \text{otherwise} \end{cases}$$

- 3 transitions to move from state 2 to 5 without changing direction

$$\begin{aligned} R_t &= \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} = r_{t+1} + \gamma\, r_{t+2} + \gamma^2 r_{t+3} \\ &= \rho(2,1,3) + 0.5 \cdot \rho(3,1,4) + 0.5^2 \cdot \rho(4,1,5) \\ &= 0 + 0 + 0.25 \cdot 5 \\ &= 1.25 \end{aligned}$$

- Decision-making mechanism for controlling a Markov process
- Specifies which action to take at a state
- Can be deterministic or non-deterministic

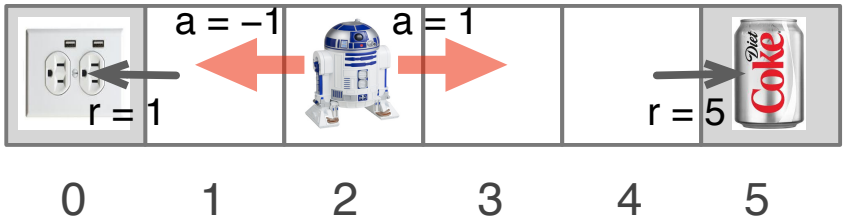
Deterministic: Specifies particular action to be taken at particular state

$$\pi : S \to A; \quad \pi(s) = a,\ s \in S, a \in A$$

Non-deterministic: Specifies probability of action being taken at particular state

$$\pi : S \times A \to [0, 1]; \quad \pi(s, a) = P\left(a_t = a \middle| s_t = s\right)$$

In this module, we consider deterministic policies only

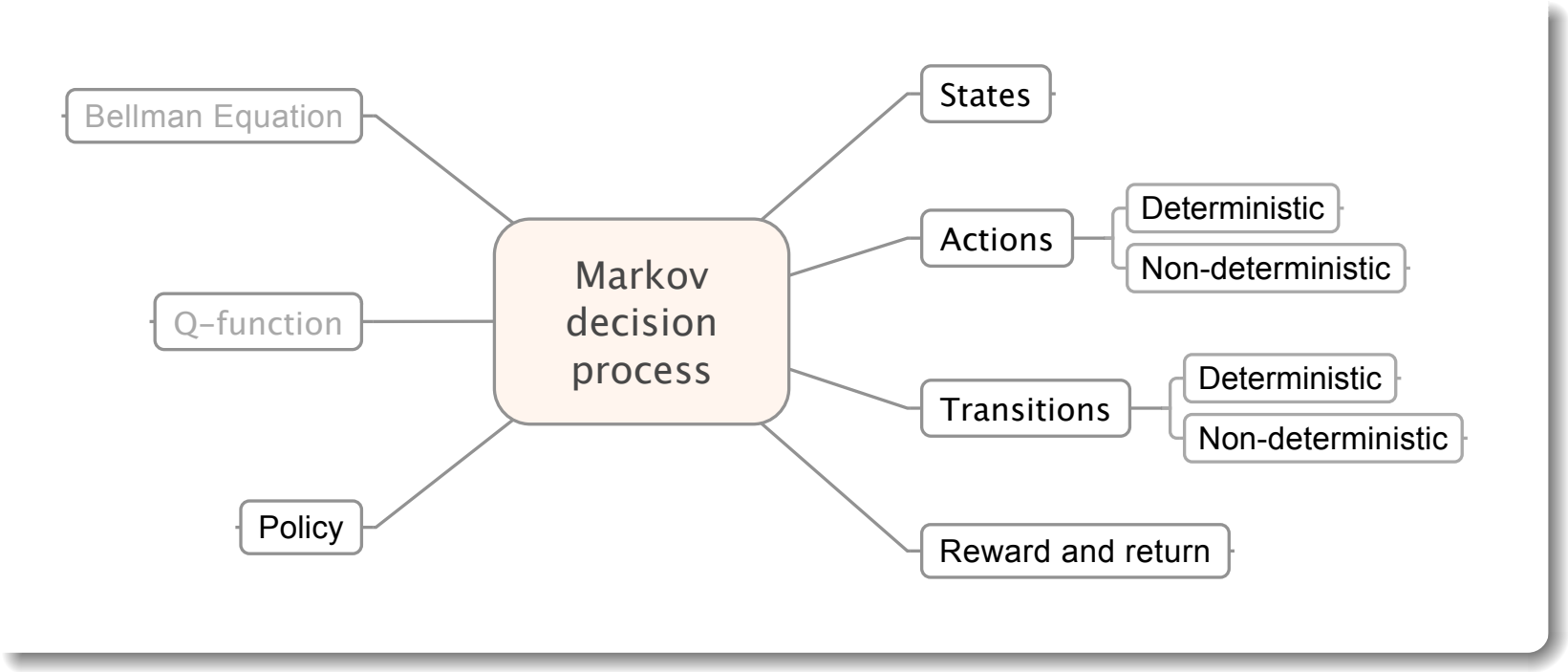


Suppose that robot must be controlled to behave as follows:

- In state 1, it must take $a = -1$
- In state 0 or 5, it can take either action
- In any of remaining states, it must take $a = 1$

The policy that governs behavior of robot is

$$\begin{cases} \pi(1) = -1 & \\ \pi(s) = \pm 1 & \text{if} \quad s = 0, 5 \\ \pi(s) = 1 & \text{if} \quad s = 2, 3, 4 \end{cases}$$



Key questions to be answered:

1. What is the "best" policy ?
2. How to find the "best" policy?

Answers require Q-function and Bellman Equation

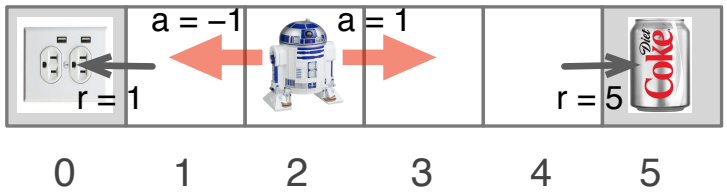
- Measures the "worth" of taking a at s under π
- "Worth" is represented by

$$Q^\pi : S \times A \to \mathbb{R}$$

defined as the expected return from taking action a at state s at time step t, and thereafter following policy π

$$Q^\pi(s, a) = \mathbb{E}^\pi \left[R_t \middle| s_t = s \right]$$

View



$$\begin{aligned} S &= \{0,1,2,3,4,5\} \\ A &= \{-1,1\} \\ \gamma &= 0.5 \end{aligned}$$

$$\bar{f}(s,a) = \begin{cases} s+a & \text{if } 1 \le s \le 4 \\ s & \text{if } s = 0 \text{ or } s = 5 \end{cases}$$

$$\rho(s,a,s') = \begin{cases} 5 & \text{if } s \neq 5 \text{ and } s' = 5 \\ 1 & \text{if } s \neq 0 \text{ and } s' = 0 \\ 0 & \text{otherwise} \end{cases}$$

