PATTERN RECOGNITION AND MACHINE LEARNING

CHAPTER 4: LINEAR MODELS FOR CLASSIFICATION

Learning Objectives

1、What are linear classification models?

2、What are the three linear classification approaches?

3、What is the Fisher's discriminant method?

4、What is the Perceptron method?

5、What is the Gaussian mixture model method?

6、What is the logistic regression method?

7、How to compare the discriminative and generative methods?

8、What is the Bayesian Information Criterion?

Outlines

- Three Approaches to Linear Classification
- Approach I: Discriminant Functions
- Least Square Classification
- Fisher's Linear Discriminants
- Perceptrons
- Approach II: Probabilistic Generative Models
- Approach III: Probabilistic Discriminative Models
- Bayesian Information Criterion

Linear Classification Models

- **Classification is intrinsically non-linear**

  It puts non-identical things in the same class, so a difference in the input vector sometimes causes zero change in the answer

- **Linear classification**: linear adaptive part
  - ✓ followed by a fixed non-linearity.
  - ✓ preceded by a fixed non-linearity (e.g. nonlinear basis functions).

$$y(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + w_0, \qquad Decision = f(y(\mathbf{x}))$$

adaptive linear function

fixed non-linear function

Three Approaches to Classification

- **Use discriminant functions directly** (without probabilities):
  - ✓ Convert the input vector into one or more real values so that a simple operation (like thresholding) can be applied to get the class.

$$y = f(w^T \boldsymbol{x})$$

- **Infer the posterior probabilities with generative models**.
  - ✓ Use prior, and likelihood models to infer posterior models.

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\mathbf{x})}$$

- **Directly construct posterior conditional class probabilities**:
  - ✓ Compute the posterior conditional probability of each class. Then make a decision that minimizes some loss function.

