Introduction to AI

Lecture 3:
Traditional Classification-

by Hong-Han Shuai

National Yang Ming Chiao Tung University

Syllabus

Week	Date	Contents
1	9/2	Lecture 1: Class Overview and Unsupervised Learning (HW#1)
2	9/9	Lecture 2: Traditional Classification-Part 1
3	9/16	Lecture 3: Traditional Classification-Part 2
4	9/23	Lecture 4: Neural Networks Basics (HW#2)
5	9/30	Hands-on Tutorials on PyTorch
6	10/7	Lecture 5: Deep Learning in Practice
7	10/14	Lecture 6: Introduction to Natural Language Processing
8	10/21	Midterm

Classification

Classification – Basic Concepts

Decision Tree Induction

Bayes Classification Methods

Rule-Based Classification

Techniques to Improve Classification Accuracy: Ensemble Methods

Lazy Learner

Support Vector Machine

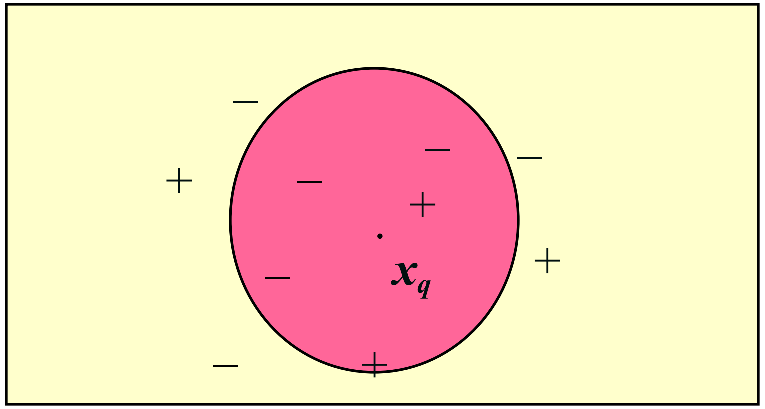
Evaluations

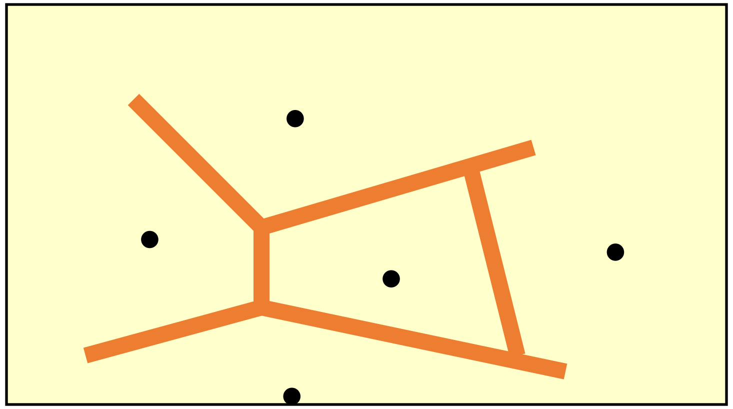
Lazy vs. Eager Learning

- Lazy vs. eager learning
  - **Lazy learning** (e.g., instance-based learning): Simply stores training data (or only minor processing) and waits until it is given a test tuple
  - **Eager learning** (the above discussed methods): Given a set of training tuples, constructs a classification model before receiving new (e.g., test) data to classify
- Lazy: less time in training but more time in predicting
- Accuracy
  - Lazy method effectively uses a richer hypothesis space since it uses many local linear functions to form an implicit global approximation to the target function
  - Eager: must commit to a single hypothesis that covers the entire instance space

The k-Nearest Neighbor Algorithm

- All instances correspond to points in the n-D space
- The nearest neighbor are defined in terms of Euclidean distance, dist($\mathbf{X_1}$, $\mathbf{X_2}$)
- Target function could be discrete- or real- valued
- For discrete-valued, k-NN returns the most common value among the k training examples nearest to x_q
- Vonoroi diagram: the decision surface induced by 1-NN for a typical set of training examples





Lazy learning example?

The Slido app must be installed on every computer you're presenting from

slido

Do not edit
How to change the design

Discussion on the *k*-NN Algorithm

- *k*-NN for real-valued prediction for a given unknown tuple
  - Returns the mean values of the *k* nearest neighbors
- Distance-weighted nearest neighbor algorithm
  - Weight the contribution of each of the *k* neighbors according to their distance to the query x_q
    - Give greater weight to closer neighbors

$$w \equiv \frac{1}{d(x_q, x_i)^2}$$

- Robust to noisy data by averaging *k*-nearest neighbors
- Curse of dimensionality: distance between neighbors could be dominated by irrelevant attributes
  - To overcome it, axes stretch or elimination of the least relevant attributes

Classification

Classification – Basic Concepts

Decision Tree Induction

Bayes Classification Methods

Rule-Based Classification

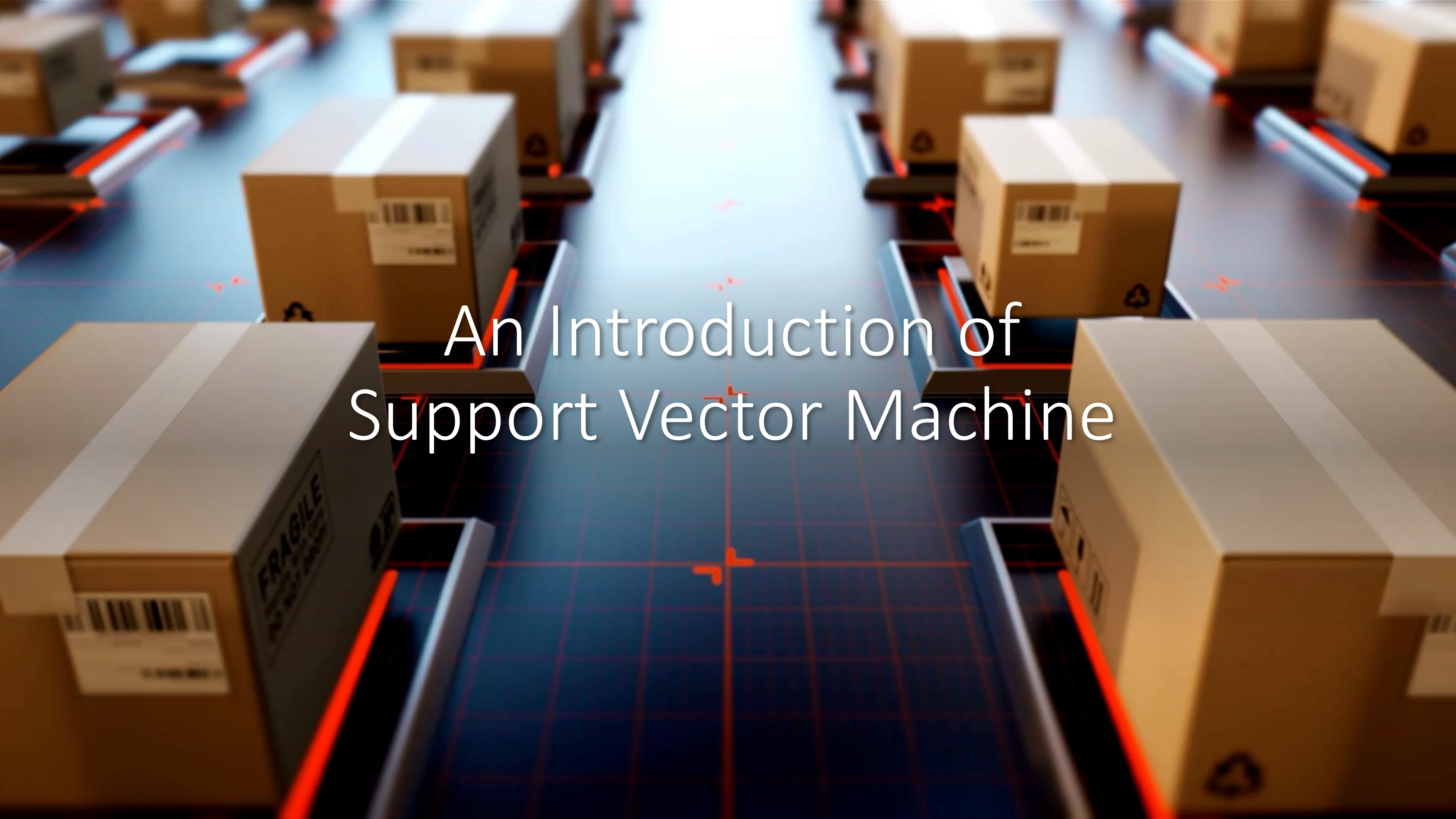
Techniques to Improve Classification Accuracy: Ensemble Methods

Lazy Learner

Support Vector Machine

Evaluations

An Introduction of Support Vector Machine



Today: Support Vector Machine (SVM)

- A classifier derived from statistical learning theory by Vapnik, et al. in 1992
- SVM became famous when, using images as input, it gave accuracy comparable to neural-network with hand-designed features in a handwriting recognition task
- Currently, SVM is widely used in object detection & recognition, content-based image retrieval, text recognition, biometrics, speech recognition, etc.
- Also used for regression (will not cover today)
- Chapter 5.1, 5.2, 5.3, 5.11 (5.4*) in textbook

V. Vapnik

Thanks for the slides from Prof. Jinwei Gu.

Outline

- Linear Discriminant Function
- Large Margin Linear Classifier
- Nonlinear SVM: The Kernel Trick
- Demo of SVM

Discriminant Function

- Chapter 2.4: the classifier is said to assign a feature vector x to class w_i if

$$g_i(\mathbf{x}) > g_j(\mathbf{x}) \qquad \text{for all} \quad j \neq i$$

- For two-category case, $g(\mathbf{x}) \equiv g_1(\mathbf{x}) - g_2(\mathbf{x})$

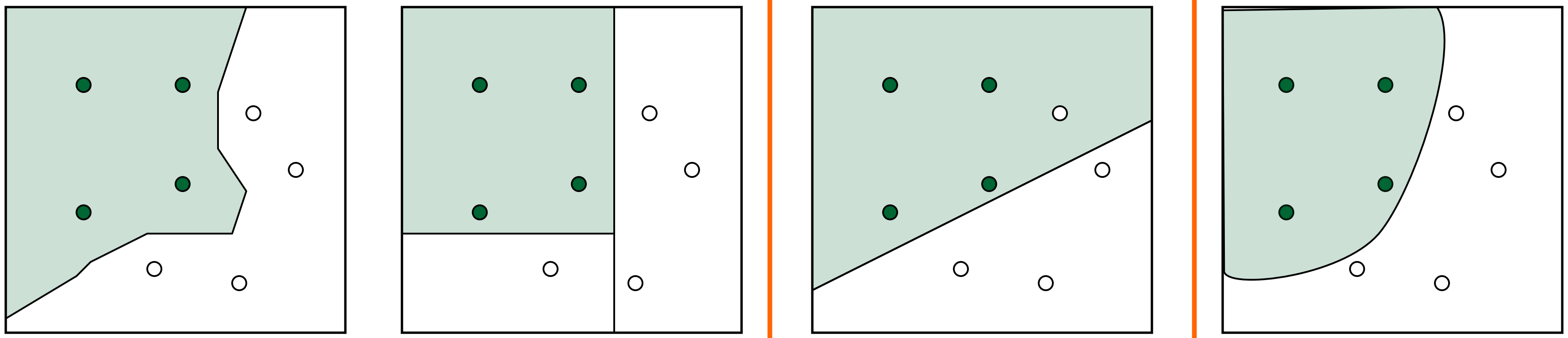
$$\text{Decide } \omega_1 \text{ if } g(\mathbf{x}) > 0; \text{ otherwise decide } \omega_2$$

- An example we've learned before:
  - Minimum-Error-Rate Classifier

$$g(\mathbf{x}) \equiv p(\omega_1 \mid \mathbf{x}) - p(\omega_2 \mid \mathbf{x})$$

Discriminant Function

- It can be arbitrary functions of *x*, such as:



Nearest Neighbor | Decision Tree | Linear Functions | Nonlinear Functions

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

Linear Discriminant Function

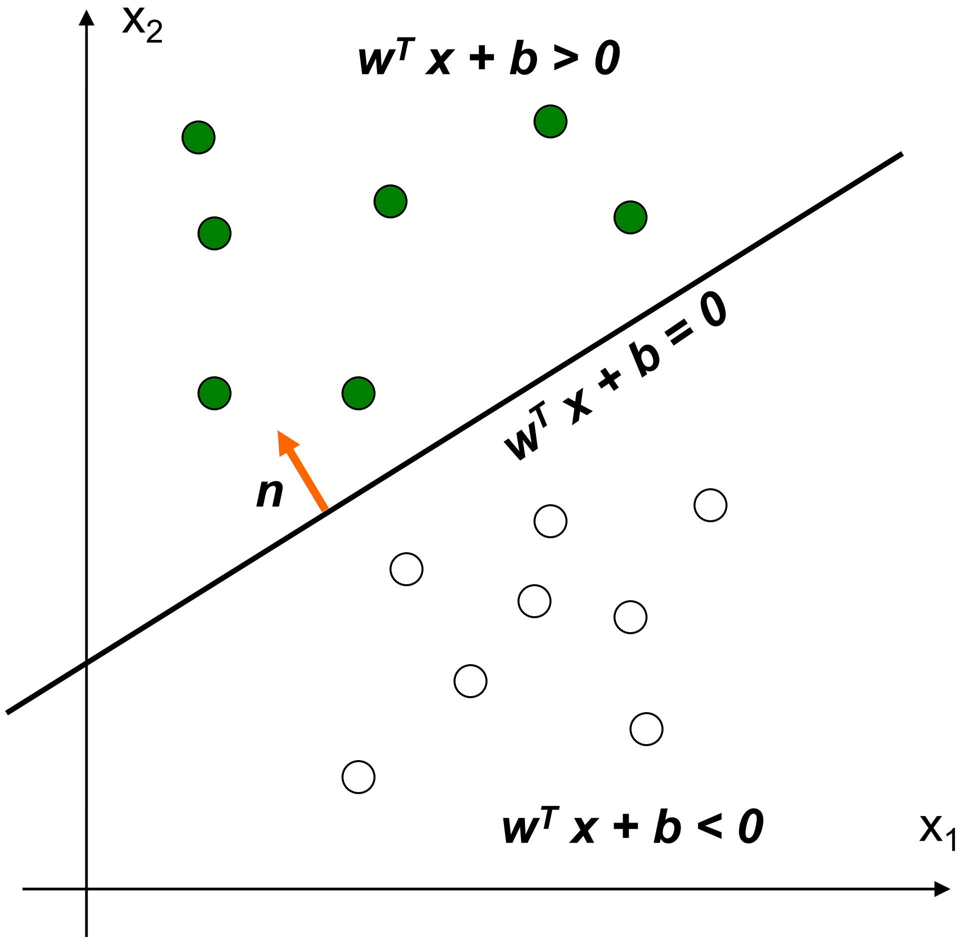
- g(x) is a linear function:

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

- A hyper-plane in the feature space

- (Unit-length) normal vector of the hyper-plane:

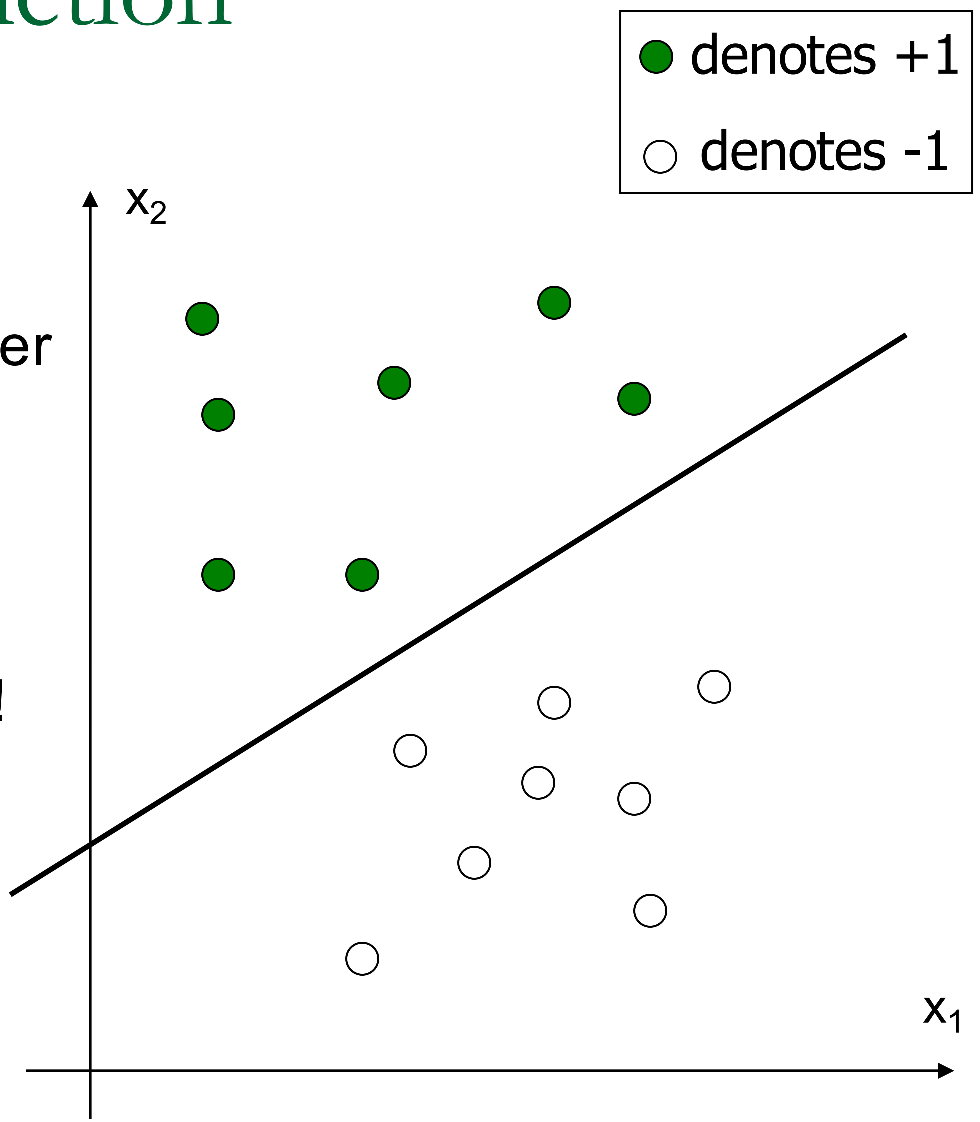
$$\mathbf{n} = \frac{\mathbf{w}}{\|\mathbf{w}\|}$$



Linear Discriminant Function

- How would you classify these points using a linear discriminant function in order to minimize the error rate?