- Calculate $Q^\pi(2,1)$ with $\pi(2) = 1, \pi(3) = 1$, and $\pi(4) = 1$

Since all transitions are deterministic, Q-function degenerates to just $Q^\pi(2,1) = (R_t | \, s_t = 2)$, i.e.,

$$\begin{aligned} Q^\pi(2,1) &= \mathbb{E}[R_t | s_t = 2] = (R_t | \, s_t = 2) \\ &= r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} \\ &= \rho(2,1,3) + 0.5 \cdot \rho(3,1,4) + 0.5^2 \cdot \rho(4,1,5) \\ &= 0 + 0 + 0.25 \cdot 5 = 1.25 \end{aligned}$$

Example about non-deterministic transitions is presented later in connection with Bellman Equation

- Q-function measures "worth" of state-action pair (s, a) in terms of rewards received when agent is executing a task
- An optimal policy is one that maximizes (with respect to a given task) values of Q-function over all possible (s, a) pairs
- Solving a reinforcement learning problems is about finding an optimal policy
- To do this requires the Bellman Equation

From definition of Q-function:

$$
\begin{aligned}
Q^{\pi}(s,a) &= \mathbb{E}^{\pi}\left[R_t \middle| s_t = s\right] \\
&= \mathbb{E}^{\pi}\left[\sum_{k=0}^{\infty} \gamma^k \, r_{t+k+1} \middle| s_t = s\right] \\
&= \mathbb{E}^{\pi}\left[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots \middle| s_t = s\right] \\
&= \mathbb{E}^{\pi}\left[\left(r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k \, r_{t+k+2}\right) \middle| s_t = s\right] \quad \left(\text{Isolating } r_{t+1}\right) \\
&= \mathbb{E}^{\pi}\left[r_{t+1}\right] + \mathbb{E}^{\pi}\left[\gamma \sum_{k=0}^{\infty} \gamma^k \, r_{t+k+2} \middle| s_t = s\right]
\end{aligned}
$$

- $\mathbb{E}\left[r_{t+1}\right] = \sum_{s'} P^{a}_{ss'} \, \rho(s,a,s')$
- Linearity of expectation operator: $\mathbb{E}[X+Y] = \mathbb{E}[X] + \mathbb{E}[Y]$

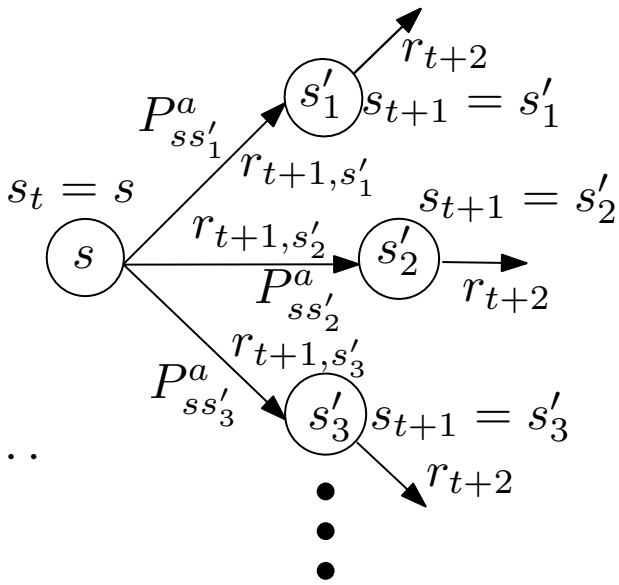
$$\sum_{k=0}^{\infty}\left[\gamma^k\, r_{t+k+2}\,\middle|\, s_t=s\right]$$

$$= P^a_{ss'_1}\sum_{k=0}^{\infty}\left[\gamma^k\, r_{t+k+2}\,\middle|\, s_{t+1}=s'_1\right]$$

$$+ P^a_{ss'_2}\sum_{k=0}^{\infty}\left[\gamma^k\, r_{t+k+2}\,\middle|\, s_{t+1}=s'_2\right]$$

$$+ P^a_{ss'_3}\sum_{k=0}^{\infty}\left[\gamma^k\, r_{t+k+2}\,\middle|\, s_{t+1}=s'_3\right]+\cdots$$

$$= \sum_{s'} P^a_{ss'}\left[\sum_{k=0}^{\infty}\gamma^k\, r_{t+k+2}\,\middle|\, s_{t+1}=s'\right]$$



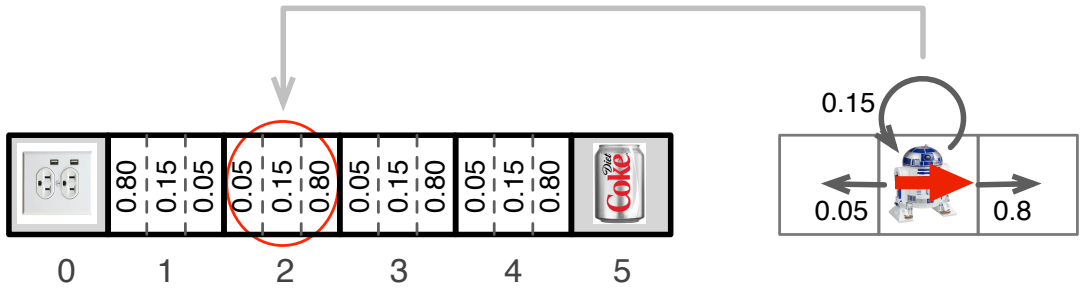
$$\mathbb{E}^\pi\left[\gamma\sum_{k=0}^{\infty}\gamma^k\, r_{t+k+2}\,\middle|\, s_t=s\right] = \mathbb{E}^\pi\left[\gamma\sum_{s'} P^a_{ss'}\left[\sum_{k=0}^{\infty}\gamma^k\, r_{t+k+2}\,\middle|\, s_{t+1}=s'\right]\right]$$

$$= \sum_{s'} P^a_{ss'}\gamma\underbrace{\mathbb{E}^\pi\left[\sum_{k=0}^{\infty}\gamma^k\, r_{t+k+2}\,\middle|\, s_{t+1}=s'\right]}_{\text{This is } Q^\pi(s',a') \text{ by definition}} = \sum_{s'} P^a_{ss'}\gamma\, Q^\pi(s',a')$$

Bellman Equation (recursive relationship between $Q^\pi(s,a)$ and $Q^\pi(s',a')$)

$$\begin{aligned} Q^\pi(s,a) &= \mathbb{E}^\pi\left[r_{t+1}\right] + \mathbb{E}^\pi\left[\gamma \sum_{k=0}^{\infty} \gamma^k \, r_{t+k+2} \middle| \, s_t = s\right] \\ &= \sum_{s'} P^a_{ss'} \, \rho(s,a,s') + \sum_{s'} P^a_{ss'} \gamma \, Q^\pi\left(s',a'\right) \\ &= \sum_{s'} P^a_{ss'} \left(\rho(s,a,s') + \gamma Q^\pi\left(s',a'\right)\right) \end{aligned}$$

Under a given policy π, value of taking a at s must equal to expected reward of transitioning into next state s', i.e., $\rho(s,a,s')$, plus discounted expected value of taking $a' = \pi(s')$ at s'