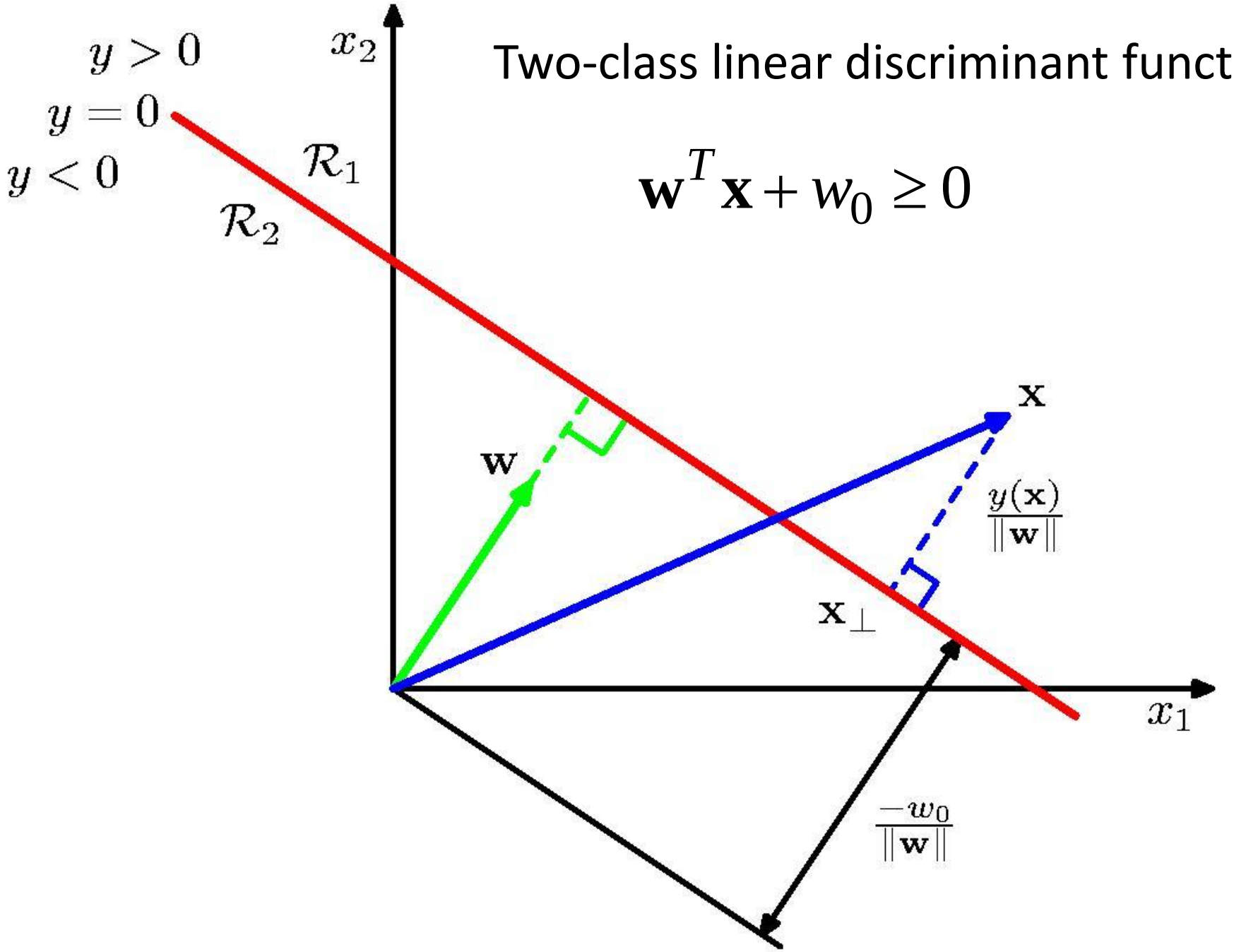
$$p(class = C_k|\boldsymbol{x})$$

Outlines

- Three Approaches to Linear Classification
- Approach I: Discriminant Functions
- Least Square Classification
- Fisher's Discriminants
- Perceptrons
- Approach II: Probabilistic Generative Models
- Approach III: Probabilistic Discriminative Models
- Bayesian Information Criterion

Two-class linear discriminant function:

$$\mathbf{w}^T\mathbf{x} + w_0 \geq 0$$



Discriminant Functions for N classes

- **To use N two-way discriminant functions**

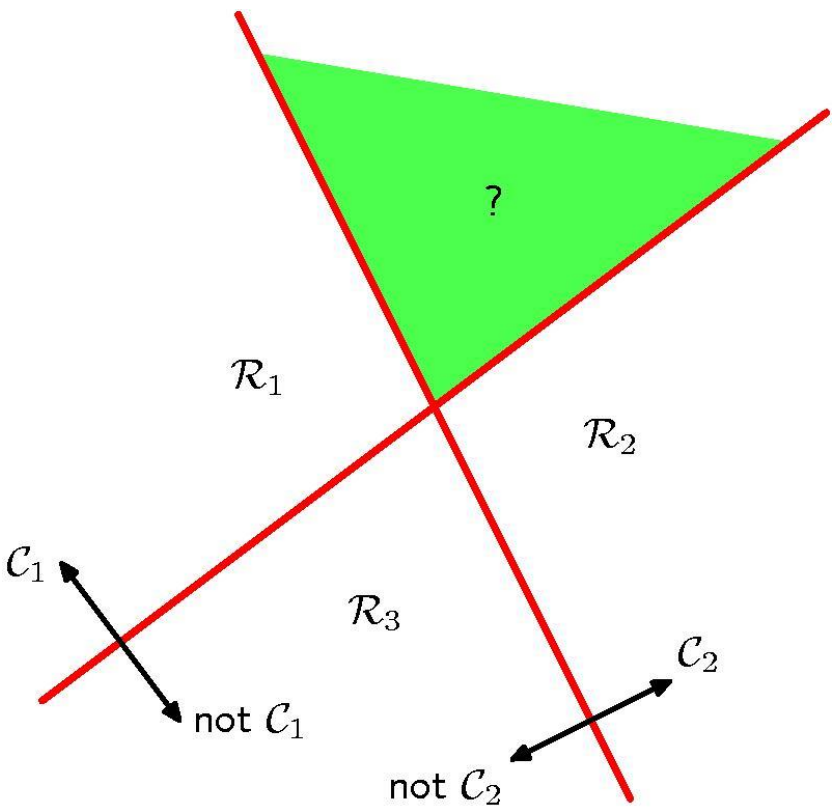
  Each function discriminates one class from the rest.