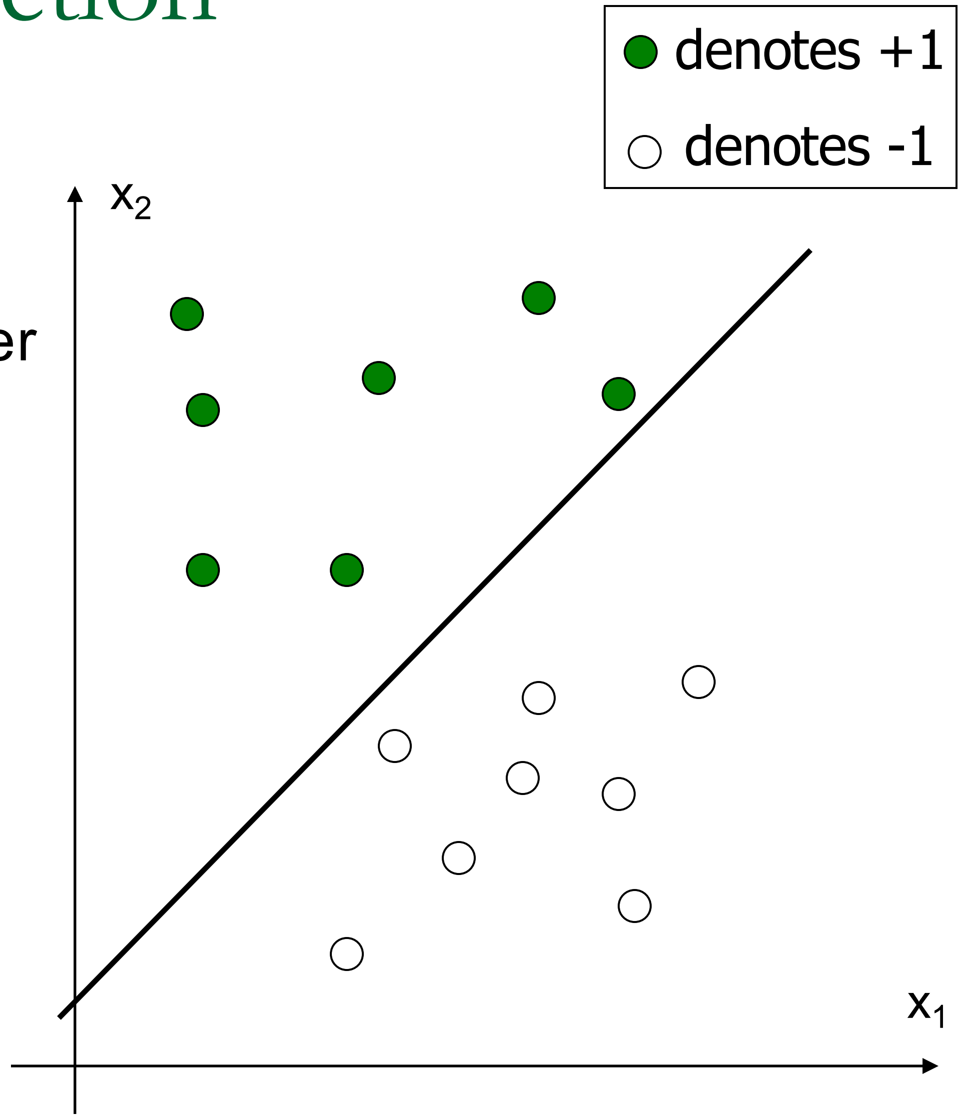
- Infinite number of answers!



Linear Discriminant Function

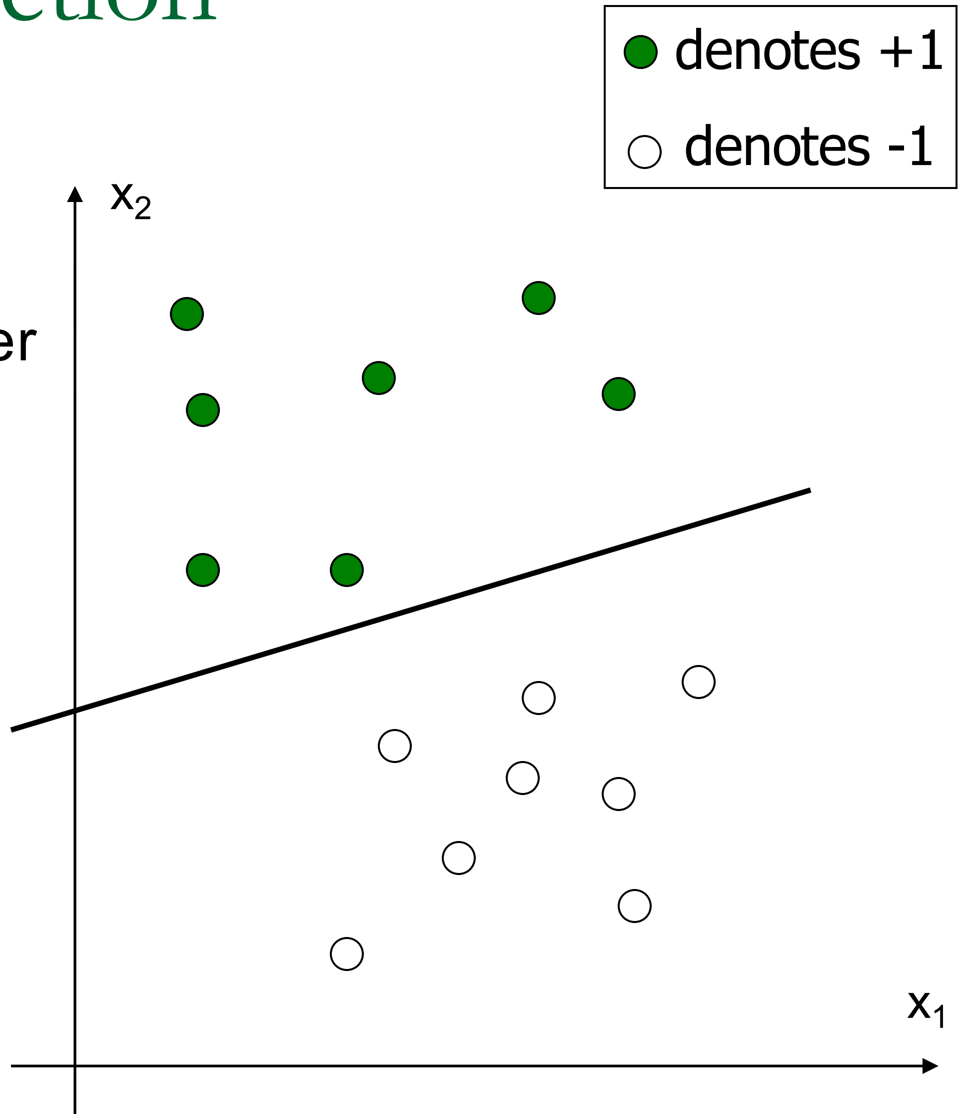
- How would you classify these points using a linear discriminant function in order to minimize the error rate?

- Infinite number of answers!



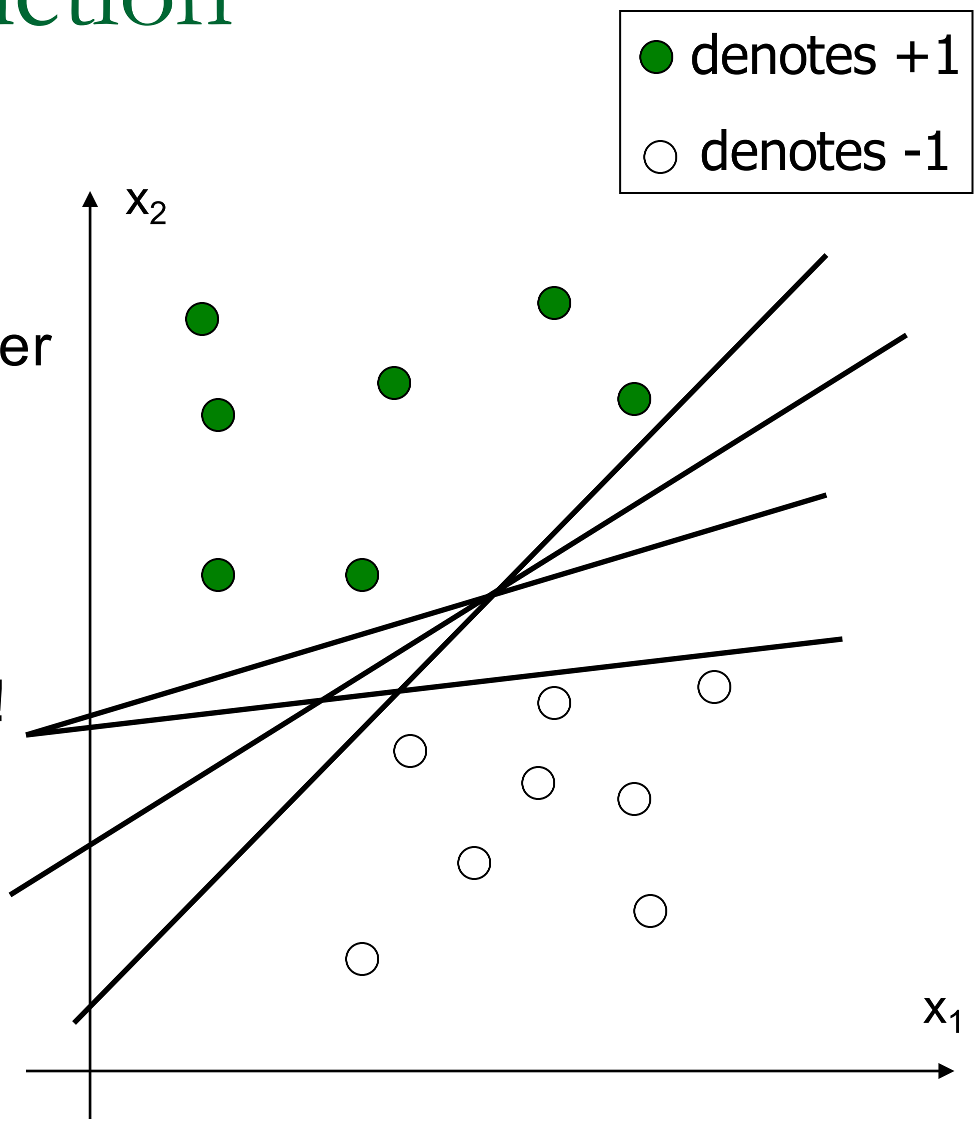
Linear Discriminant Function

- How would you classify these points using a linear discriminant function in order to minimize the error rate?
- Infinite number of answers!



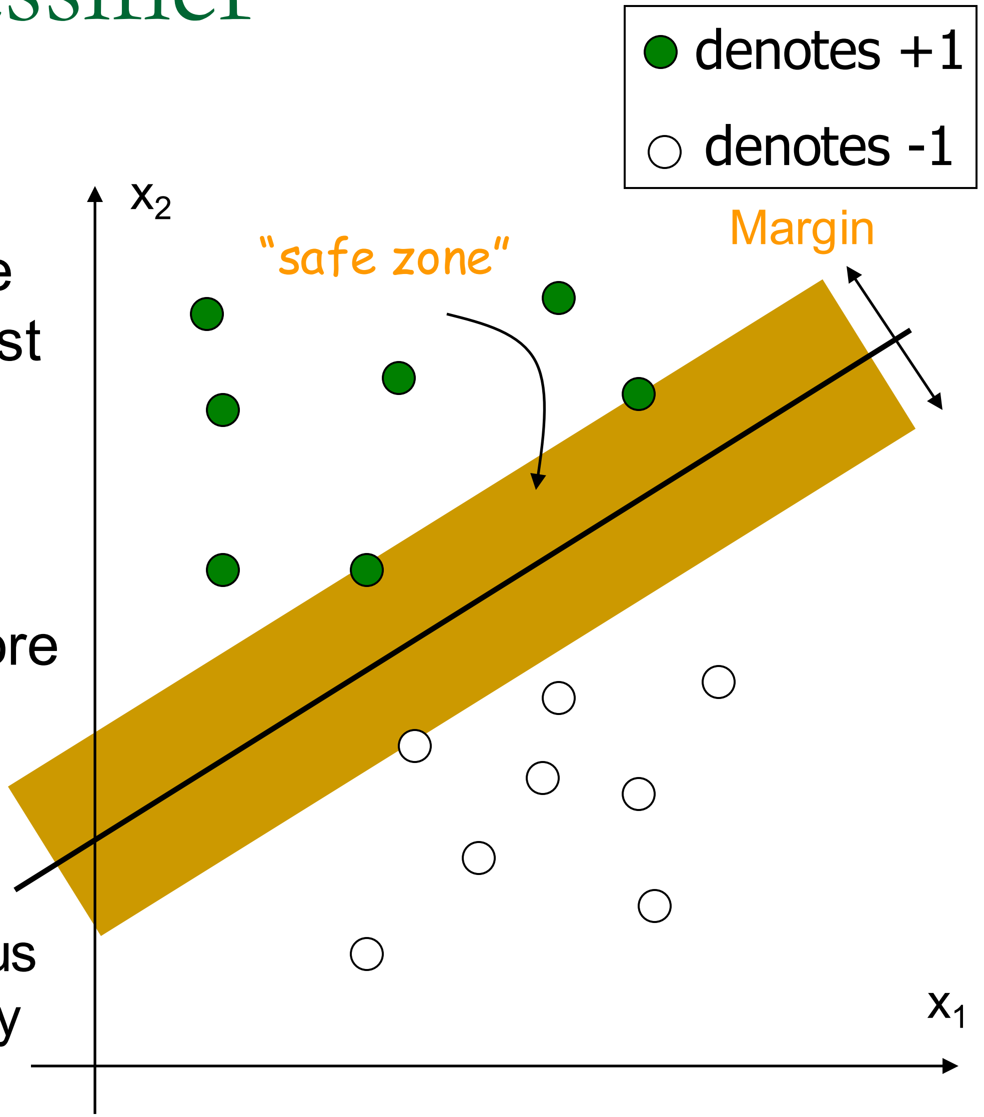
Linear Discriminant Function

- How would you classify these points using a linear discriminant function in order to minimize the error rate?
- Infinite number of answers!
- Which one is the best?