- Robot stops at 0 or 5
- $\gamma = 0.5$

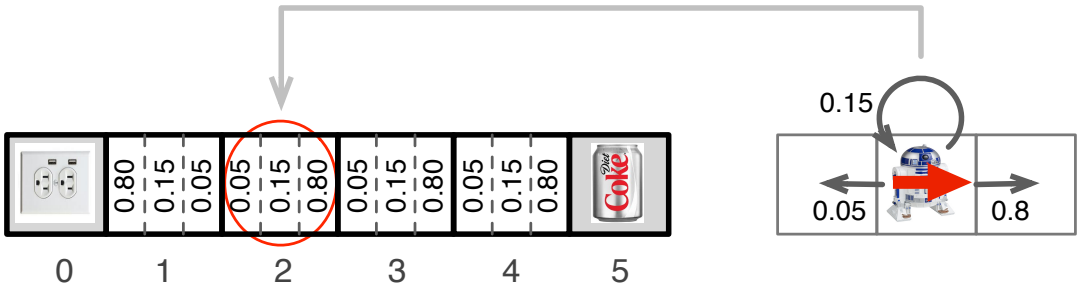
Reward function:

$$\rho(s, a, s') = \begin{cases} 5 & \text{if } s \neq 5 \text{ and } s' = 5 \\ 1 & \text{if } s \neq 0 \text{ and } s' = 0 \\ 0 & \text{otherwise} \end{cases}$$

Policy:

$$\begin{aligned} \pi(1) &= -1 \\ \pi(s) &= 1 \quad \text{if} \quad s = 2, 3, 4 \\ \pi(s) &= \pm 1 \quad \text{if} \quad s = 0, 5 \end{aligned}$$

- Determine the value of $Q^{\pi}(2, 1)$



- Robot stops at 0 or 5
- $\gamma = 0.5$

$$f(1,-1,0) = 0.80 \equiv P_{10}^{-1} \qquad f(2,1,1) = 0.05 \equiv P_{21}^{1}$$
$$f(1,-1,1) = 0.15 \equiv P_{11}^{-1} \qquad f(2,1,2) = 0.15 \equiv P_{22}^{1}$$
$$f(1,-1,2) = 0.05 \equiv P_{12}^{-1} \qquad f(2,1,3) = 0.80 \equiv P_{23}^{1}$$

$$f(3,1,2) = 0.05 \equiv P_{32}^{1} \qquad f(4,1,3) = 0.05 \equiv P_{43}^{1}$$
$$f(3,1,3) = 0.15 \equiv P_{33}^{1} \qquad f(4,1,4) = 0.15 \equiv P_{44}^{1}$$
$$f(3,1,4) = 0.80 \equiv P_{34}^{1} \qquad f(4,1,5) = 0.80 \equiv P_{45}^{1}$$
$$f(0,\pm 1,0) = 1 \equiv P_{00}^{\pm 1} \qquad f(5,\pm 1,5) = 1 \equiv P_{55}^{\pm 1}$$

$$\rho(0,1,0) = 0 \quad \rho(0,-1,0) = 0 \quad \rho(1,1,2) = 0$$
$$\rho(1,-1,0) = 1 \quad \rho(2,-1,1) = 0 \quad \rho(2,1,2) = 0$$
$$\rho(3,-1,2) = 0 \quad \rho(3,1,4) = 0 \quad \rho(4,1,5) = 5$$
$$\rho(4,-1,3) = 0 \quad \rho(5,1,5) = 0 \quad \rho(5,-1,5) = 0$$

Applying the Bellman Equation yields

$$
\begin{aligned}
Q^{\pi}(2,1) &= \sum_{s'} P^{a}_{ss'} \left(\rho(s,a,s') + \gamma Q^{\pi}\left(s',a'\right) \right) \\
&= \sum_{j=1}^{3} P^{1}_{2j} \left(\rho(2,1,j) + \gamma Q^{\pi}\left(j,\pi(j)\right) \right) \\
&= P^{1}_{21} \left(\rho(2,1,1) + \gamma Q^{\pi}\left(1,-1\right) \right) \quad \text{(Note: } j=1 \text{ and } \pi(1)=-1\text{)} \\
&+ P^{1}_{22} \left(\rho(2,1,2) + \gamma Q^{\pi}\left(2,1\right) \right) \quad \text{(Note: } j=2 \text{ and } \pi(2)=1\text{)} \\
&+ P^{1}_{23} \left(\rho(2,1,3) + \gamma Q^{\pi}\left(3,1\right) \right) \quad \text{(Note: } j=3 \text{ and } \pi(3)=1\text{)} \\
&= 0.05 \cdot \left(0 + 0.5 Q^{\pi}\left(1,-1\right) \right) \\
&+ 0.15 \cdot \left(0 + 0.5 Q^{\pi}\left(2,1\right) \right) \\
&+ 0.80 \cdot \left(0 + 0.5 Q^{\pi}\left(3,1\right) \right) \\
&= 0.025 Q^{\pi}(1,-1) + 0.075 Q^{\pi}(2,1) + 0.4 Q^{\pi}(3,1)
\end{aligned}
$$

Expressions for other Q-function terms are derived in supplementary notes

$$\begin{cases} Q^\pi(2,1) &= 0.025Q^\pi(1,-1) + 0.075Q^\pi(2,1) + 0.4Q^\pi(3,1) \\ Q^\pi(1,-1) &= 0.8 + 0.075Q^\pi(1,-1) + 0.025Q^\pi(2,1) \\ Q^\pi(3,1) &= 0.025Q^\pi(2,1) + 0.075Q^\pi(3,1) + 0.4Q^\pi(4,1) \\ Q^\pi(4,1) &= 0.025Q^\pi(3,1) + 0.075Q^\pi(4,1) + 4 \end{cases}$$

Solving this set of equations yields

$$\begin{aligned} Q^\pi(1,-1) &= 0.888 \\ Q^\pi(2,1) &= 0.852 \\ Q^\pi(3,1) &= 1.915 \\ Q^\pi(4,1) &= 4.376 \end{aligned}$$

Hence, $Q^\pi(2,1) = 0.852$.

Details in supplementary notes

- For a given task, number of possible policies is finite
- Q-function can be used to compare policies
- The policy whose Q-function values are greater or equal to that of all other policies is called an optimal policy

  This Q-function is called the optimal Q-function

$$Q^*(s,a) = \max_\pi Q^\pi(s,a)$$

- Any function π^* that selects at s the action a with optimal $Q^*(s,a)$ is an optimal policy