- **To use N(N-1)/2 two-way discriminant functions**

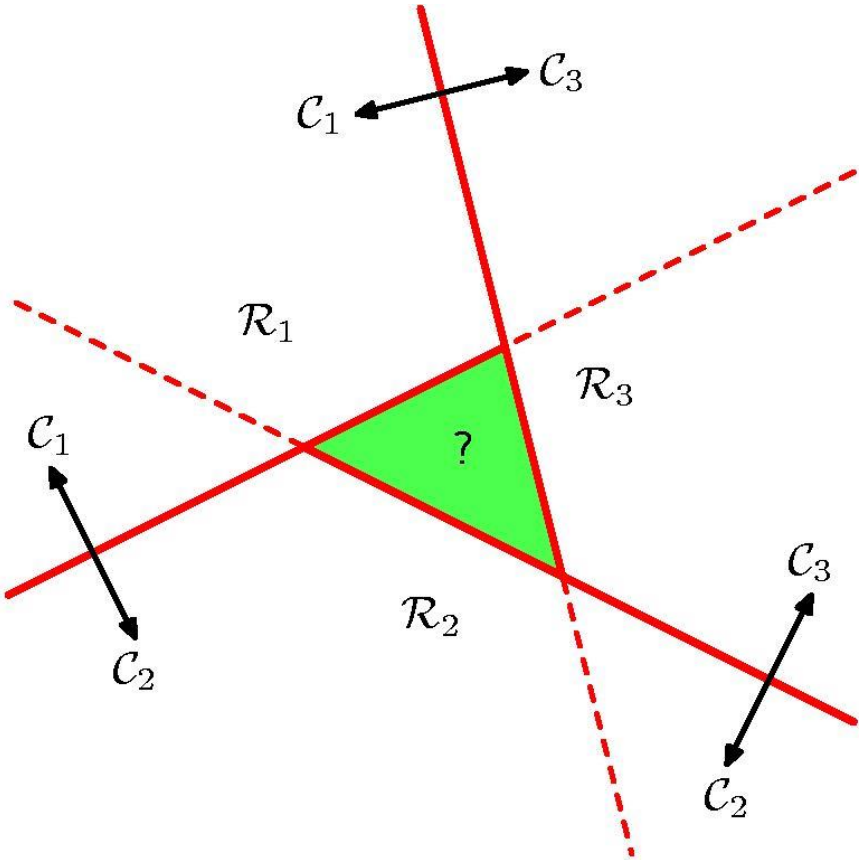
  Each function discriminates between two particular classes.

- **Both methods have problems**

Multi-class Using Two-class Discriminants



More than one good answer

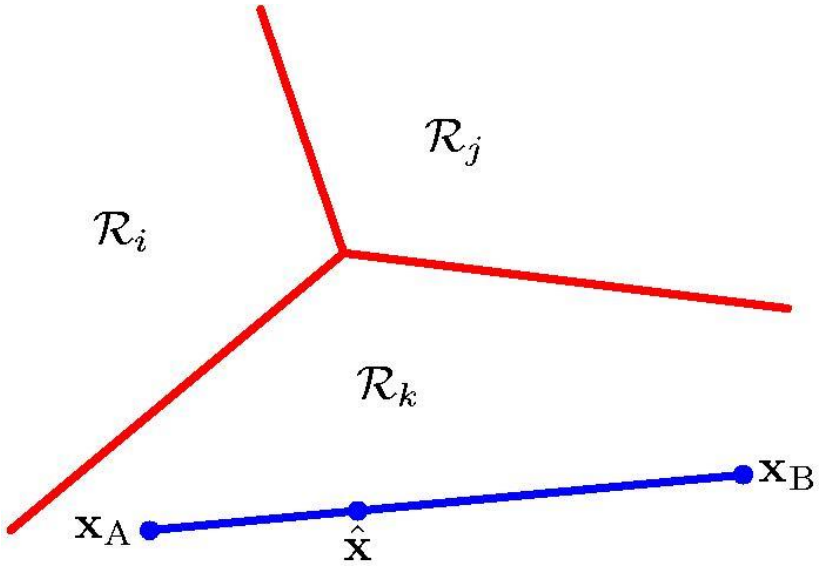


Two-way preferences need not be transitive!

A Simple Solution

Use K discriminant functions, and pick the max. $y_i, y_j, y_k \ldots$

This is guaranteed to give consistent and convex decision regions if y is linear.



$$y_k(\mathbf{x}_A) > y_j(\mathbf{x}_A) \;\; and \;\; y_k(\mathbf{x}_B) > y_j(\mathbf{x}_B)$$

$$implies\,(\,for\;\; positive\;\alpha)\;\; that$$

$$y_k\left(\alpha\,\mathbf{x}_A + (1-\alpha)\,\mathbf{x}_B\right) > y_j\left(\alpha\,\mathbf{x}_A + (1-\alpha)\,\mathbf{x}_B\right)$$

Outlines

- Three Approaches to Linear Classification
- Approach I: Discriminant Functions
- Least Square Classification
- Fisher Discriminant Function
- Perceptrons
- Approach II: Probabilistic Generative Models
- Approach III: Probabilistic Discriminative Models
- Bayesian Information Criterion

Least Squares for Classification

- This is not the right thing to do and it doesn't work as well as better methods, but it is easy:
  - ✓ It reduces classification to least squares regression.
  - ✓ We already know how to do regression. We can just solve for the optimal weights with some matrix algebra .