Large Margin Linear Classifier

- The linear discriminant function (classifier) with the maximum margin is the best
- Margin is defined as the width that the boundary could be increased by before hitting a data point
- Why it is the best?
  - Robust to outliners and thus strong generalization ability



Large Margin Linear Classifier

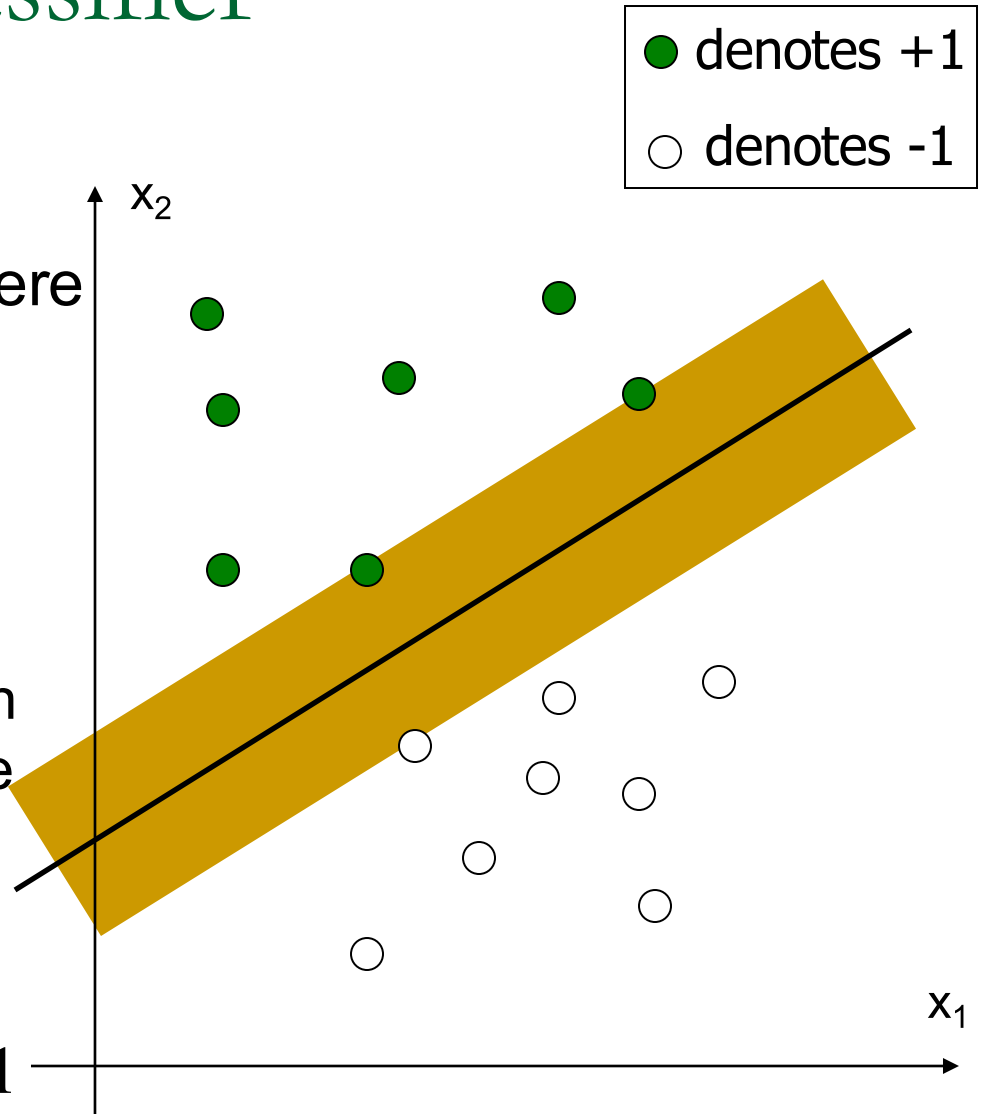
- Given a set of data points:

  $\{(\mathbf{x}_i, y_i)\},\ i = 1, 2, \ldots, n$, where

$$\text{For } y_i = +1, \quad \mathbf{w}^T \mathbf{x}_i + b > 0$$
$$\text{For } y_i = -1, \quad \mathbf{w}^T \mathbf{x}_i + b < 0$$

- With a scale transformation on both *w* and *b*, the above is equivalent to

$$\text{For } y_i = +1, \quad \mathbf{w}^T \mathbf{x}_i + b \geq 1$$
$$\text{For } y_i = -1, \quad \mathbf{w}^T \mathbf{x}_i + b \leq -1$$



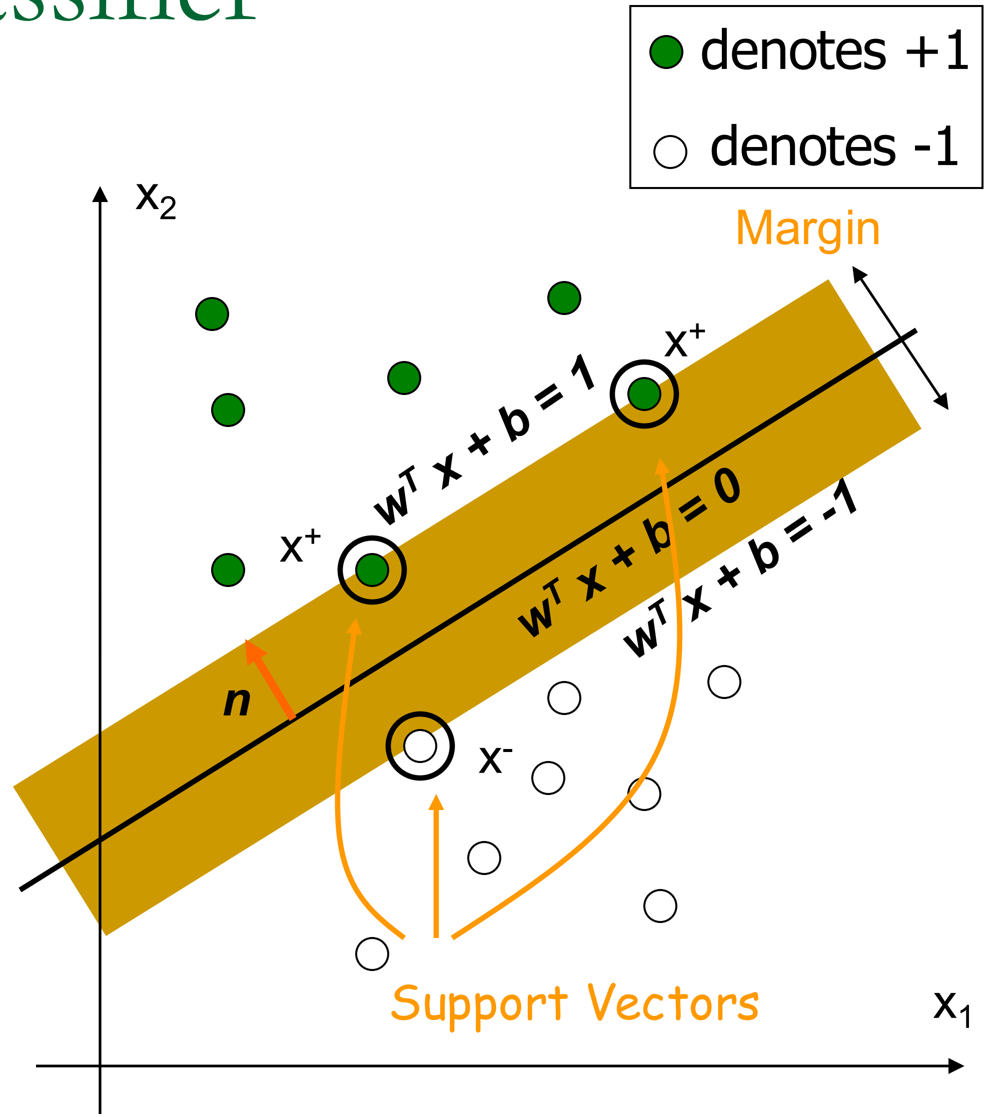
Large Margin Linear Classifier

- We know that

$$\mathbf{w}^T\mathbf{x}^+ + b = 1$$
$$\mathbf{w}^T\mathbf{x}^- + b = -1$$

- The margin width is:

$$M = (\mathbf{x}^+ - \mathbf{x}^-)\cdot\mathbf{n}$$
$$= (\mathbf{x}^+ - \mathbf{x}^-)\cdot\frac{\mathbf{w}}{\|\mathbf{w}\|} = \frac{2}{\|\mathbf{w}\|}$$



Large Margin Linear Classifier

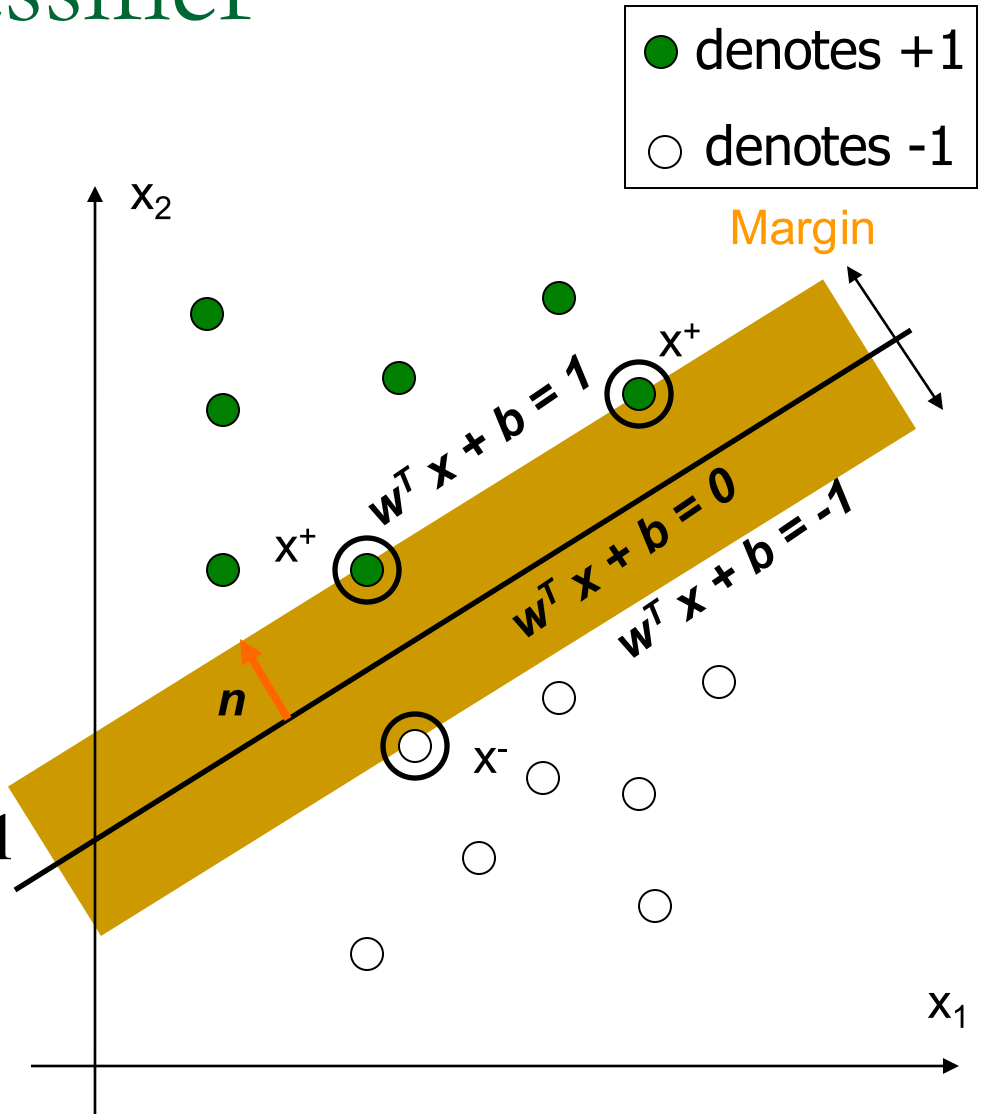
- Formulation:

$$\text{maximize} \quad \frac{2}{\|\mathbf{w}\|}$$

such that

$$\text{For } y_i = +1, \quad \mathbf{w}^T \mathbf{x}_i + b \geq 1$$

$$\text{For } y_i = -1, \quad \mathbf{w}^T \mathbf{x}_i + b \leq -1$$



Large Margin Linear Classifier

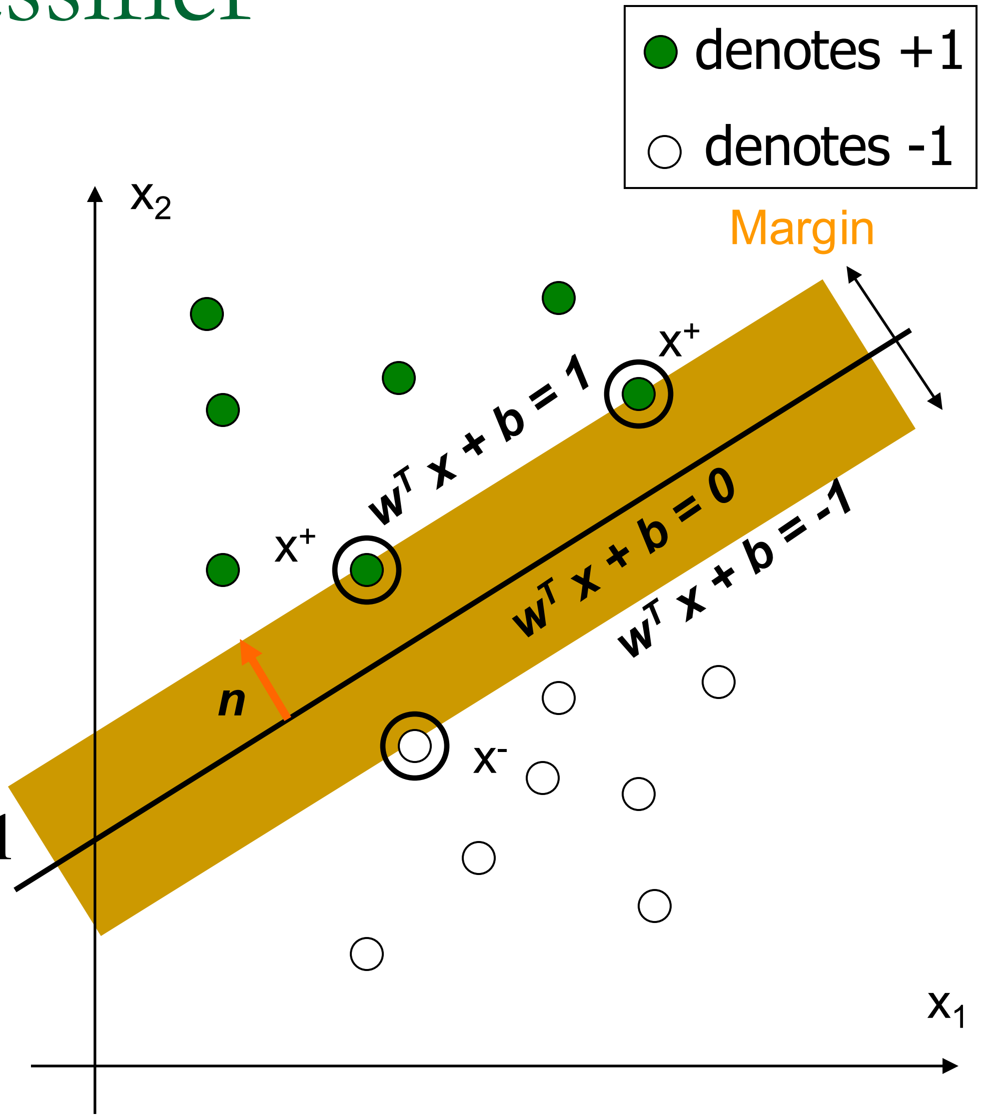
- Formulation:

$$\text{minimize} \quad \frac{1}{2}\|\mathbf{w}\|^2$$

such that

$$\text{For } y_i = +1, \quad \mathbf{w}^T\mathbf{x}_i + b \geq 1$$

$$\text{For } y_i = -1, \quad \mathbf{w}^T\mathbf{x}_i + b \leq -1$$



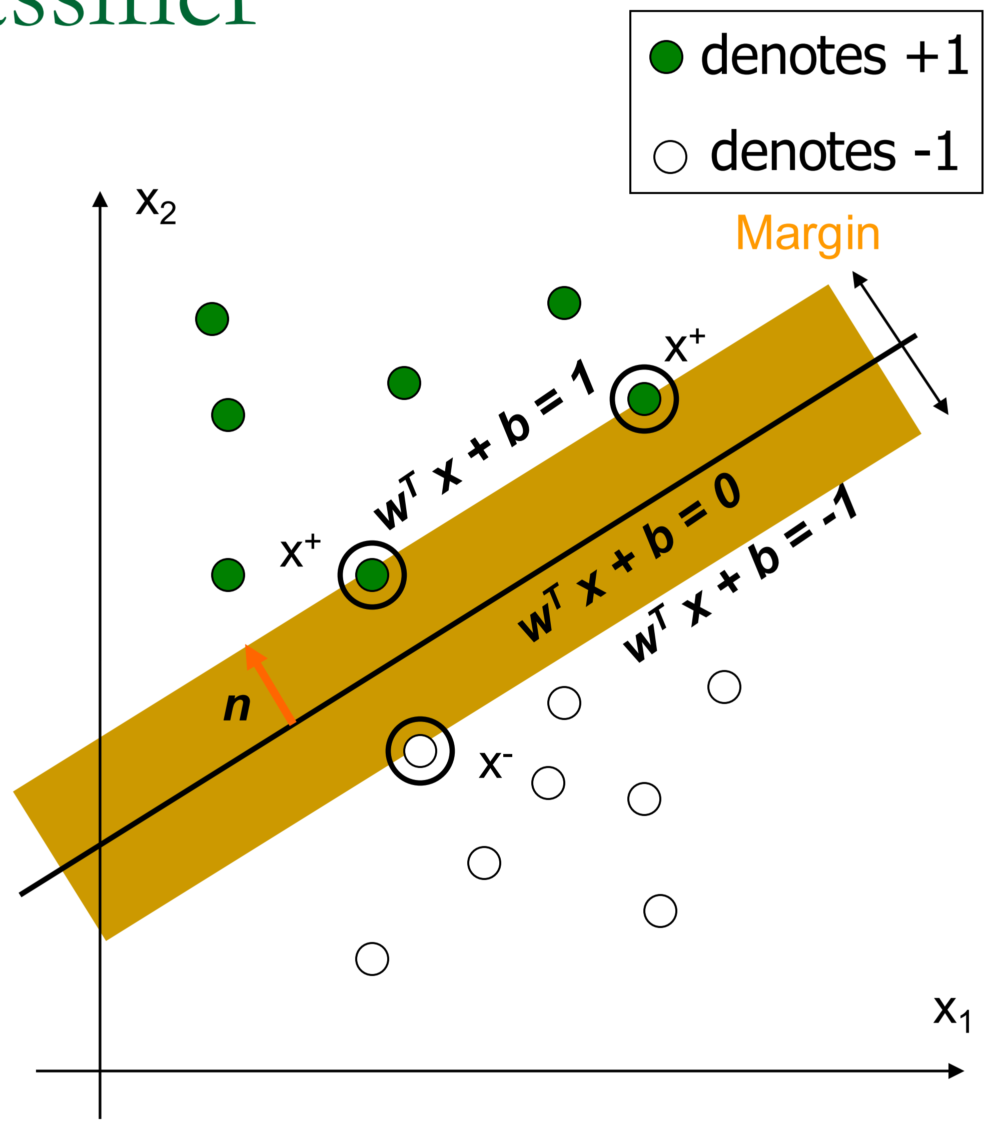
Large Margin Linear Classifier

- Formulation:

$$\text{minimize} \quad \frac{1}{2}\|\mathbf{w}\|^2$$

such that

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1$$



Solving the Optimization Problem

Quadratic programming with linear constraints

$$\text{minimize} \quad \frac{1}{2}\|\mathbf{w}\|^2$$

$$\text{s.t.} \quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1$$

Lagrangian Function

$$\text{minimize} \quad L_p(\mathbf{w}, b, \alpha_i) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i \left(y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 \right)$$

$$\text{s.t.} \quad \alpha_i \geq 0$$

Solving the Optimization Problem

$$\text{minimize} \quad L_p(\mathbf{w}, b, \alpha_i) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i \left(y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 \right)$$

$$\text{s.t.} \quad \alpha_i \geq 0$$

$$\frac{\partial L_p}{\partial \mathbf{w}} = 0 \quad \Rightarrow \quad \mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$$

$$\frac{\partial L_p}{\partial b} = 0 \quad \Rightarrow \quad \sum_{i=1}^{n} \alpha_i y_i = 0$$