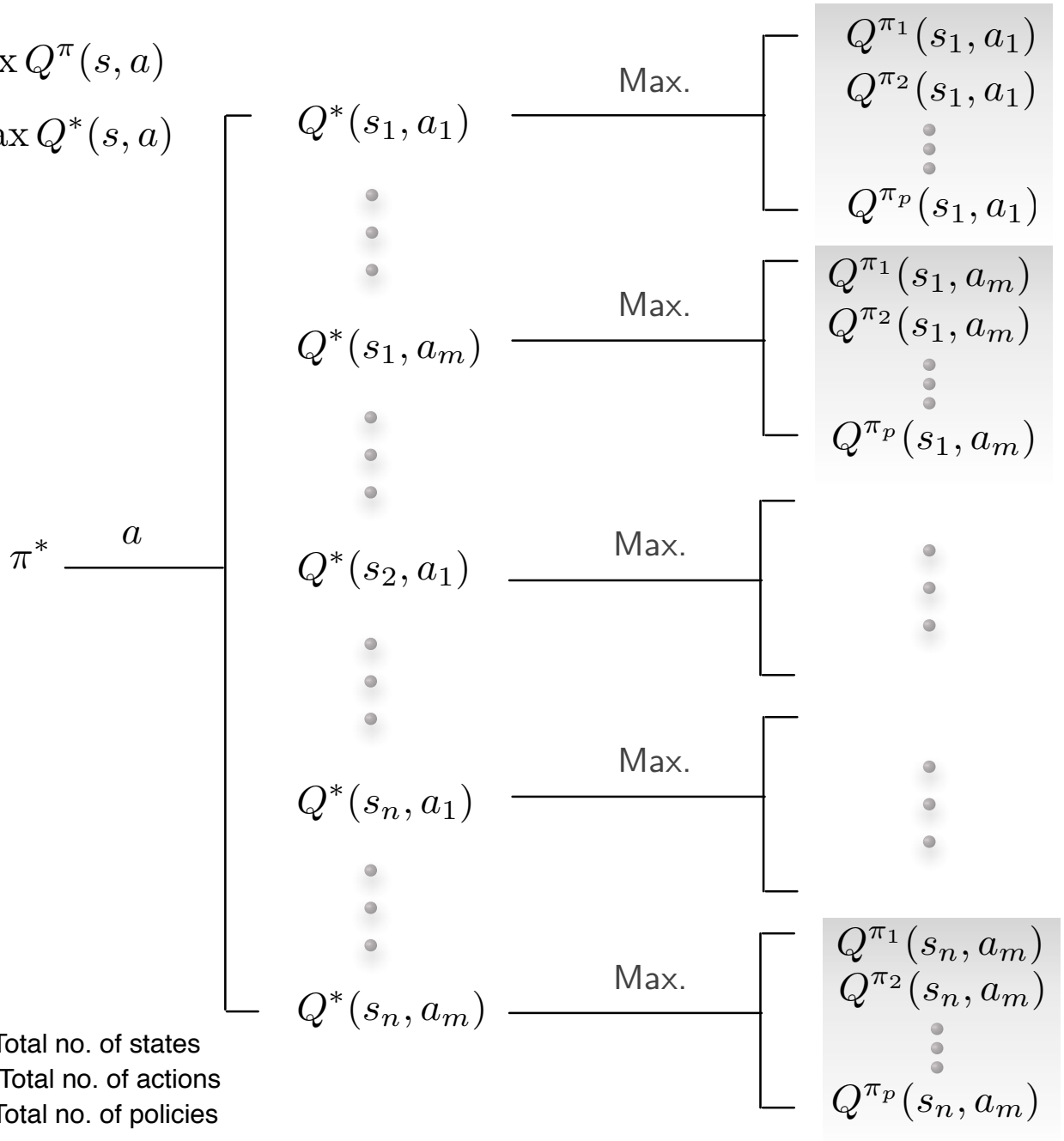
$$\pi^*(s) \in \arg\max_a Q^*(s,a)$$ ▸ View

- Values of $Q^*(s,a)$ are unique
- Possibly more than one π^* for the same $Q^*(s,a)$ values

Optimal policy and optimal Q-function: Summary

$$Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a)$$

$$\pi^*(s) \in \arg\max_{a} Q^*(s,a)$$



n: Total no. of states
m: Total no. of actions
p: Total no. of policies

$Q^*(s,a)$ satisfies Bellman Equation:

$$Q^*(s,a) = \sum_{s'} P^a_{ss'} \left(\rho(s,a,s') + \gamma \max_{a'} Q^* \left(s', a' \right) \right)$$

$Q^*(s,a)$ is the sum of

1. Reward in current state s for the chosen action a, and
2. Discounted value of Q-function for the "best" action available in the successor state s'

$$Q^*(s,a) = \sum_{s'} P^a_{ss'} \left(\rho(s,a,s') + \gamma \max_{a'} Q^* \left(s', a' \right) \right)$$

We can determine π^* recursively by picking at s' the action that yields maximum Q^* for s'

Principle of Optimality

An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision.

Also see example in supplementary notes

Use Dynamic Programming to solve for Q^* to get π^*

- Value iteration
- Policy iteration

Input: state-transition probability f
reward function ρ
discount factor γ

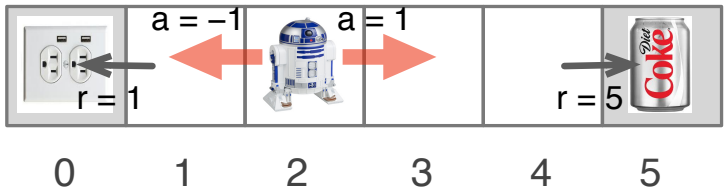
Initialize Q-function, e.g., $Q_0 \leftarrow 0$
Repeat for each l
$Q \leftarrow Q_l$
For every (s, a) do

$$Q(s,a) \leftarrow \sum_{s'} P^a_{ss'} \left(\rho(s,a,s') + \gamma \max_{a'} Q\left(s', a'\right) \right)$$

End for-loop
$Q_{l+1} \leftarrow Q$
Until $Q_{l+1} = Q_l$

Output: $Q^* = Q_l$

In practice, stop when $|Q_{l+1} - Q_l|$ is below pre-defined threshold



$$S = \{0, 1, 2, 3, 4, 5\}$$
$$A = \{-1, 1\}$$
$$\gamma = 0.5$$

$$\bar{f}(s, a) = \begin{cases} s + a & \text{if } 1 \leq s \leq 4 \\ s & \text{if } s = 0 \text{ or } s = 5 \end{cases}$$

$$\rho(s, a, s') = \begin{cases} 5 & \text{if } s \neq 5 \text{ and } s' = 5 \\ 1 & \text{if } s \neq 0 \text{ and } s' = 0 \\ 0 & \text{otherwise} \end{cases}$$

State	0	1	2	3	4	5
Q_0	0.0|0.0	0.000|0.000	0.000|0.000	0.000|0.000	0.000|0.000	0.0|0.0
Q_1	0.0|0.0	1.000|0.000	0.500|0.000	0.250|0.000	0.125|5.000	0.0|0.0
Q_2	0.0|0.0	1.000|0.250	0.500|0.125	0.250|0.250	1.250|5.000	0.0|0.0
Q_3	0.0|0.0	1.000|0.250	0.500|1.250	0.625|2.500	1.250|5.000	0.0|0.0
Q_4	0.0|0.0	1.000|0.625	0.500|1.250	0.625|2.500	1.250|5.000	0.0|0.0
Q_5	0.0|0.0	1.000|0.625	0.500|1.250	0.625|2.500	1.250|5.000	0.0|0.0

Note: $Q^* = Q_5$

Expression "$m|n$" indicates $Q = m$ when $a = -1$ and $Q = n$ when $a = 1$

Detailed calculations are illustrated in supplementary notes

Value iteration: Example with non-deterministic transitions

(s,a)	$f(s,a,0)$	$f(s,a,1)$	$f(s,a,2)$	$f(s,a,3)$	$f(s,a,4)$	$f(s,a,5)$
$(0,-1)$	1	0	0	0	0	0
$(1,-1)$	0.8	0.15	0.05	0	0	0
$(2,-1)$	0	0.8	0.15	0.05	0	0
$(3,-1)$	0	0	0.8	0.15	0.05	0
$(4,-1)$	0	0	0	0.8	0.15	0.05
$(5,-1)$	1	0	0	0	0	1
$(0,1)$	1	0	0	0	0	0
$(1,1)$	0.05	0.15	0.8	0	0	0
$(2,1)$	0	0.05	0.15	0.8	0	0
$(3,1)$	0	0	0.05	0.15	0.8	0
$(4,1)$	0	0	0	0.05	0.15	0.8
$(5,1)$	0	0	0	0	0	1