- We use targets that are equal to the conditional probability of the class given the input.
  - ✓ When there are more than two classes, we treat each class as a separate problem (we cannot get away with this if we use the "max" decision function).

Least Squares Regression

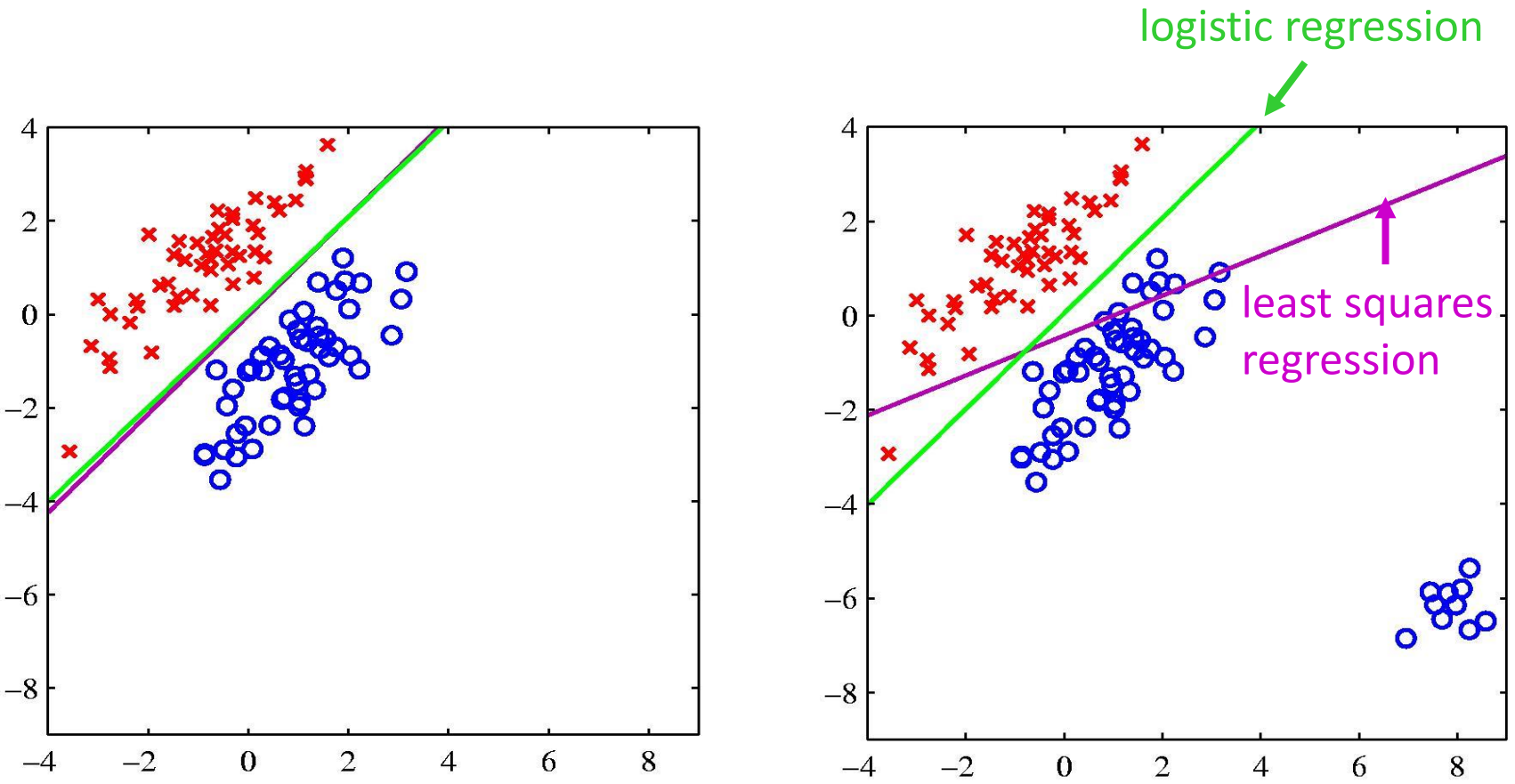
$$y_k(\mathbf{x}) = \mathbf{w}_k^{\mathrm{T}}\mathbf{x} + w_{k0}$$

$$\mathbf{y}(\mathbf{x}) = \widetilde{\mathbf{W}}^{\mathrm{T}}\widetilde{\mathbf{x}}$$

$$E_D(\widetilde{\mathbf{W}}) = \frac{1}{2}\mathrm{Tr}\left\{(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T})^{\mathrm{T}}(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T})\right\}$$