Solving the Optimization Problem

$$\text{minimize } \; L_p(\mathbf{w}, b, \alpha_i) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i \left(y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 \right)$$

$$\text{s.t.} \quad \alpha_i \geq 0$$

Lagrangian Dual Problem

$$\text{maximize } \; \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{s.t.} \quad \alpha_i \geq 0 \text{ , and } \sum_{i=1}^{n} \alpha_i y_i = 0$$

Next, substitute $\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$ into the Lagrangian:

The term $\frac{1}{2}\|\mathbf{w}\|^2$ becomes:

$$\frac{1}{2}\|\mathbf{w}\|^2 = \frac{1}{2}\left(\sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i\right)^T \left(\sum_{j=1}^{n} \alpha_j y_j \mathbf{x}_j\right) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

The remaining terms simplify as follows:

$$-\sum_{i=1}^{n} \alpha_i \left[y_i(\mathbf{w}^T \mathbf{x}_i + b) - (\quad 1 \quad)\right] = \sum_{i=1}^{n} \alpha_i$$

because $y_i(\mathbf{w}^T \mathbf{x}_i + b) = 1$ for support vectors.

1. 原問題與拉格朗日

$$\min_{\mathbf{w},b} \ \tfrac{1}{2}\|\mathbf{w}\|^2 \quad \text{s.t.} \quad y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1,\ i = 1,\dots,n$$

對每個不等式配 $\alpha_i \geq 0$：

$$\mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha}) = \tfrac{1}{2}\|\mathbf{w}\|^2 + \sum_{i=1}^{n} \alpha_i \big(1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)\big).$$

把項目展開、整理係數：

$$\mathcal{L} = \tfrac{1}{2}\mathbf{w}^\top \mathbf{w} - \Big(\sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i\Big)^\top \mathbf{w} - b\sum_{i=1}^{n} \alpha_i y_i + \sum_{i=1}^{n} \alpha_i.$$

令 $\mathbf{v} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$。

則

$$\mathcal{L} = \tfrac{1}{2}\mathbf{w}^\top \mathbf{w} - \mathbf{v}^\top \mathbf{w} - b\sum_{i=1}^{n} \alpha_i y_i + \sum_{i=1}^{n} \alpha_i.$$

2. 對 $\mathbf{w}$ 與 b 取極小

一階條件：

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \mathbf{w} - \mathbf{v} = \mathbf{0} \;\Rightarrow\; \mathbf{w}^\star = \mathbf{v} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i,$$

$$\frac{\partial \mathcal{L}}{\partial b} = -\sum_{i=1}^{n} \alpha_i y_i = 0 \;\Rightarrow\; \sum_{i=1}^{n} \alpha_i y_i = 0.$$

對 $\mathbf{w}$ 的最小值可以用「完全平方」看出來：

$$\tfrac{1}{2}\mathbf{w}^\top \mathbf{w} - \mathbf{v}^\top \mathbf{w} = \tfrac{1}{2}\|\mathbf{w} - \mathbf{v}\|^2 - \tfrac{1}{2}\|\mathbf{v}\|^2.$$

極小發生在 $\mathbf{w} = \mathbf{v}$，其值為 $-\tfrac{1}{2}\|\mathbf{v}\|^2$。

3. 代回得到對偶函數 $g(\boldsymbol{\alpha})$

把 $\mathbf{w}^\star$ 與條件 $\sum_i \alpha_i y_i = 0$ 代回 $\mathcal{L}$：

$$g(\boldsymbol{\alpha}) = \inf_{\mathbf{w},b} \mathcal{L} = \Big(-\tfrac{1}{2}\|\mathbf{v}\|^2\Big) - b\underbrace{\sum_{i=1}^{n} \alpha_i y_i}_{=0} + \sum_{i=1}^{n} \alpha_i$$

$$= \sum_{i=1}^{n} \alpha_i - \tfrac{1}{2}\|\mathbf{v}\|^2.$$

把 $\|\mathbf{v}\|^2$ 展開成雙和：

$$\|\mathbf{v}\|^2 = \Big\|\sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i\Big\|^2 = \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \, \mathbf{x}_i^\top \mathbf{x}_j.$$

因此

$$\boxed{g(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \tfrac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \, \mathbf{x}_i^\top \mathbf{x}_j}$$

[Supplements]

1. Primal 問題的形式

以 hard-margin SVM 為例，原始問題是：

$$\min_{w,b} \frac{1}{2}\|w\|^2 \quad \text{s.t.} \quad y_i(w^T x_i + b) \geq 1,\ \forall i$$

這是一個 **帶有不等式約束的凸優化問題**。

2. Lagrangian 函數

我們引入 Lagrange 乘子 $\alpha_i \geq 0$，得到拉格朗日函數：

$$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_i \alpha_i \big(y_i(w^T x_i + b) - 1\big)$$

這是一個 **同時含有 primal 變數 (w, b) 與 dual 變數 (α)** 的函數。

3. min–max 結構的由來

- 原始問題是 **最小化** $\frac{1}{2}\|w\|^2$，同時要滿足 constraint。
- 在 Lagrangian 表達式裡，為了確保 constraint 成立，需要對 α 做一個 **最大化**（因為 $\alpha \geq 0$ 會懲罰違反 constraint 的情況）。

因此整體問題就變成：

$$\min_{w,b} \max_{\alpha \geq 0} L(w, b, \alpha)$$

4. 為什麼會寫成 max–min？

這跟 **弱對偶性 (weak duality)** 與 **凸性條件 (Slater's condition)** 有關：

- 一般定義 **dual problem** 時，會把次序反過來寫成

$$\max_{\alpha \geq 0} \min_{w,b} L(w, b, \alpha)$$

- 這是因為對於 **凸優化問題**，在滿足 Slater's condition 的情況下，min–max 與 max–min 會相等（即 **強對偶性 strong duality** 成立）。
- 所以圖中寫的

$$\max_{\alpha \geq 0} \min_{w,b} L(w, b, \alpha)$$

  就是 SVM 的 **對偶問題**。

Advantages of using the dual form

(a) 把限制處理掉

- 原始 primal 問題有很多 inequality constraints，直接求解很麻煩。
- Dual problem 只剩下 $\alpha_i \geq 0$ 的簡單條件。

(b) 維度更低

- Primal 裡的變數是 (w, b)，維度跟特徵數 d 有關。
- Dual 裡的變數是 α_i，維度跟樣本數 n 有關。
  - 當特徵維度 d 很大（甚至無窮大，kernel trick 情況），dual 問題反而更好算。

(c) Kernel trick

- 在 dual problem 裡，w 只出現於 $\langle x_i, x_j \rangle$，因此可以用 **kernel function** 替代內積。
- 這就是為什麼 SVM 可以用 kernel 把資料映射到高維空間，卻不用真的算高維向量。

(d) 支持向量出現

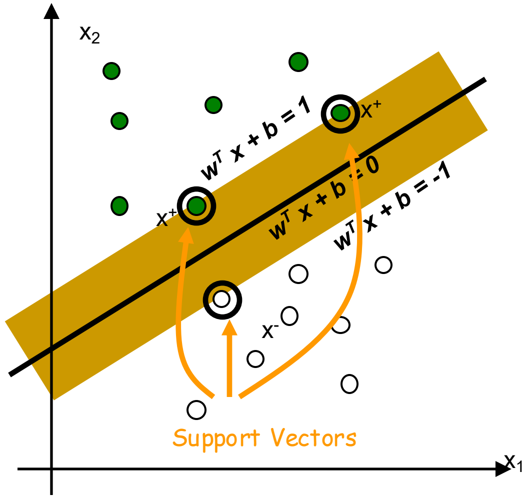
- Dual 的解只有少數 $\alpha_i > 0$，這些對應到的點就是 **support vectors**。
- 訓練後模型只需要記住這些支持向量，而不是全部資料。

Solving the Optimization Problem

- From KKT condition, we know:

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

- Thus, only support vectors have $\alpha_i \neq 0$



- The solution has the form:

$$\mathbf{w}=\sum_{i=1}^{n}\alpha_i y_i \mathbf{x}_i=\sum_{i\in \mathrm{SV}}\alpha_i y_i \mathbf{x}_i$$

get b from $y_i(\mathbf{w}^T\mathbf{x}_i+b)-1=0,$

where $\mathbf{x}_i$ is support vector

Solving the Optimization Problem

- The linear discriminant function is:

$$g(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b = \sum_{i \in \mathrm{SV}} \alpha_i \mathbf{x}_i^T \mathbf{x} + b$$

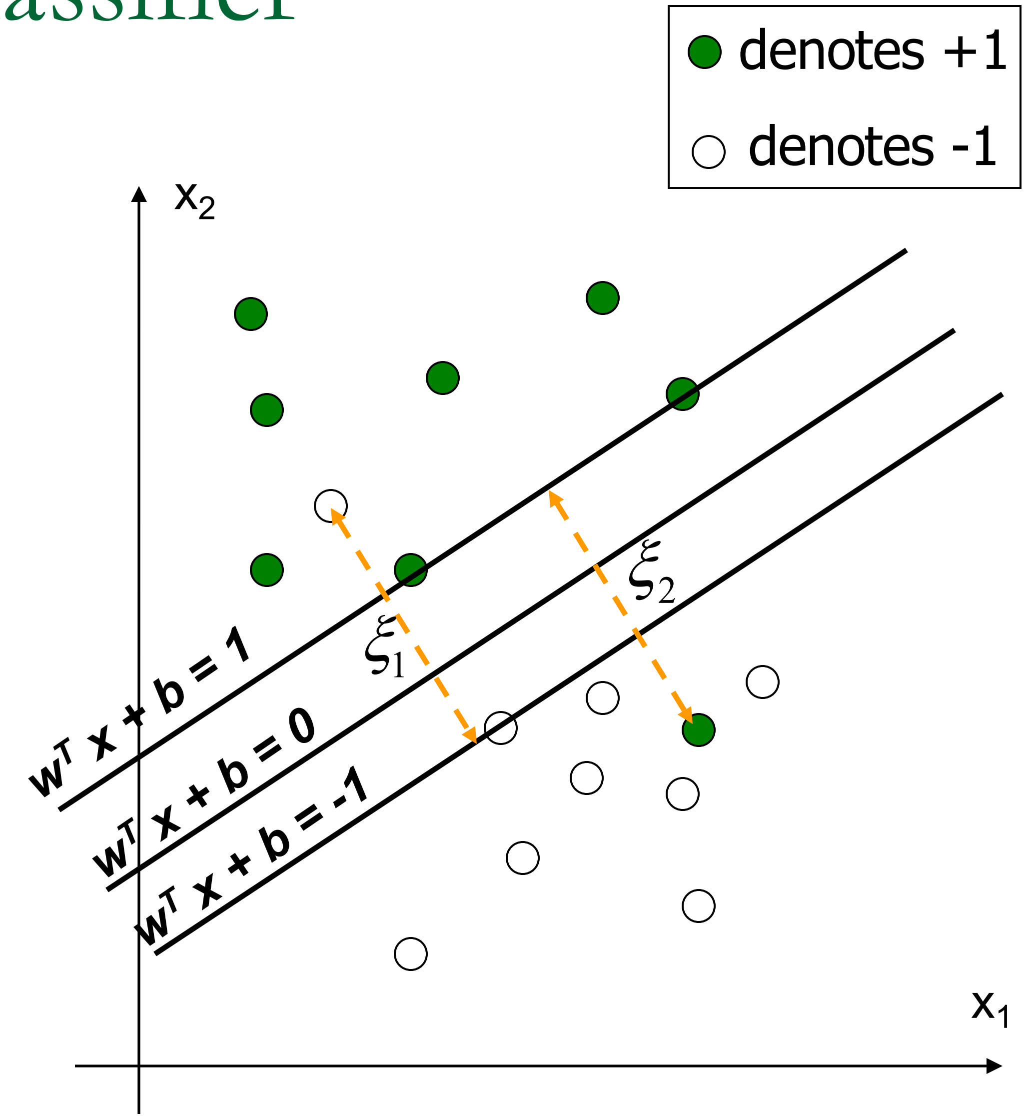
- Notice it relies on a *dot product* between the test point $\boldsymbol{x}$ and the support vectors $\boldsymbol{x}_i$

- Also keep in mind that solving the optimization problem involved computing the dot products $\boldsymbol{x}_i^T \boldsymbol{x}_j$ between all pairs of training points

Large Margin Linear Classifier

- What if data is not linear separable? (noisy data, outliers, etc.)

- Slack variables ξ_i can be added to allow mis-classification of difficult or noisy data points



Large Margin Linear Classifier

- Formulation:

$$\text{minimize} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i$$

such that

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

$$\xi_i \geq 0$$

- Parameter *C* can be viewed as a way to control over-fitting.

Large Margin Linear Classifier

- Formulation: (Lagrangian Dual Problem)

$$\text{maximize } \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

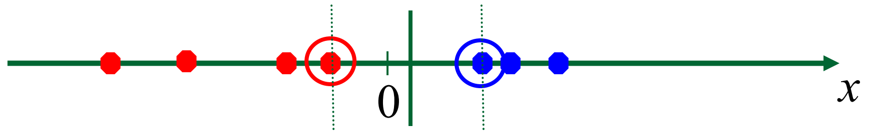
such that

$$0 \le \alpha_i \le C$$

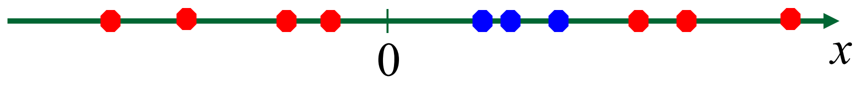
$$\sum_{i=1}^{n}\alpha_i y_i = 0$$

Non-linear SVMs

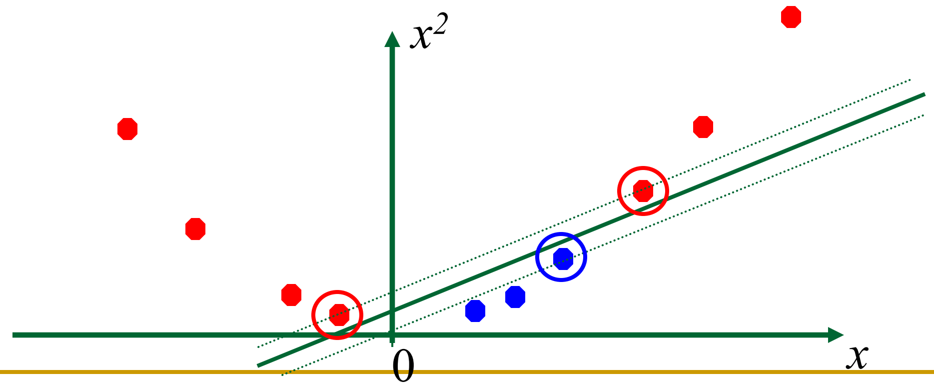
- Datasets that are linearly separable with noise work out great:



- But what are we going to do if the dataset is just too hard?