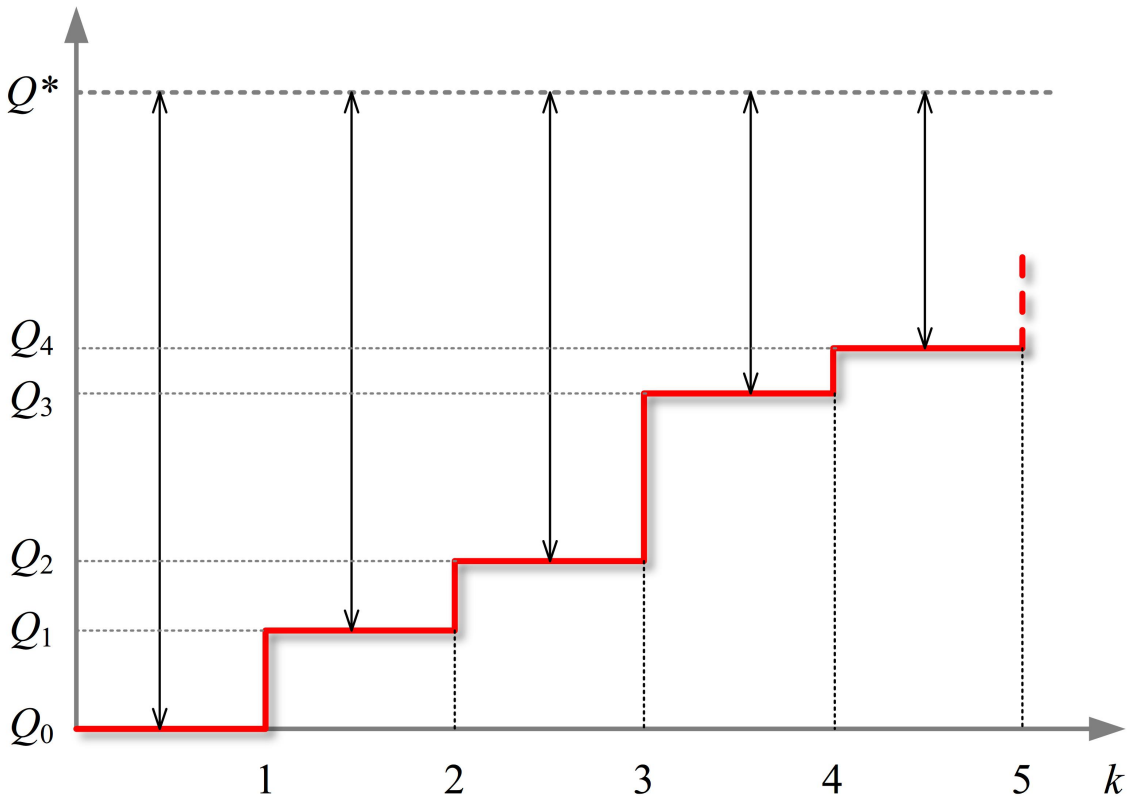
State	0	1	2	3	4	5
Q_0	$0.0\|0.0$	$0.000\|0.000$	$0.000\|0.000$	$0.000\|0.000$	$0.000\|0.000$	$0.0\|0.0$
Q_1	$0.0\|0.0$	$0.800\|0.110$	$0.320\|0.044$	$0.128\|0.018$	$0.301\|4.026$	$0.0\|0.0$
Q_2	$0.0\|0.0$	$8.868\|0.243$	$0.374\|0.101$	$0.260\|1.639$	$1.208\|4.343$	$0.0\|0.0$
Q_3	$0.0\|0.0$	$0.874\|0.265$	$0.419\|0.709$	$0.515\|1.878$	$1.327\|4.373$	$0.0\|0.0$
Q_4	$0.0\|0.0$	$0.883\|0.400$	$0.453\|0.826$	$0.581\|1.911$	$1.342\|4.376$	$0.0\|0.0$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
Q_{12}	$0.0\|0.0$	$0.888\|0.458$	$0.467\|0.852$	$0.594\|1.915$	$1.344\|4.376$	$0.0\|0.0$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
Q_{22}	$0.0\|0.0$	$0.888\|0.458$	$0.467\|0.852$	$0.594\|1.915$	$1.344\|4.376$	$0.0\|0.0$

Note: $Q^* = Q_{22}$

Detailed calculations are illustrated in supplementary notes

Proof of convergence takes two steps:

1. Show values of Q-function are bounded from one iteration to the next
2. Show difference between $Q_k(s,a)$ and $Q^*(s,a)$ decreases as k increases



Consider $Q_k(s,a)$ at iteration k

If $Q_k(s,a) \neq Q^*(s,a)$, then express maximum difference as δ_k

$$\delta_k = \max_{s,a} |Q^*(s,a) - Q_k(s,a)| \ , \quad \text{for all } s \text{ and } a$$

Removing maximum operator yields

$$|Q^*(s,a) - Q_k(s,a)| \leq \delta_k$$

That is

$$\begin{aligned} & -\delta_k \leq Q^*(s,a) - Q_k(s,a) \leq \delta_k \\ \Rightarrow \quad & -Q^*(s,a) - \delta_k \leq -Q_k(s,a) \leq -Q^*(s,a) + \delta_k \\ \Rightarrow \quad & Q^*(s,a) + \delta_k \geq Q_k(s,a) \geq Q^*(s,a) - \delta_k \\ \Rightarrow \quad & Q^*(s,a) - \delta_k \leq Q_k(s,a) \leq Q^*(s,a) + \delta_k \end{aligned}$$

Consider iteration $(k+1)$. The update rule is:

$$Q_{k+1}(s,a) = \sum_{s'} P^a_{ss'} \left(\rho(s,a,s') + \gamma \max_{a'} Q_k\left(s',a'\right) \right)$$

Since $Q_k(s',a') \leq Q^*(s',a') + \delta_k$, we have

$$\begin{aligned}
Q_{k+1}(s,a) &= \sum_{s'} P^a_{ss'} \left(\rho(s,a,s') + \gamma \max_{a'} Q_k\left(s',a'\right) \right) \\
&\leq \sum_{s'} P^a_{ss'} \left(\rho(s,a,s') + \gamma \left(\max_{a'} Q^*\left(s',a'\right) + \delta_k \right) \right) \\
&= \sum_{s'} P^a_{ss'} \left(\rho(s,a,s') + \gamma \max_{a'} Q^*\left(s',a'\right) + \gamma\delta_k \right) \\
&= \underbrace{\sum_{s'} P^a_{ss'} \left(\rho(s,a,s') + \gamma \max_{a'} Q^*\left(s',a'\right) \right)}_{\text{This is } Q^*(s,a)} + \gamma\delta_k \\
&= Q^*(s,a) + \gamma\delta_k
\end{aligned}$$

that is

$$Q_{k+1}(s,a) \leq Q^*(s,a) + \gamma\delta_k$$

Since $Q^*(s,a)$ is finite, $Q_{k+1}(s,a)$ is bounded if δ_k is bounded. But initially Q_0 is set to zero, and δ_k is the maximum absolute difference between Q and Q^* in each iteration, so Q_{k+1} is bounded

From Step 1: $Q_k(s',a') \geq Q^*(s',a') - \delta_k$. So

$$\begin{aligned} Q_{k+1}(s,a) &= \sum_{s'} P^a_{ss'} \left(\rho(s,a,s') + \gamma \max_{a'} Q_k\left(s',a'\right) \right) \\ &\geq \sum_{s'} P^a_{ss'} \left(\rho(s,a,s') + \gamma \max_{a'} \left(Q^*\left(s',a'\right) - \delta_k \right) \right) \\ &= \underbrace{\sum_{s'} P^a_{ss'} \left(\rho(s,a,s') + \gamma \max_{a'} Q^*\left(s',a'\right) \right)}_{\text{This is } Q^*(s,a)} - \gamma\delta_k \\ &= Q^*(s,a) - \gamma\delta_k \end{aligned}$$

that is $$Q^*(s,a) - \gamma\delta_k \leq Q_{k+1}(s,a)$$

or $$Q^*(s,a) - Q_{k+1}(s,a) \leq \gamma\delta_k$$

Therefore

$$\delta_{k+1} \equiv \max_{s,a} |Q^*(s,a) - Q_{k+1}(s,a)| \leq \max_{s,a} |\gamma\,\delta_k| = \gamma\,\delta_k$$

i.e.,

$$\delta_{k+1} \leq \gamma\,\delta_k$$

Since $0 \leq \gamma < 1$, error measure δ_k decreases to 0 as k goes to infinity

$$\delta_k = \max_{s,a} |Q^*(s,a) - Q_k(s,a)|$$

1. Q_{k+1} is bounded because

$$Q_{k+1}(s,a) \leq Q^*(s,a) + \gamma \delta_k$$

2. Error measure $\delta_k \to 0$ as $k \to \infty$ because

$$\delta_{k+1} \leq \gamma \, \delta_k$$

Hence, value iteration algorithm converges to $Q^*(s,a)$

Values of Q-function are optimal if they are greater or equal to that of all other policies for all (s, a) pairs, i.e.,

$$Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a)$$

Greedy policy

At each s, select a that yields the largest value for the Q-function. When multiple choices are available, such a can be picked randomly

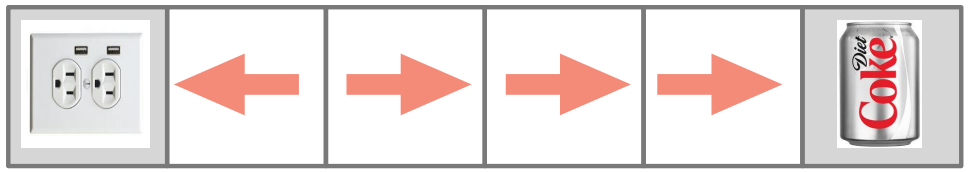
A greedy policy is optimal, i.e.,

$$\pi^*(s) \in \arg\max_{a} Q^*(s,a)$$

State	0	1	2	3	4	5
Q_5	$0.0\|0.0$	$1.0\|0.625$	$0.5\|1.25$	$0.625\|2.5$	$1.25\|5.0$	$0.0\|0.0$
π^*		-1	1	1	1	

- Optimal values are $Q^* = Q_5$
- Take state 1 for example:
  - Optimal values are: $Q^*(1,-1) = 1$ and $Q^*(1,1) = 0.625$
  - So $\pi^*(1) = -1$
- Applying same reasoning on other states yields

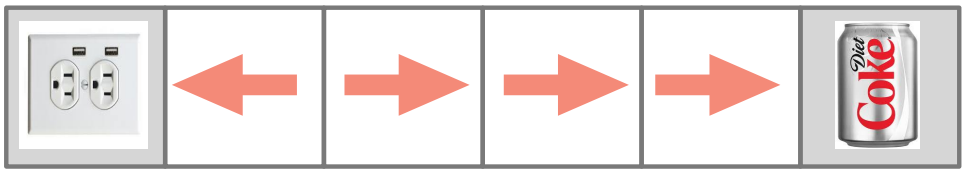
$$\pi^*(1) = -1, \quad \pi^*(2) = 1, \quad \pi^*(3) = 1, \quad \pi^*(4) = 1$$

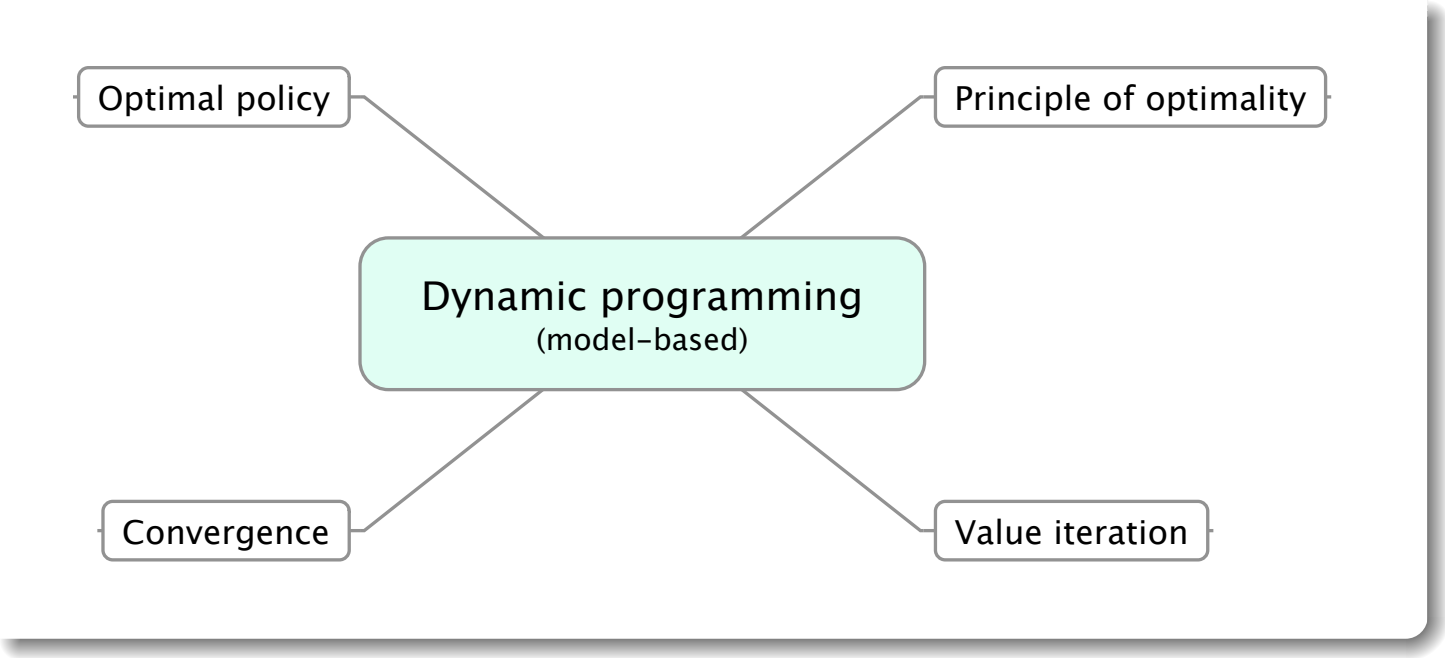


State	0	1	2	3	4	5
Q_{22}	$0\|0$	$0.888\|0.458$	$0.467\|0.852$	$0.594\|1.915$	$1.344\|4.376$	$0\|0$
π^*		-1	1	1	1	

- Optimal values are $Q^* = Q_{22}$
- Optimal policy is

$$\pi^*(1) = -1, \quad \pi^*(2) = 1, \quad \pi^*(3) = 1, \quad \pi^*(4) = 1$$





- Dynamic programming requires state-transition model
- What if no such model is available?