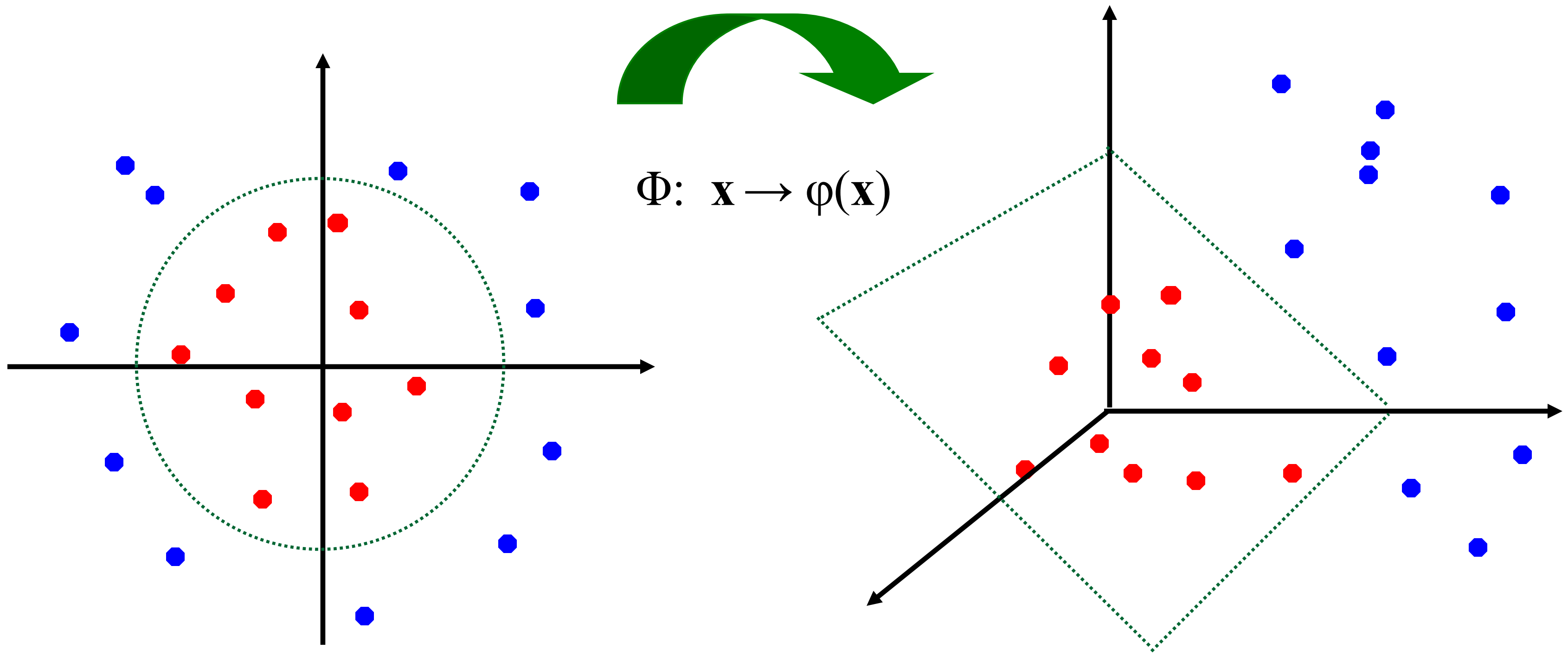
- How about… mapping data to a higher-dimensional space:



This slide is courtesy of *www.iro.umontreal.ca/~pift6080/documents/papers/**svm**_tutorial.**ppt***

Non-linear SVMs: Feature Space

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



This slide is courtesy of *www.iro.umontreal.ca/~pift6080/documents/papers/**svm**_tutorial.**ppt***

Nonlinear SVMs: The Kernel Trick

- With this mapping, our discriminant function is now:

$$g(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b = \sum_{i \in \mathrm{SV}} \alpha_i \boxed{\phi(\mathbf{x}_i)^T \phi(\mathbf{x})} + b$$

- No need to know this mapping explicitly, because we only use the dot product of feature vectors in both the training and test.

- A *kernel function* is defined as a function that corresponds to a dot product of two feature vectors in some expanded feature space:

$$K(\mathbf{x}_i, \mathbf{x}_j) \equiv \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$

Nonlinear SVMs: The Kernel Trick

- An example:

2-dimensional vectors $\mathbf{x}=[x_1\ \ x_2]$;

let $K(\mathbf{x}_i,\mathbf{x}_j)=(1 + \mathbf{x}_i^T\mathbf{x}_j)^2$,

Need to show that $K(\mathbf{x}_i,\mathbf{x}_j) = \varphi(\mathbf{x}_i)^T\varphi(\mathbf{x}_j)$:

$$
\begin{aligned}
K(\mathbf{x}_i,\mathbf{x}_j)&=(1 + \mathbf{x}_i^T\mathbf{x}_j)^2,\\
&= 1+ x_{i1}^2x_{j1}^2 + 2\, x_{i1}x_{j1}\, x_{i2}x_{j2}+ x_{i2}^2x_{j2}^2 + 2x_{i1}x_{j1} + 2x_{i2}x_{j2}\\
&= [1\ \ x_{i1}^2\ \ \sqrt{2}\, x_{i1}x_{i2}\ \ x_{i2}^2\ \ \sqrt{2}x_{i1}\ \ \sqrt{2}x_{i2}]^T\, [1\ \ x_{j1}^2\ \ \sqrt{2}\, x_{j1}x_{j2}\ \ x_{j2}^2\ \ \sqrt{2}x_{j1}\ \ \sqrt{2}x_{j2}]\\
&= \varphi(\mathbf{x}_i)^T\varphi(\mathbf{x}_j), \quad \text{where } \varphi(\mathbf{x}) = [1\ \ x_1^2\ \ \sqrt{2}\, x_1x_2\ \ x_2^2\ \ \sqrt{2}x_1\ \ \sqrt{2}x_2]
\end{aligned}
$$

This slide is courtesy of *www.iro.umontreal.ca/~pift6080/documents/papers/**svm**_tutorial.**ppt***

Nonlinear SVMs: The Kernel Trick

- Examples of commonly-used kernel functions:
  - Linear kernel: $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$
  - Polynomial kernel: $K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i^T \mathbf{x}_j)^p$
  - Gaussian (Radial-Basis Function (RBF)) kernel:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\left\|\mathbf{x}_i - \mathbf{x}_j\right\|^2}{2\sigma^2}\right)$$

  - Sigmoid:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\beta_0 \mathbf{x}_i^T \mathbf{x}_j + \beta_1)$$

- In general, functions that satisfy *Mercer's condition* can be kernel functions.

等價於存在某個特徵映射 Φ，使k(x,y)=⟨Φ(x),Φ(y)⟩.

Nonlinear SVM: Optimization

- Formulation: (Lagrangian Dual Problem)

$$\text{maximize} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

such that

$$0 \le \alpha_i \le C$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0$$

- The solution of the discriminant function is

$$g(\mathbf{x}) = \sum_{i \in \mathrm{SV}} \alpha_i K(\mathbf{x}_i, \mathbf{x}) + b$$

- The optimization technique is the same.

Support Vector Machine: Algorithm

- 1. Choose a kernel function
- 2. Choose a value for *C*
- 3. Solve the quadratic programming problem (many software packages available)
- 4. Construct the discriminant function from the support vectors

Some Issues

- Choice of kernel
  - Gaussian or polynomial kernel is default
  - if ineffective, more elaborate kernels are needed
  - domain experts can give assistance in formulating appropriate similarity measures

- Choice of kernel parameters
  - e.g. σ in Gaussian kernel
  - σ is the distance between closest points with different classifications
  - In the absence of reliable criteria, applications rely on the use of a validation set or cross-validation to set such parameters.

- Optimization criterion – Hard margin v.s. Soft margin
  - a lengthy series of experiments in which various parameters are tested

This slide is courtesy of *www.iro.umontreal.ca/~pift6080/documents/papers/**svm**_tutorial.**ppt***

Summary: Support Vector Machine

- 1. Large Margin Classifier
  - Better generalization ability & less over-fitting
- 2. The Kernel Trick
  - Map data points to higher dimensional space in order to make them linearly separable.
  - Since only dot product is used, we do not need to represent the mapping explicitly.

Additional Resource

- http://www.kernel-machines.org/

Classification

Classification – Basic Concepts

Decision Tree Induction

Bayes Classification Methods

Rule-Based Classification

Techniques to Improve Classification Accuracy: Ensemble Methods

Lazy Learner

Support Vector Machine

Evaluations

Model Evaluation and Selection

- Evaluation metrics: How can we measure accuracy? Other metrics to consider?
- Use **validation test set** of class-labeled tuples instead of training set when assessing accuracy
- Methods for estimating a classifier's accuracy:
  - Holdout method, random subsampling
  - Cross-validation
  - Bootstrap
- Comparing classifiers:
  - Cost-benefit analysis and ROC Curves

Classifier Evaluation Metrics: Confusion Matrix

Confusion Matrix:

Actual class\Predicted class	C_1	$\neg C_1$
C_1	**True Positives (TP)**	**False Negatives (FN)**
$\neg C_1$	**False Positives (FP)**	**True Negatives (TN)**

Example of Confusion Matrix:

Actual class\Predicted class	buy_computer = yes	buy_computer = no	Total
buy_computer = yes	**6954**	**46**	7000
buy_computer = no	**412**	**2588**	3000
Total	7366	2634	10000

- Given *m* classes, an entry, $\boldsymbol{CM_{i,j}}$ in a **confusion matrix** indicates # of tuples in class *i* that were labeled by the classifier as class *j*
- May have extra rows/columns to provide totals

Classifier Evaluation Metrics: Accuracy, Error Rate, Sensitivity and Specificity

A\P	C	¬C	
C	**TP**	**FN**	**P**
¬C	**FP**	**TN**	**N**
	P'	**N'**	**All**

- **Classifier Accuracy,** or recognition rate: percentage of test set tuples that are correctly classified
  - **Accuracy = (TP + TN)/All**
- **Error rate:** *1 – accuracy*, or
  - **Error rate = (FP + FN)/All**

- **Class Imbalance Problem**:
  - One class may be *rare*, e.g. fraud, or HIV-positive
  - Significant *majority of the negative class* and minority of the positive class
  - **Sensitivity**: True Positive recognition rate
    - **Sensitivity = TP/P**
  - **Specificity**: True Negative recognition rate
    - **Specificity = TN/N**

Classifier Evaluation Metrics: Precision and Recall, and F-measures

- **Precision**: exactness – what % of tuples that the classifier labeled as positive are actually positive

$$precision = \frac{TP}{TP + FP}$$

  - Does not care how many positive instances are mislabeled as negative, i.e., FN
- **Recall:** completeness – what % of positive tuples did the classifier label as positive?

$$recall = \frac{TP}{TP + FN} = \frac{TP}{P}$$

  - But did not tell how many negative instances are mislabeled as positive, i.e., FP
- Perfect score is 1.0
- Inverse relationship between precision & recall
- ***F* measure (*F_1* or *F*-score)**: harmonic mean of precision and recall,

$$F = \frac{2 \times precision \times recall}{precision + recall}$$

- ***F_β***: weighted measure of precision and recall
  - assigns ß times as much weight to recall as to precision

$$F_\beta = \frac{(1 + \beta^2) \times precision \times recall}{\beta^2 \times precision + recall}$$

Classifier Evaluation Metrics: Example

Actual Class\Predicted class	cancer = yes	cancer = no	Total	Recognition(%)
cancer = yes	**90**	**210**	300	30.00 (*sensitivity)*
cancer = no	**140**	**9560**	9700	98.56 (*specificity)*
Total	230	9770	10000	96.50 (*accuracy*)

- *Precision* = 90/230 = 39.13% *Recall* = 90/300 = 30.00%

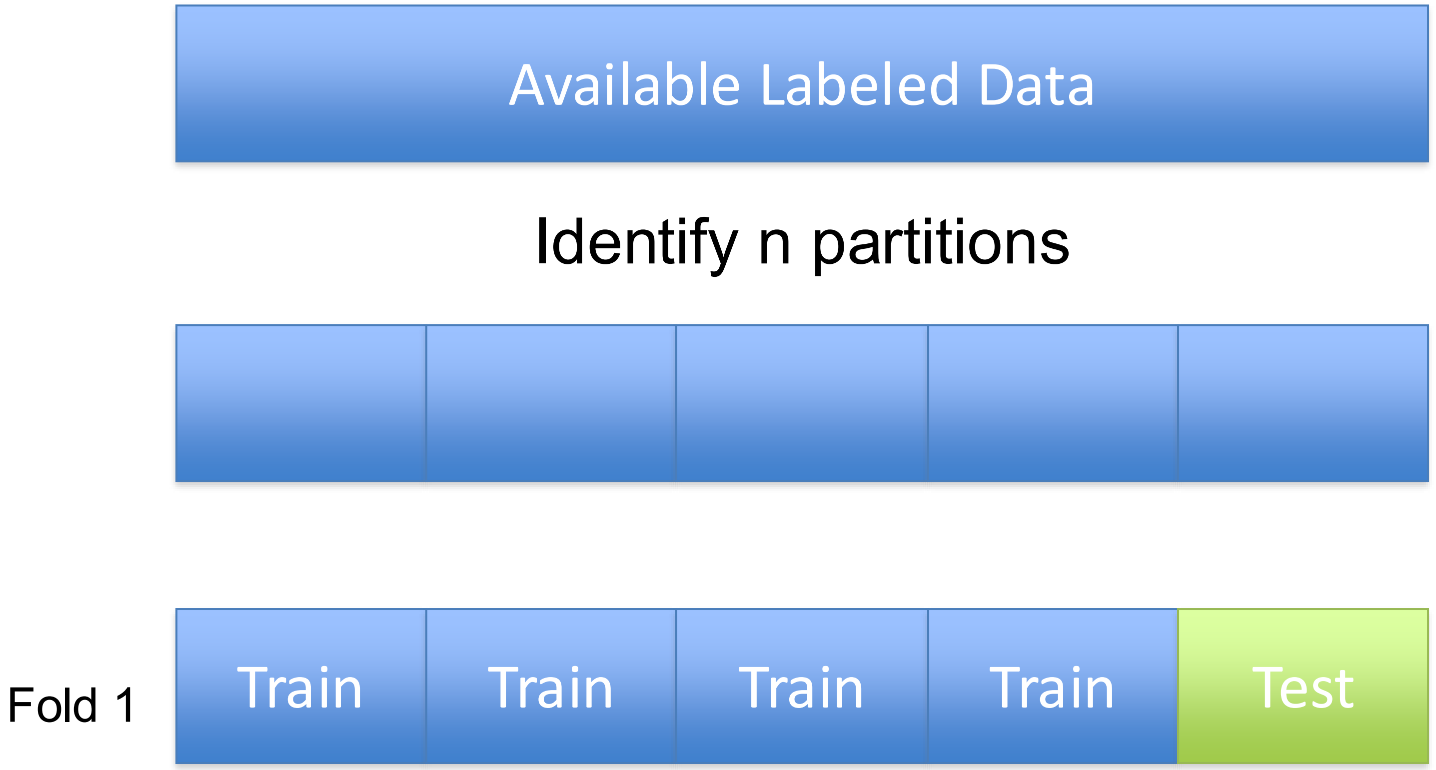
Measure	Formula
accuracy, recognition rate	$\frac{TP+TN}{P+N}$
error rate, misclassification rate	$\frac{FP+FN}{P+N}$
sensitivity, true positive rate, recall	$\frac{TP}{P}$
specificity, true negative rate	$\frac{TN}{N}$
precision	$\frac{TP}{TP+FP}$
F, F_1, F-score, harmonic mean of precision and recall	$\frac{2 \times precision \times recall}{precision + recall}$
F_β, where β is a non-negative real number	$\frac{(1+\beta^2) \times precision \times recall}{\beta^2 \times precision + recall}$