Q-learning

- No need for state-transition model
- Uses solely reward received from observed state transitions
- Iterative update rule

$$
\begin{aligned}
& Q_{k+1}(s_k, a_k) \\
= \quad & Q_k(s_k, a_k) + \alpha_k \left(\underbrace{r_{k+1} + \gamma \max_{a'} Q_k\left(s_{k+1}, a'\right)}_{\text{Estimate of } Q^*(s_k, a_k)} - Q_k(s_k, a_k) \right)
\end{aligned}
$$

$\alpha_k \in (0, 1]$ is the learning rate

Temporal difference: Difference between estimate of $Q^*(s_k, a_k)$ and current estimate $Q_k(s_k, a_k)$

Q-learning update rule converges to Q^* as $k \to \infty$ if

(1) $\sum_{k=0}^{\infty} \alpha_k^2 < \infty$ and $\sum_{k=0}^{\infty} \alpha_k \to \infty$

(2) All (s, a) pairs are (asymptotically) visited infinitely often

- (1) can be met by setting $\alpha_k = \frac{1}{k}$ with finite α_0 (e.g., 1)
- (2) can be satisfied if agent has non-zero probability of selecting any available action in every state it visits. This is called exploration

Proof can be found in: Watkins, C. J. C. H. and Dayan, P., Q-learning, Machine Learning, 8:279-292, 1992

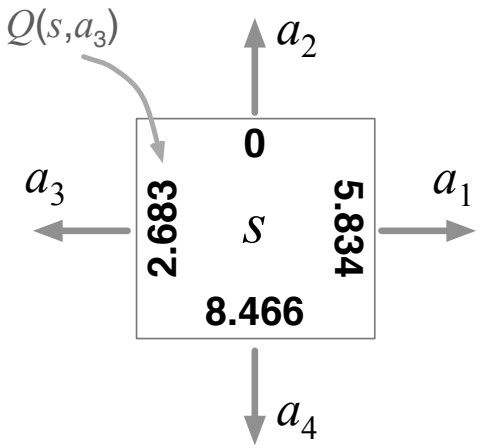
$$Q_{k+1}(s_k, a_k) = Q_k(s_k, a_k) + \alpha_k \left(r_{k+1} + \gamma \max_{a'} Q_k\left(s_{k+1}, a'\right) - Q_k(s_k, a_k) \right)$$

Exploitation: Use greedy policy to select currently known best action

$$a_{k+1} = \max_{a'} Q_k\left(s_{k+1}, a'\right)$$

Exploration: Try action other than currently known best action

$$a_{k+1} \neq \max_{a'} Q_k\left(s_{k+1}, a'\right)$$



Exploitation: Take a_4

Exploration: Take a_1, a_2, or a_3

- Exploration is essential for learning
- If we do not try new things, then we will never develop new capabilities and skills
- However, if we always explore and do not utilize what we have learned, then we will most likely not be able to achieve anything significant

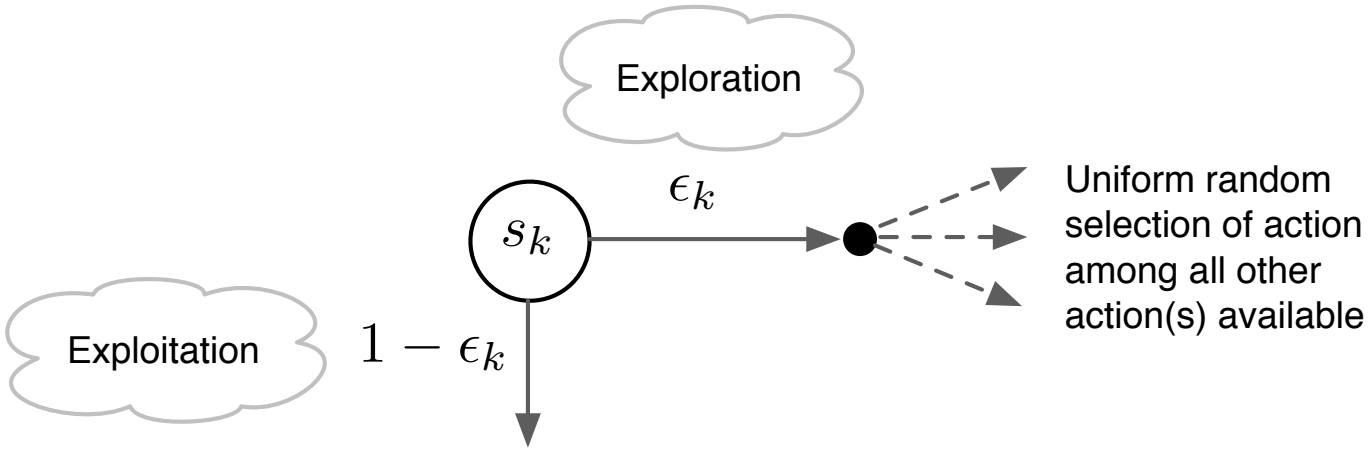
Sometimes too much exploration can be dangerous

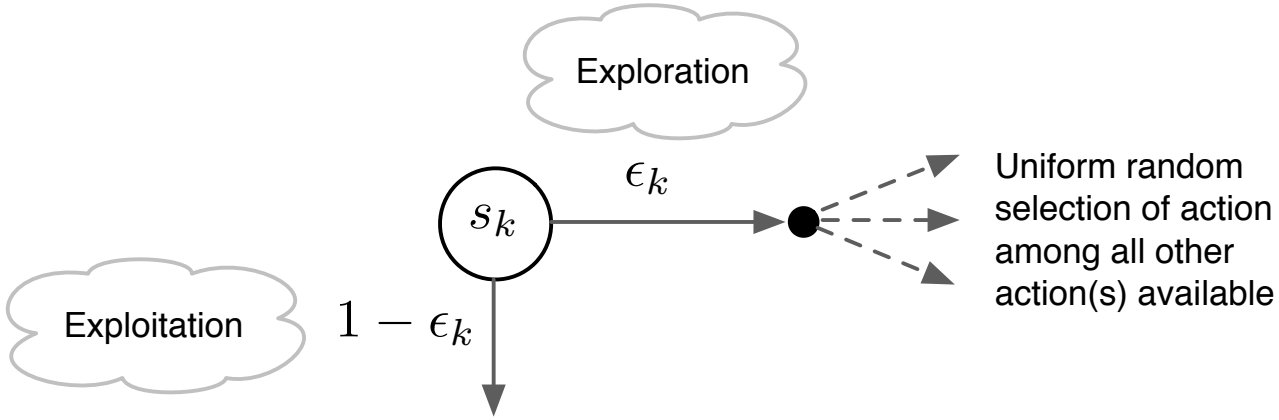
There must be balance between exploration and exploitation