Evaluating Classifier Accuracy: Holdout & Cross-Validation Methods

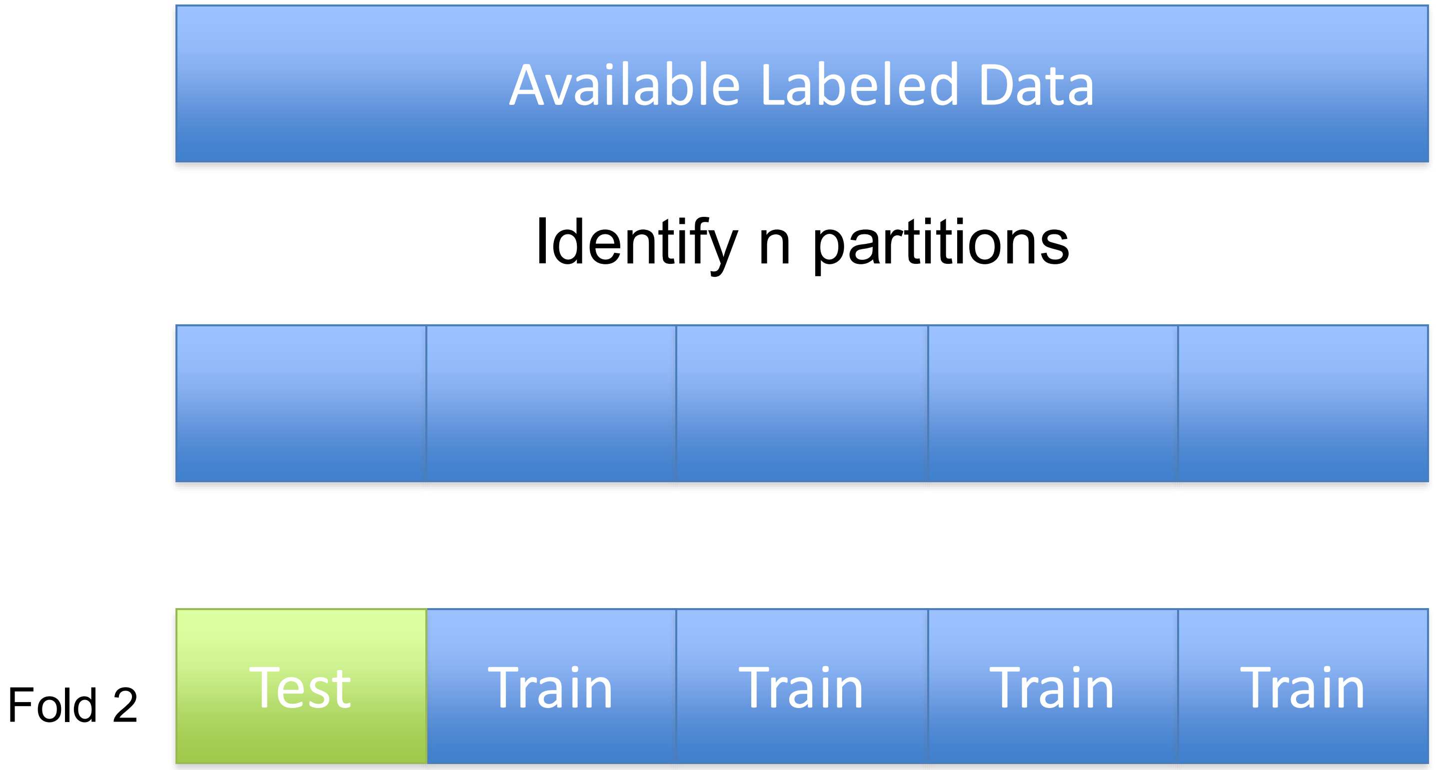
- **Holdout method**
  - Given data is randomly partitioned into two independent sets
    - Training set (e.g., 2/3) for model construction
    - Test set (e.g., 1/3) for accuracy estimation
  - Random sampling: a variation of holdout
    - Repeat holdout k times, accuracy = avg. of the accuracies obtained

- **Cross-validation** (*k*-fold, where k = 10 is most popular)
  - Randomly partition the data into *k mutually exclusive* subsets, each approximately equal size
  - At *i*-th iteration, use D_i as test set and others as training set
  - Leave-one-out: *k* folds where *k* = # of tuples, for small sized data
  - ***Stratified cross-validation***: folds are stratified so that class dist. in each fold is approx. the same as that in the initial data

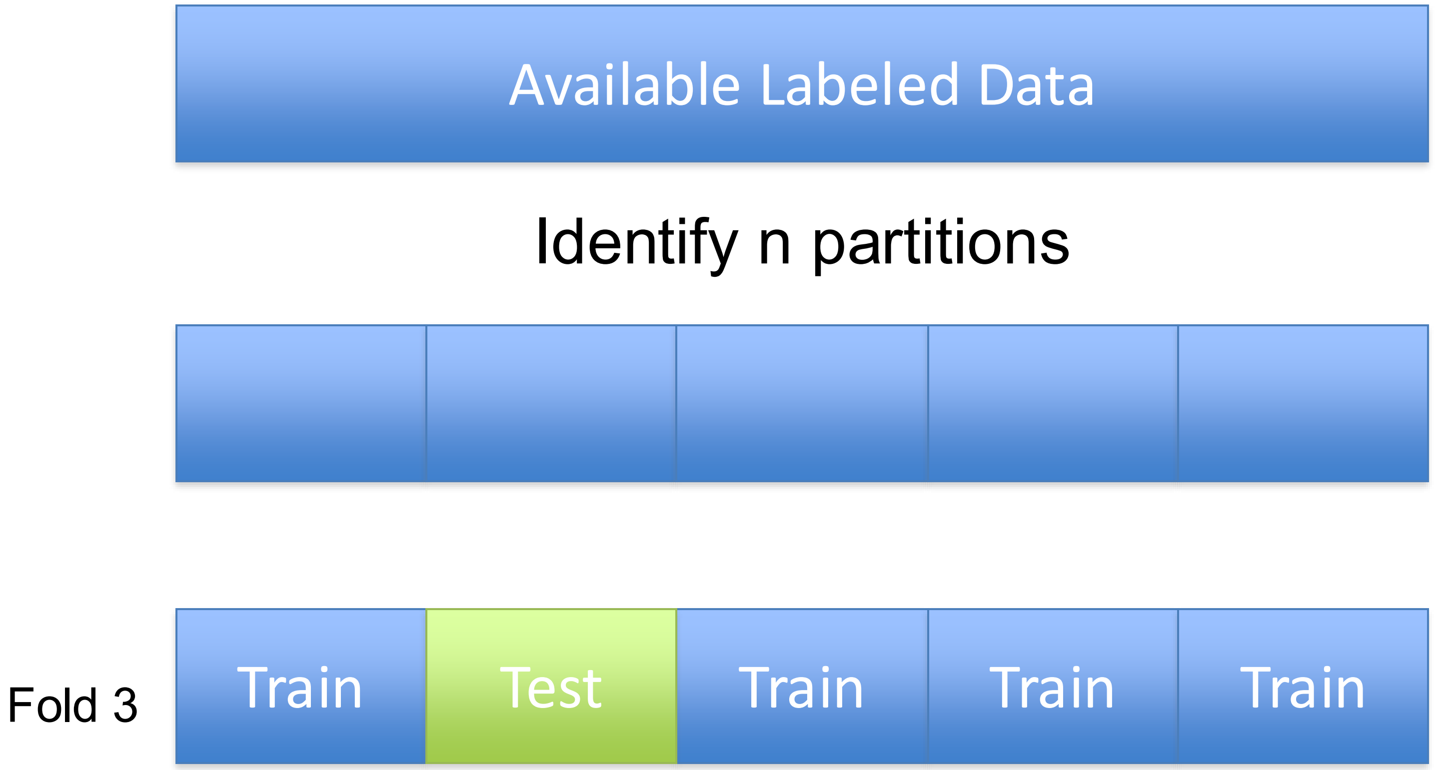
Cross-validation visualized



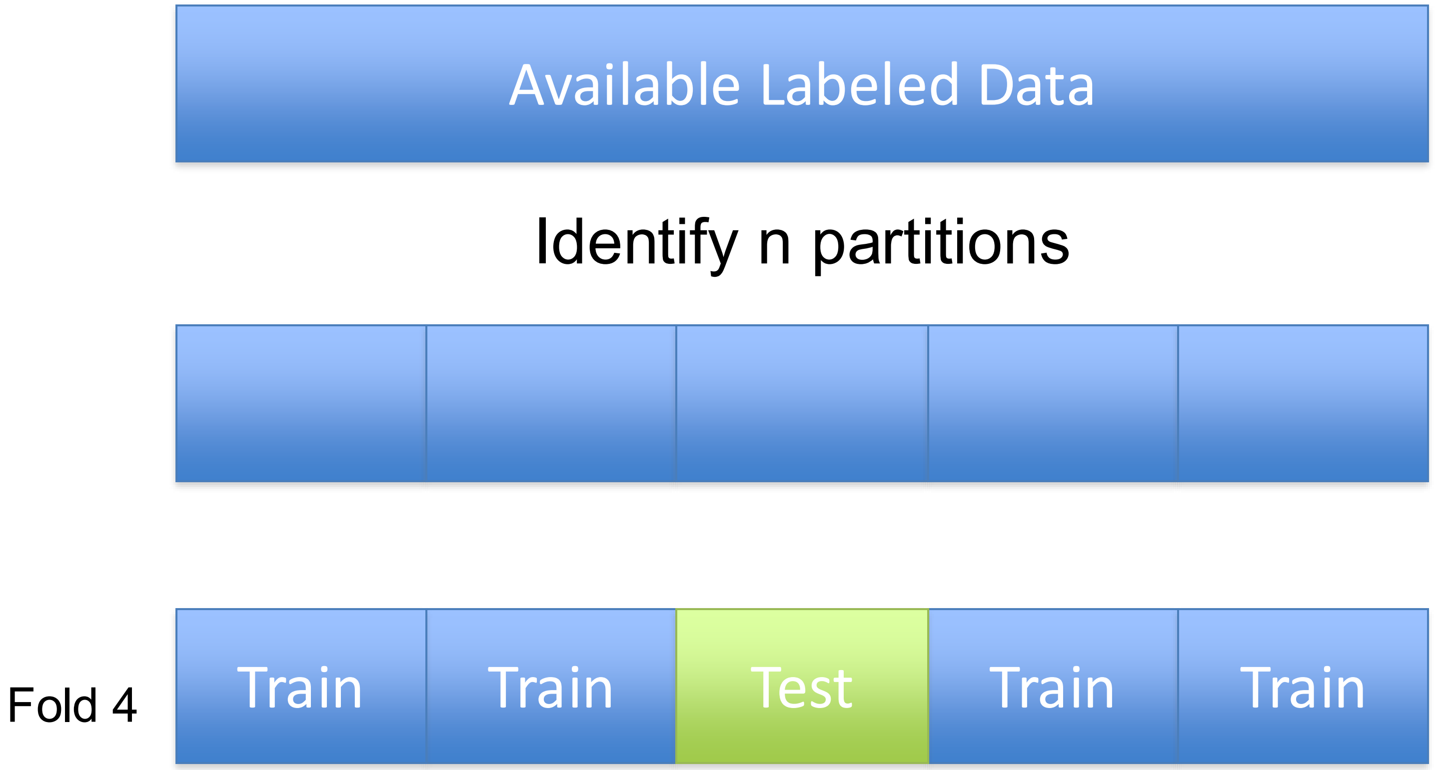
Cross-validation visualized



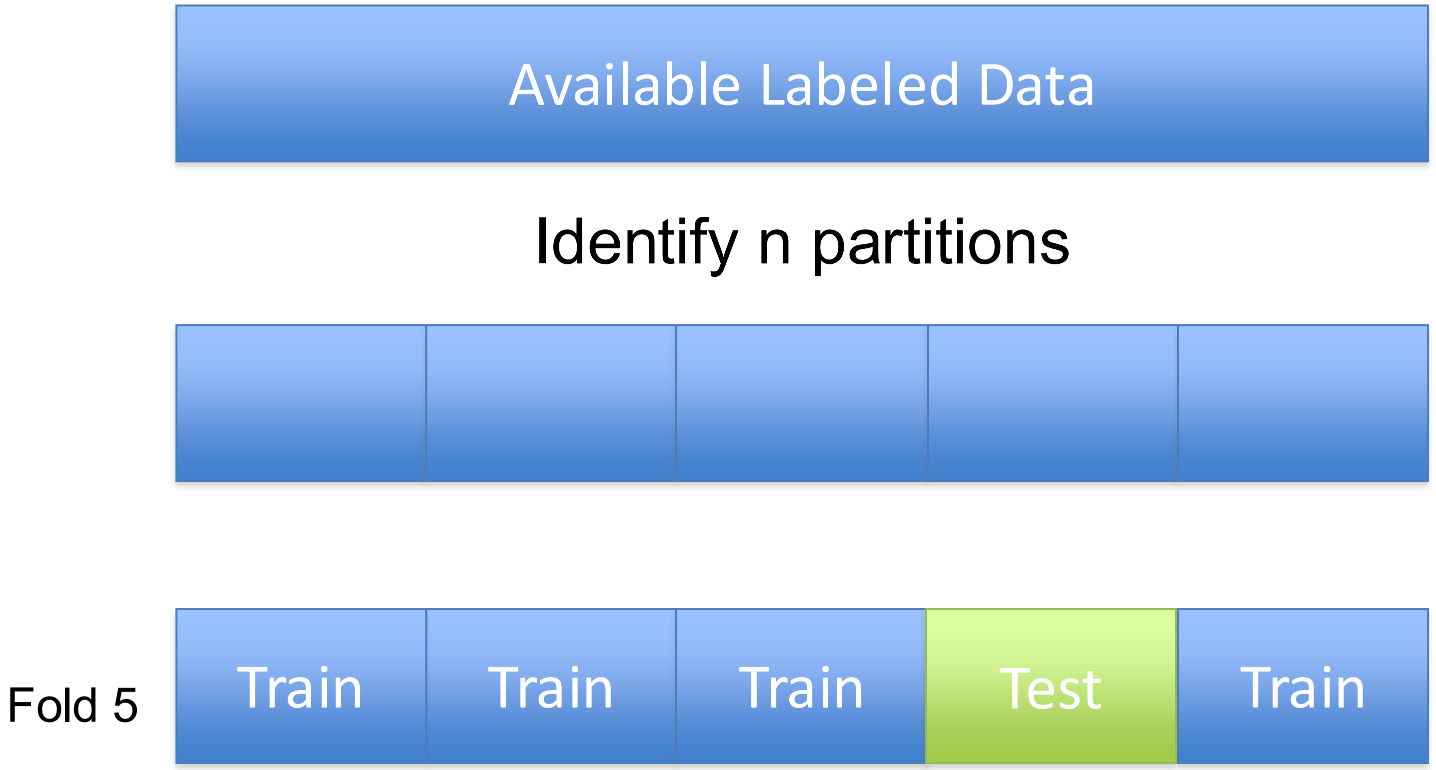
Cross-validation visualized



Cross-validation visualized

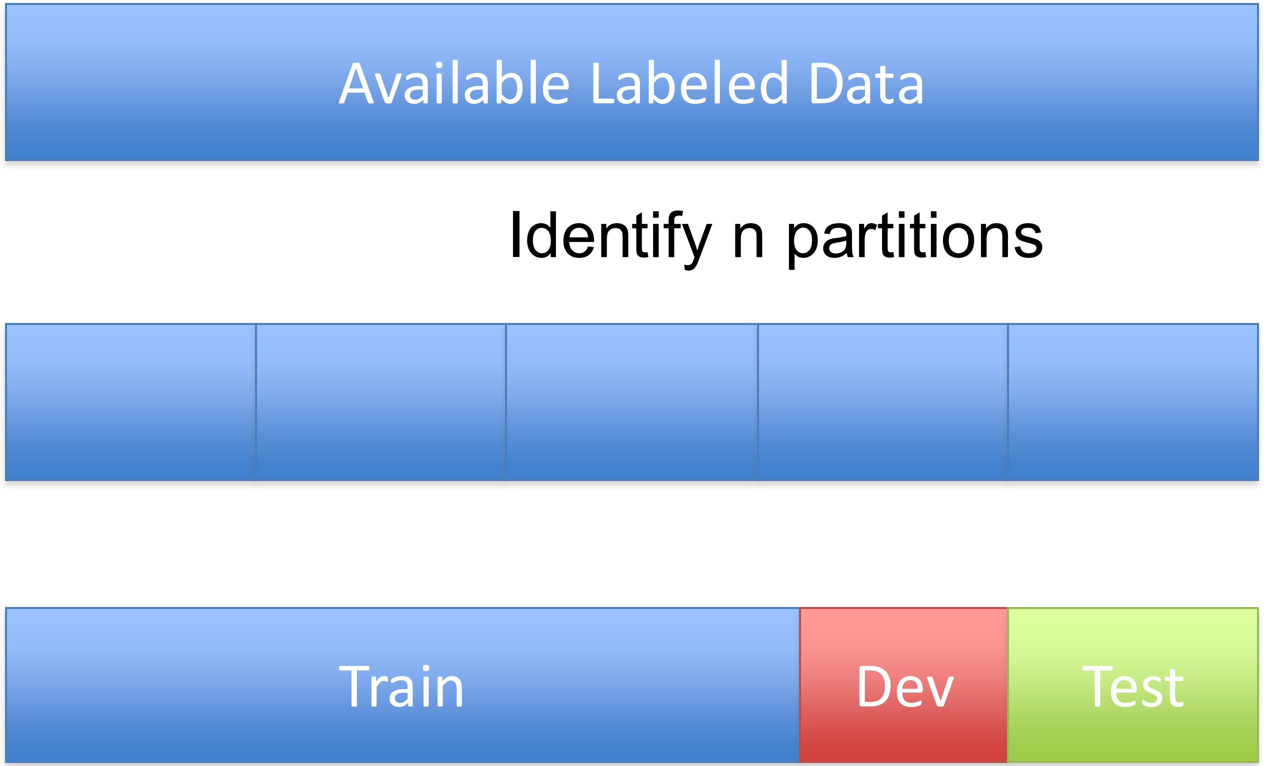


Cross-validation visualized



Calculate Average Performance

Cross-validation visualized



Develop/Tuning set for adjusting hyper parameters

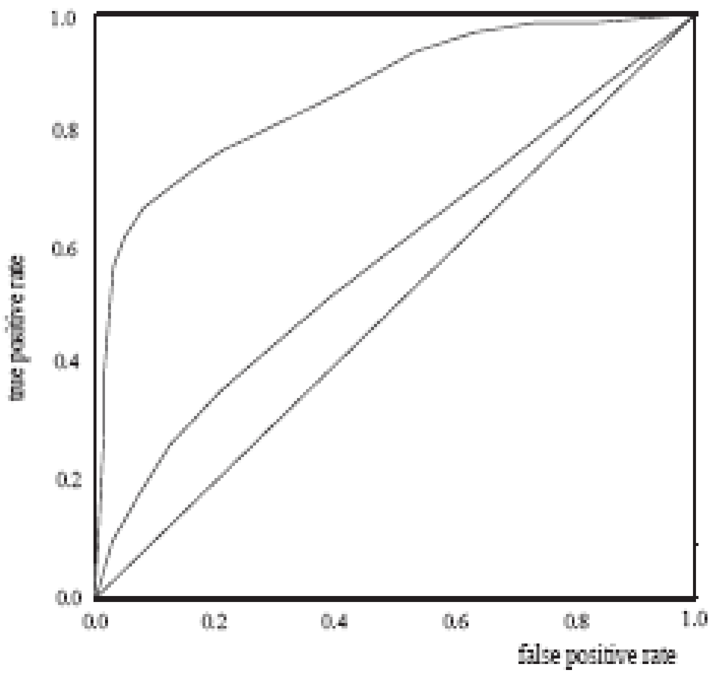
Evaluating Classifier Accuracy: Bootstrap