Photo from http://www.artlebedev.com/everything/vilcus/

Balancing exploration-exploitation trade-off: ϵ-greedy exploration

$$a_k = \begin{cases} a \in \arg\max_{\hat{a}} Q_k(s_k, \hat{a}) & \text{with probability } 1-\epsilon_k \\ \text{action uniformly randomly selected from all other actions available at state } s_k & \text{with probability } \epsilon_k \end{cases}$$





- ϵ is kept small so that agent will focus on task at hand most of the time, and will explore only occasionally
- Reduce exploration as learning continues by setting $\epsilon_k = \frac{1}{k}$

Input: Discount factor γ; exploration probability ϵ_k; learning rate α_k

- Initialize Q-function, e.g., $Q_0 \leftarrow 0$
- Determine the initial state s_0
- For time step k, select action a_k according to:

$$a_k = \begin{cases} a \in \arg\max_{\hat{a}} Q_k(s_k, \hat{a}) & \text{with probability } 1-\epsilon_k \\ \text{an action uniformly randomly selected from all other actions available at state } s_k & \text{with probability } \epsilon_k \end{cases}$$

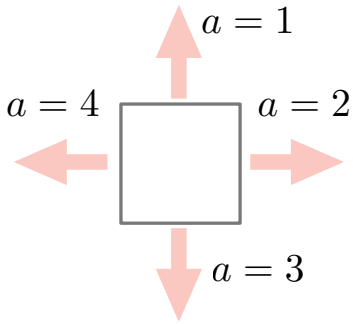
- Apply action a_k, receive reward r_{k+1}, then observe next state s_{k+1}
- Update Q-function with:

$$Q_{k+1}(s_k, a_k) = Q_k(s_k, a_k) + \alpha_k \left(r_{k+1} + \gamma \max_{a'} Q_k\left(s_{k+1}, a'\right) - Q_k(s_k, a_k) \right)$$

- Set $k = k+1$ and repeat for-loop for the next time step

Note: This basic algorithm may be (very) slow to converge. Some modified forms of ε_k and α are often used in practice to speed up the learning process.

0	1	2
3	4	5
6	7	8



$$\rho(s, a, s') = \begin{cases} 1 & \text{if } s \neq 8 \text{ and } s' = 8 \\ 0 & \text{otherwise} \end{cases}$$

$$\gamma = 0.9\,, \quad a_k = 1/k\,, \quad a_0 = 1$$

- Learning consists of series of trials
- Starting at initial state robot makes transitions according to ϵ-greedy exploration and stops when reaching $s = 8$
- Will not actually simulate probabilistic selection of action in this example
- Will simply assume sequence of actions for a trial (as if they were observed to have occurred) so as to carry out calculation of values for Q-function

- For record keeping, define function $N(s, a)$ which counts accumulative number of times (over a set of trials) a has been taken at s
- Two types of diagrams
  - One showing values of $N(s, a)$ at end of a trial
  - The other showing values of Q-function at end of a trial
- Initial values of $N(s, a)$ and Q-function:

0 0 0 0 0	0 0 1 0 0	0 0 2 0 0
0 0 3 0 0	0 0 4 0 0	0 0 5 0 0
0 0 6 0 0	0 0 7 0 0	0 0 8 0 0

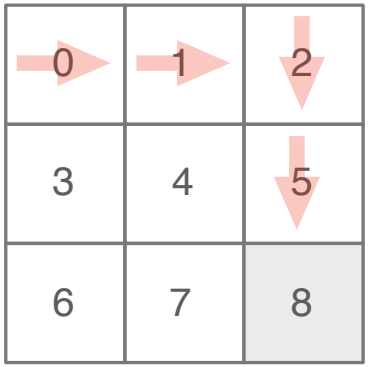
$N(s, a)$

0 0 0 0 0	0 0 1 0 0	0 0 2 0 0
0 0 3 0 0	0 0 4 0 0	0 0 5 0 0
0 0 6 0 0	0 0 7 0 0	0 0 8 0 0

Values of Q-function

Trial 1:

- Assume action sequence as shown in diagram
- $\alpha_k = 1/k,\ \alpha_0 = 1$
- $\gamma = 0.9$



Sample calculations:

$$
\begin{aligned}
Q_1(0,2) &= Q_0(0,2) + \alpha_0\Big(\rho(0,2,1) + \gamma \max_{a'} Q_0\left(1,a'\right) - Q_0(0,2)\Big) \\
&= 0 + 1\cdot(0 + 0.9\cdot 0 - 0) \\
&= 0 \\
&\cdots \\
Q_4(5,3) &= Q_3(5,3) + \alpha_3\Big(\rho(5,3,8) + \gamma \max_{a'} Q_3\left(8,a'\right) - Q_3(5,3)\Big) \\
&= 0 + \frac{1}{3}\Big(1 + 0.9\cdot 0 - 0\Big) \\
&= 0.333
\end{aligned}
$$

Detailed calculations are shown in supplementary notes

Values of $N(s,a)$ and Q-function at end of Trial 1

0 0 0	0 1 0	0 2 0 **1**
0 3 0	0 4 0	0 5 0 **1**
0 6 0	0 7 0	0 8 0

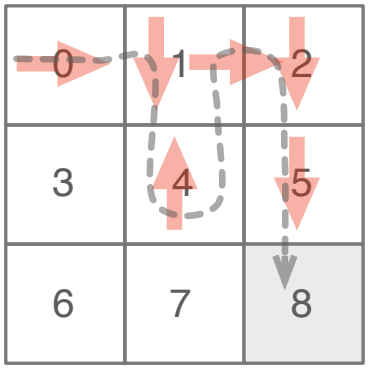
(a) $N(s,a)$

0 0 0	0 1 0	0 2 0
0 3 0	0 4 0	0 5 0 **0.333**
0 6 0	0 7 0	0 8 0

(b) Q-function

Trial 2:

- Assume action sequence as shown in diagram
- $\alpha_0 = 1$
- $\gamma = 0.9$



Sample calculations:

$$
\begin{aligned}
Q_1(0,2) &= Q_0(0,2) + \alpha_0\left(\rho(0,2,1) + \gamma \max_{a'} Q_0\left(1,a'\right) - Q_0(0,2)\right) \\
&= 0 + 1 \cdot (0 + 0.9 \cdot 0 - 0) \\
&= 0 \\
&\cdots \\
&\cdots
\end{aligned}
$$

Detailed calculations are shown in supplementary notes

Values of $N(s,a)$ and Q-function at end of Trial 2

0 0 0	0 1 **2**	0 2 0
0 3 0	**1** 0 4 0	0 5 0 **2**
0 6 0	0 7 0	0 8 0

(a) $N(s,a)$

0 0 0	0 1 0	0 2 0 **0.075**
0 3 0	0 4 0	0 5 0 **0.466**
0 6 0	0 7 0	0 8 0

(b) Q-function

Iterative process repeats until values of Q-function converge to optimal values, upon which an optimal policy can be obtained

Q-learning: Summary