- **Bootstrap**
  - Works well with small data sets
  - Samples the given training tuples uniformly *with replacement*
    - i.e., each time a tuple is selected, it is equally likely to be selected again and re-added to the training set
- Several bootstrap methods, and a common one is **.632 boostrap**
  - A data set with *d* tuples is sampled *d* times, with replacement, resulting in a training set of *d* samples. The data tuples that did not make it into the training set end up forming the test set. About 63.2% of the original data end up in the bootstrap, and the remaining 36.8% form the test set (since $(1 - 1/d)^d \approx e^{-1} = 0.368$) Out-OF-BAG (OOB)
  - Repeat the sampling procedure *k* times, overall accuracy of the model:

$$Acc(M) = \frac{1}{k}\sum_{i=1}^{k}(0.632 \times Acc(M_i)_{test\_set} + 0.368 \times Acc(M_i)_{train\_set})$$

Model Selection: ROC Curves

- **ROC** (Receiver Operating Characteristics) curves: for visual comparison of classification models
- Shows the trade-off between the true positive rate and the false positive rate
- The area under the ROC curve is a measure of the accuracy of the model
- Rank the test tuples in decreasing order: the one that is most likely to belong to the positive class appears at the top of the list
- The closer to the diagonal line (i.e., the closer the area is to 0.5), the less accurate is the model



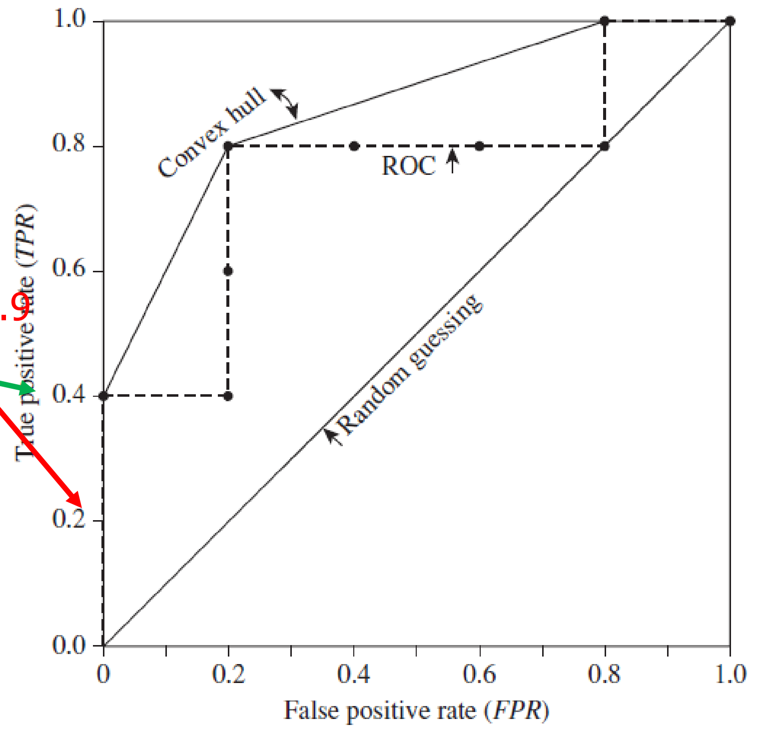
- Vertical axis represents the true positive rate
- Horizontal axis rep. the false positive rate
- The plot also shows a diagonal line
- A model with perfect accuracy will have an area of 1.0

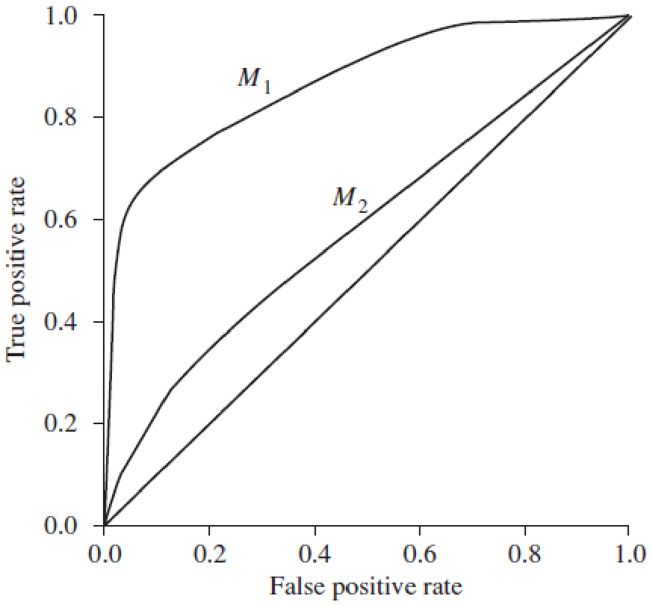
ROC curves

Tuple #	Class	Prob.	TP	FP	TN	FN	TPR	FPR
1	P	0.90	1	0	5	4	0.2	0
2	P	0.80	2	0	5	3	0.4	0
3	N	0.70	2	1	4	3	0.4	0.2
4	P	0.60	3	1	4	2	0.6	0.2
5	P	0.55	4	1	4	1	0.8	0.2
6	N	0.54	4	2	3	1	0.8	0.4
7	N	0.53	4	3	2	1	0.8	0.6
8	N	0.51	4	4	1	1	0.8	0.8
9	P	0.50	5	4	1	0	1.0	0.8
10	N	0.40	5	5	0	0	1.0	1.0

t=0.9

t=0.8





Classification

- Classification – Basic Concepts
- Decision Tree Induction
- Bayes Classification Methods
- Rule-Based Classification
- Techniques to Improve Classification Accuracy: Ensemble Methods
- Lazy Learner
- Support Vector Machine
- Summary

Issues Affecting Model Selection

- **Accuracy**
  - classifier accuracy: predicting class label
- **Speed**
  - time to construct the model (training time)
  - time to use the model (classification/prediction time)
- **Robustness**: handling noise and missing values
- **Scalability**: efficiency in disk-resident databases
- **Interpretability**
  - understanding and insight provided by the model
- Other measures, e.g., goodness of rules, such as decision tree size or compactness of classification rules

References (1)

- C. Apte and S. Weiss. **Data mining with decision trees and decision rules**. Future Generation Computer Systems, 13, 1997
- C. M. Bishop, **Neural Networks for Pattern Recognition**. Oxford University Press, 1995
- L. Breiman, J. Friedman, R. Olshen, and C. Stone. **Classification and Regression Trees**. Wadsworth International Group, 1984
- C. J. C. Burges. **A Tutorial on Support Vector Machines for Pattern Recognition**. *Data Mining and Knowledge Discovery*, 2(2): 121-168, 1998
- P. K. Chan and S. J. Stolfo. **Learning arbiter and combiner trees from partitioned data for scaling machine learning**. KDD'95
- H. Cheng, X. Yan, J. Han, and C.-W. Hsu, **Discriminative Frequent Pattern Analysis for Effective Classification**, ICDE'07
- H. Cheng, X. Yan, J. Han, and P. S. Yu, **Direct Discriminative Pattern Mining for Effective Classification**, ICDE'08
- W. Cohen. **Fast effective rule induction**. ICML'95
- G. Cong, K.-L. Tan, A. K. H. Tung, and X. Xu. **Mining top-k covering rule groups for gene expression data**. SIGMOD'05

Agenda

- Philosophy
- Machine Learning Concept
- Clustering
- Classification
- Reflection

REFLECTION


THIS IS YOUR MACHINE LEARNING SYSTEM?
YUP! YOU POUR THE DATA INTO THIS BIG PILE OF LINEAR ALGEBRA, THEN COLLECT THE ANSWERS ON THE OTHER SIDE.
WHAT IF THE ANSWERS ARE WRONG?
JUST STIR THE PILE UNTIL THEY START LOOKING RIGHT.
DATA
ANSWERS

THREE LAWS OF ROBOTICS (ASIMOV'S LAWS)

- A robot may not injure a human being or, through inaction, allow a human being to come to harm.
- A robot must obey the orders given it by human beings except where such orders would conflict with the First Law.
- A robot must protect its own existence as long as such protection does not conflict with the First or Second Laws.

http://www.continue-play.com/2015/08/17/baymax-from-big-hero-6-will-be-in-kingdom-

Do not edit
How to change the design

Join at slido.com
#2829580

The Slido app must be installed on every computer you're presenting from

slido

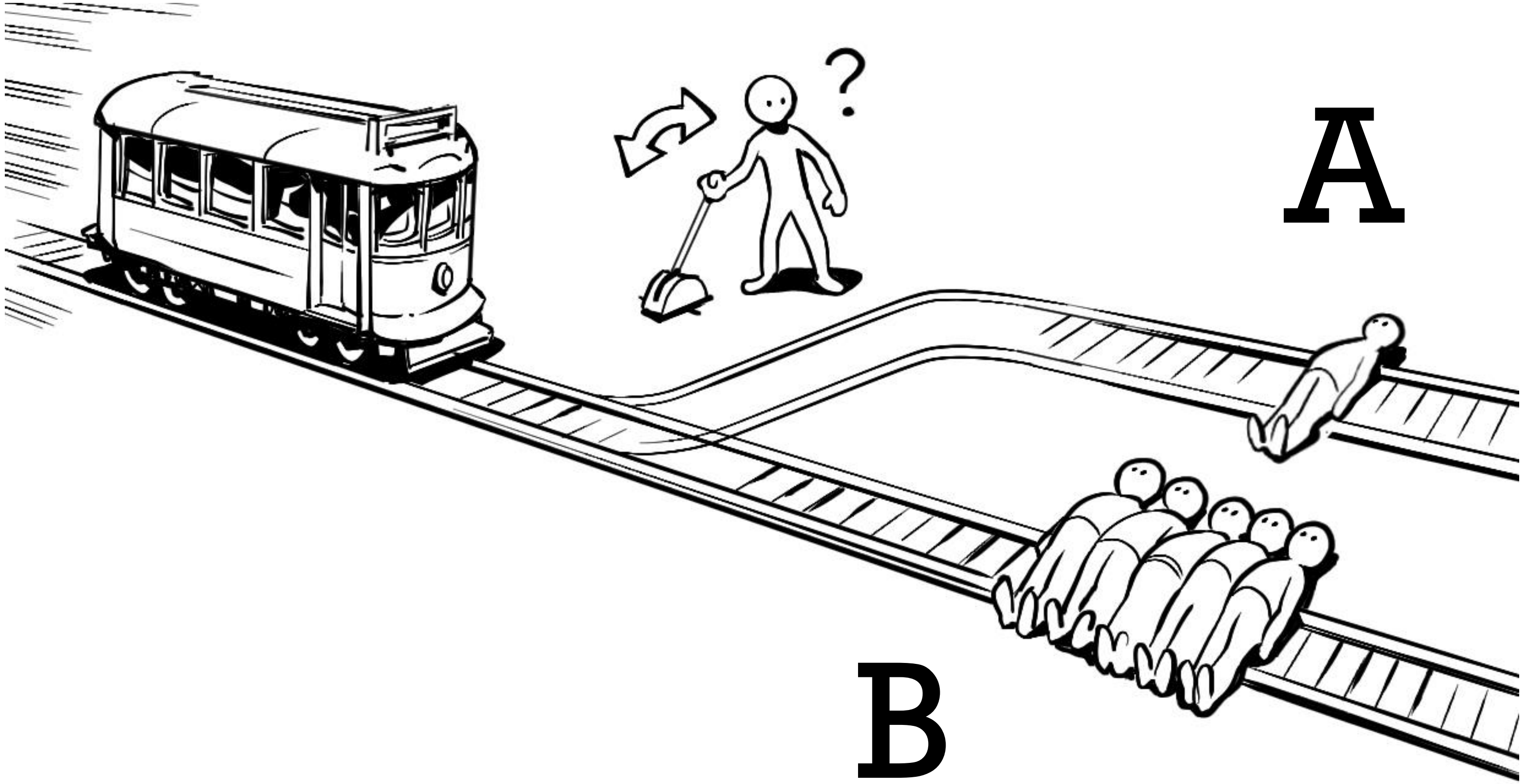
slido

Please download and install the
Slido app on all computers you use

Join at slido.com
#2829580

ⓘ Start presenting to display the joining instructions on this slide.

WHAT IF...



http://hornydragon.blogspot.com/2016/10/780.

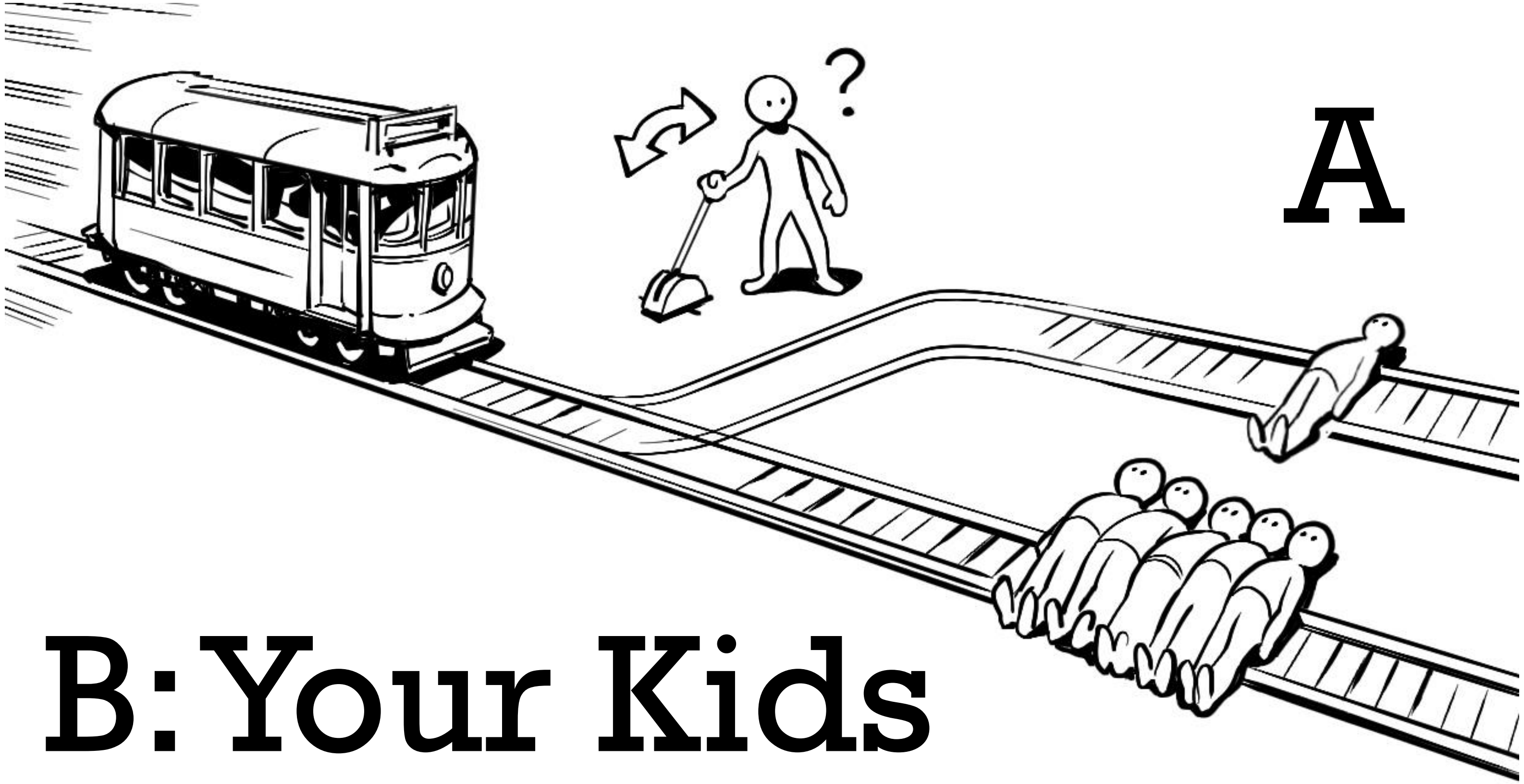
slido

Please download and install the
Slido app on all computers you use

A or B?

ⓘ Start presenting to display the poll results on this slide.

WHAT IF...



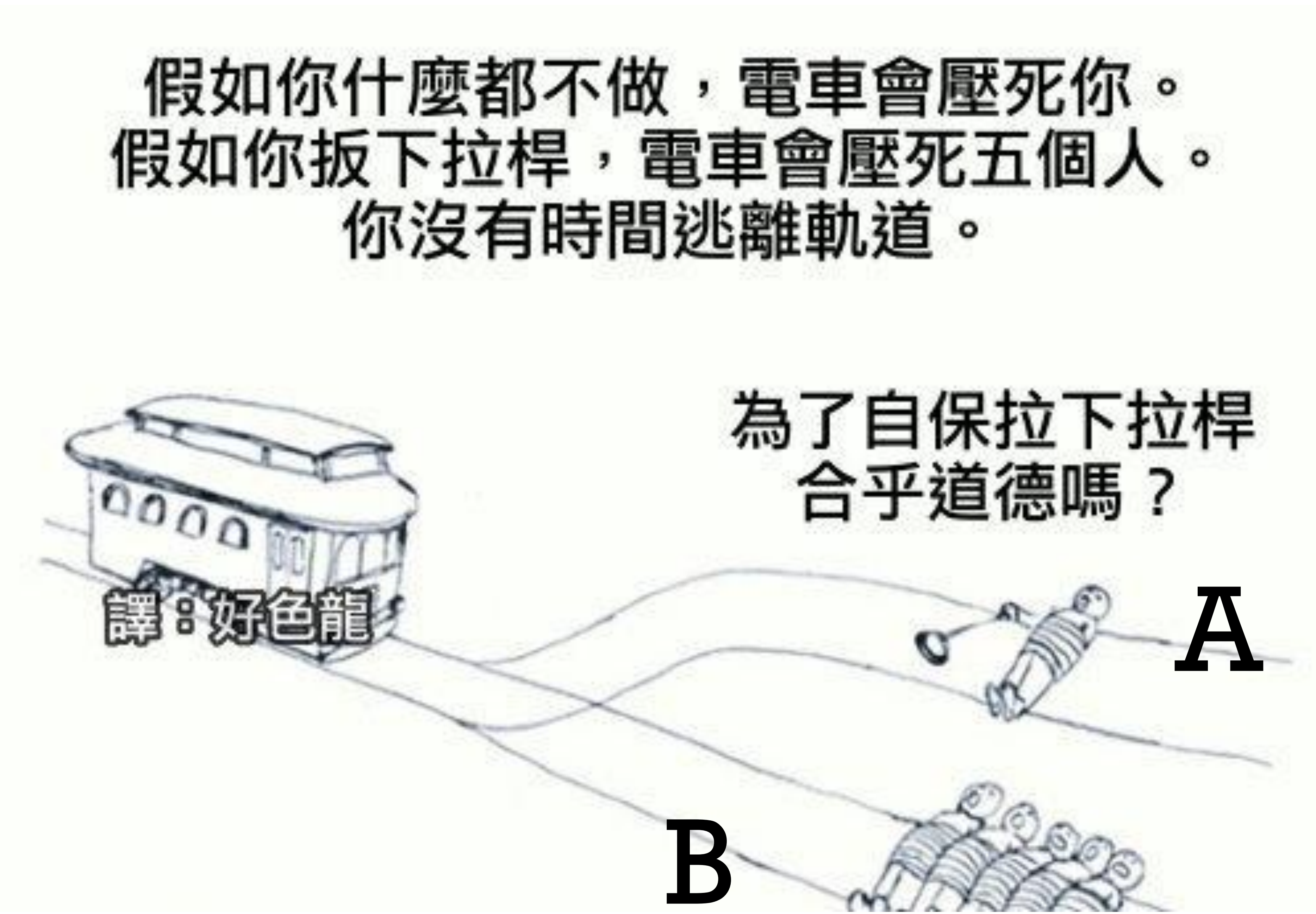
http://hornydragon.blogspot.com/2016/10/780.

slido

Please download and install the
Slido app on all computers you use

A or B?

ⓘ Start presenting to display the poll results on this slide.


假如你什麼都不做，電車會壓死你。
假如你扳下拉桿，電車會壓死五個人。
你沒有時間逃離軌道。
為了自保拉下拉桿
合乎道德嗎？
譯：好色龍
A
B

slido

Please download and install the
Slido app on all computers you use

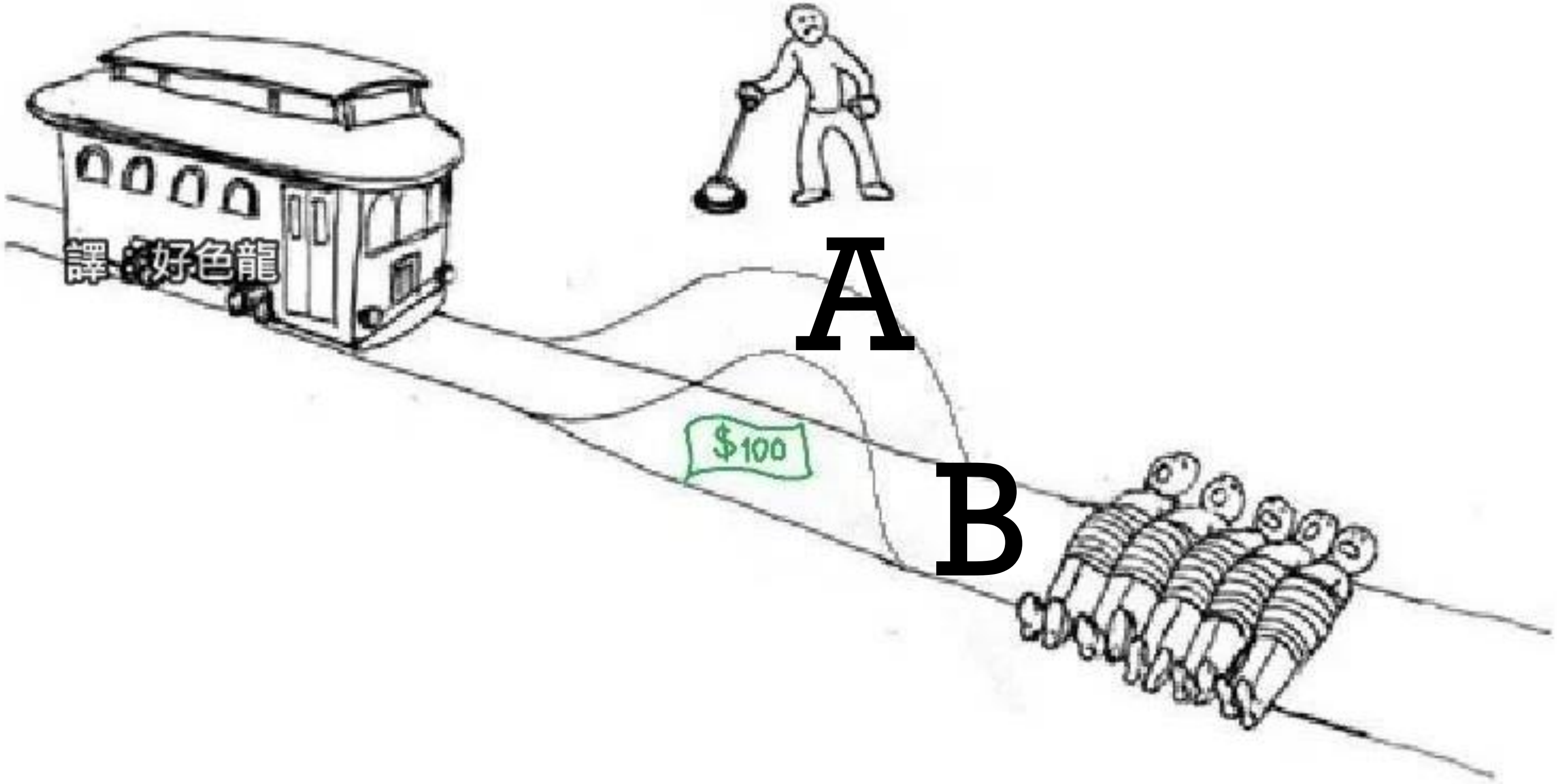
A or B?

ⓘ Start presenting to display the poll results on this slide.

1M USDS!

鐵軌上放著一張百元美金鈔票。
假如你什麼都不做，電車會輾爛鈔票然後撞死五個人。
假如你拉下拉桿，電車會繞過鈔票，然後撞死五個人，
然後你可以把鈔票撿走。

假如你拉下了拉桿，你的舉動該受到譴責嗎？
還是說這是完全合理的作法？

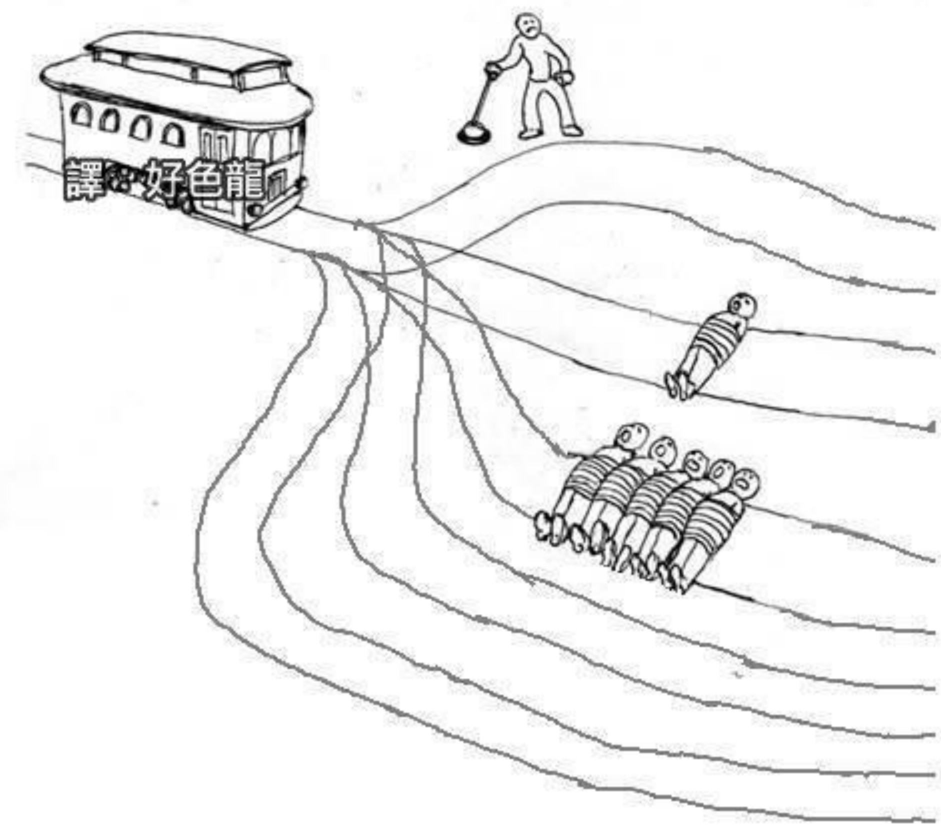


slido

Please download and install the
Slido app on all computers you use

A or B

ⓘ Start presenting to display the poll results on this slide.



賭徒電車難題

假如你什麼都不做，
電車會撞死一個人。

假如你拉下拉桿，
電車會隨機改變軌道，
有3/4的機率開上空軌，
但是有1/4的機率
撞死五個人。

你會拉下拉桿嗎？

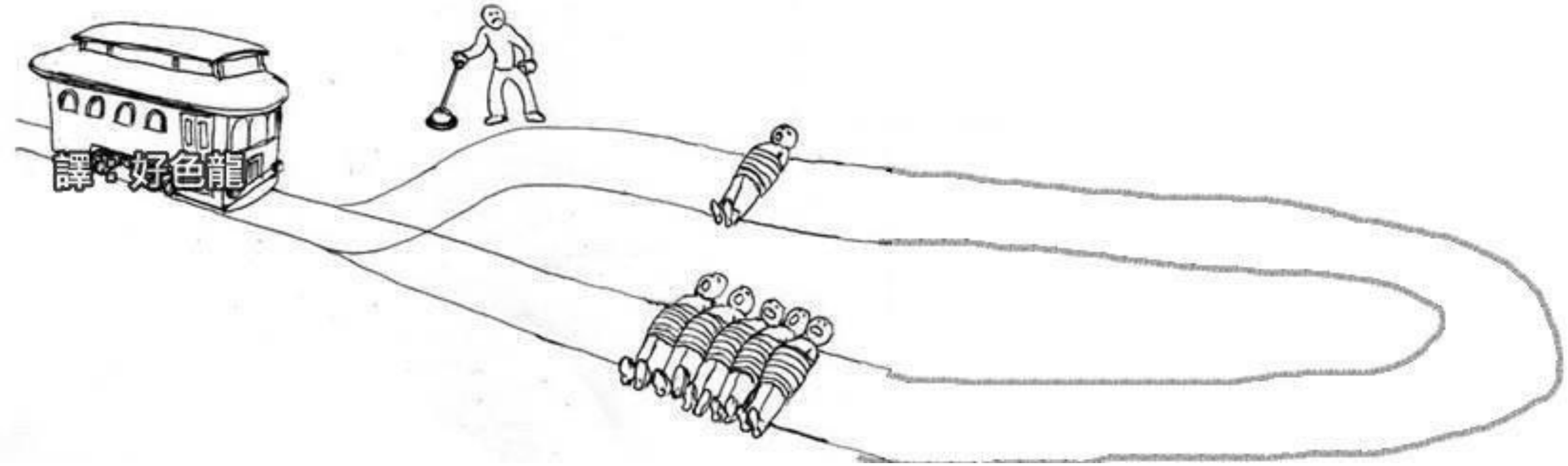
slido

Please download and install the
Slido app on all computers you use

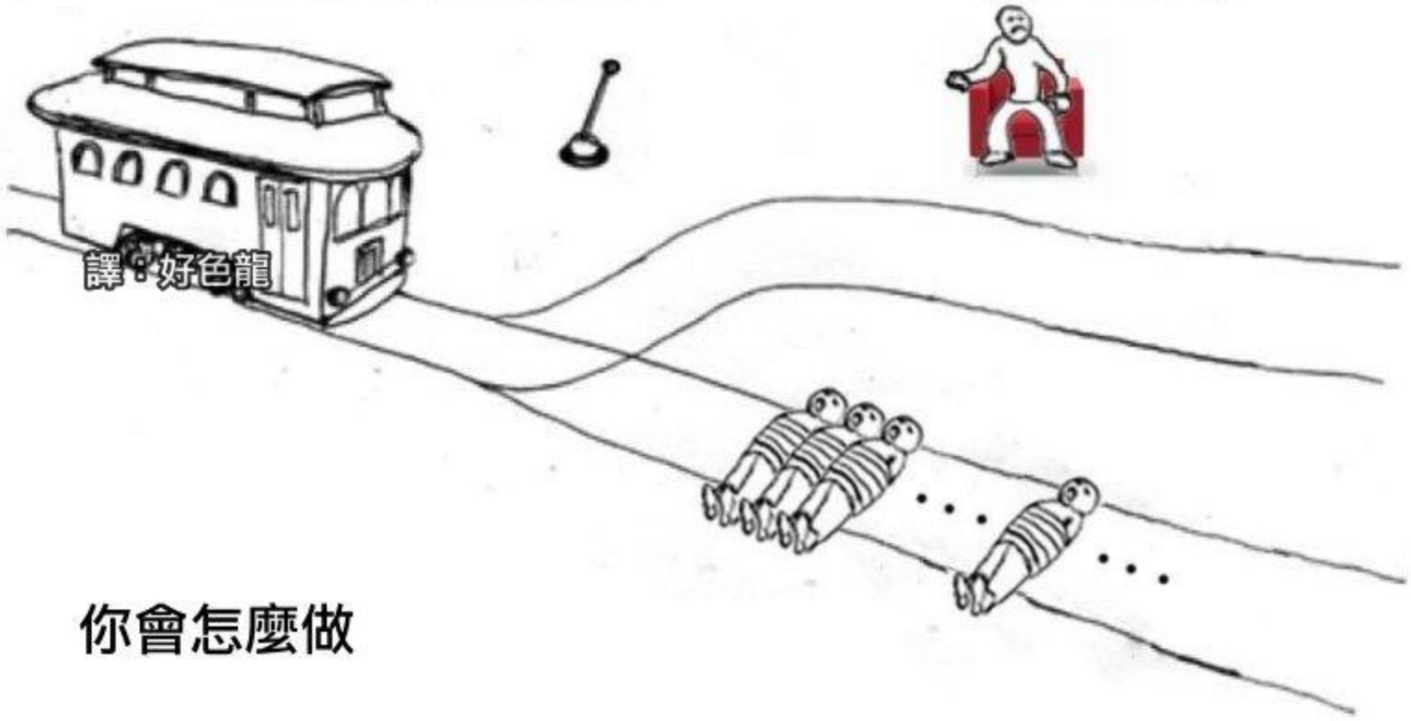
Choice?

ⓘ Start presenting to display the poll results on this slide.

何者較人道：讓五個人在死前看見一個人死在眼前，
或是讓一個人在死前看見五個人死在眼前？



假如你什麼都不做，電車會撞死數不清的人。
假如你拉下拉桿，所有人都會得救。
但是拉桿距離你的沙發有五公尺遠，
而且這張沙發舒服到夭壽。



http://hornydragon.blogspot.com/2016/10/780.

MORAL MACHINE

- http://moralmachine.mit.edu/hl/zh

https://kahoot.it/

1st-3rd: 1 pt
4th and 5th: 0.5 pts