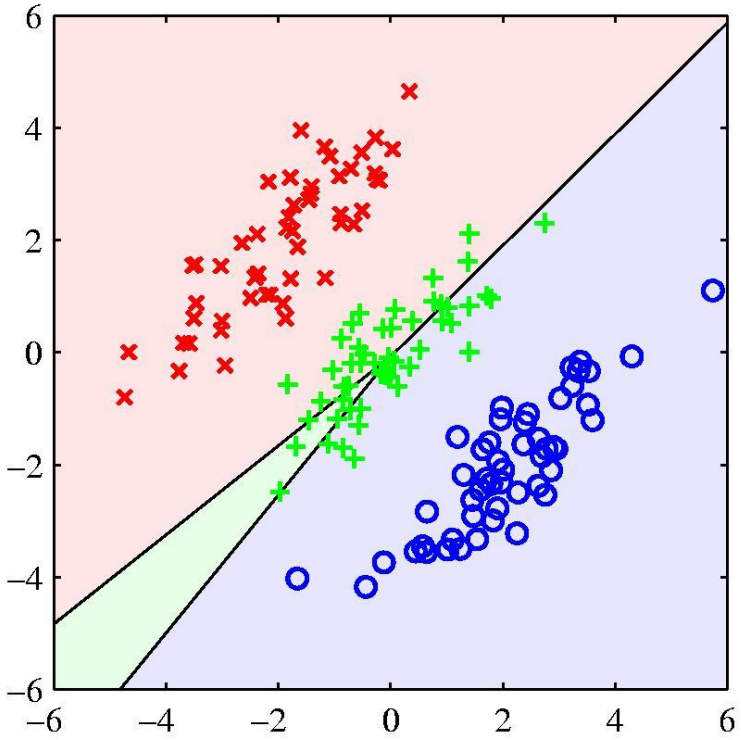
$$\widetilde{\mathbf{W}} = (\widetilde{\mathbf{X}}^{\mathrm{T}}\widetilde{\mathbf{X}})^{-1}\widetilde{\mathbf{X}}^{\mathrm{T}}\mathbf{T} = \widetilde{\mathbf{X}}^{\dagger}\mathbf{T}$$

Problems with Least Square Classification

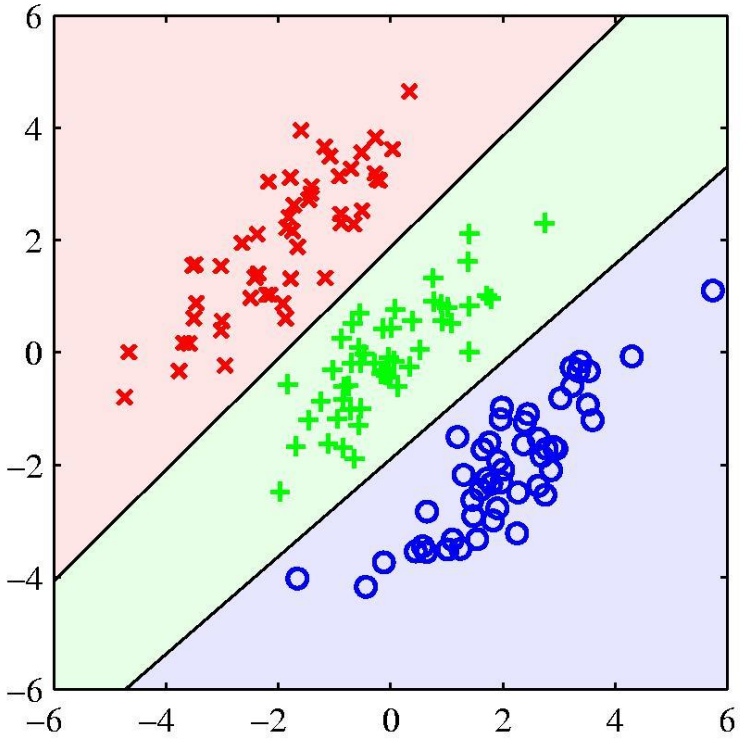


If the right answer is 1 and the model says 1.5, it loses, so it changes the boundary to avoid being “too correct”

Problems with Least Squares Classification



least squares regression



logistic regression

Outlines

- Three Approaches to Linear Classification
- Approach I: Discriminant Functions
- Least Square Classification
- Fisher's Linear Discriminants
- Perceptrons
- Approach II: Probabilistic Generative Models
- Approach III: Probabilistic Discriminative Models
- Bayesian Information Criterion

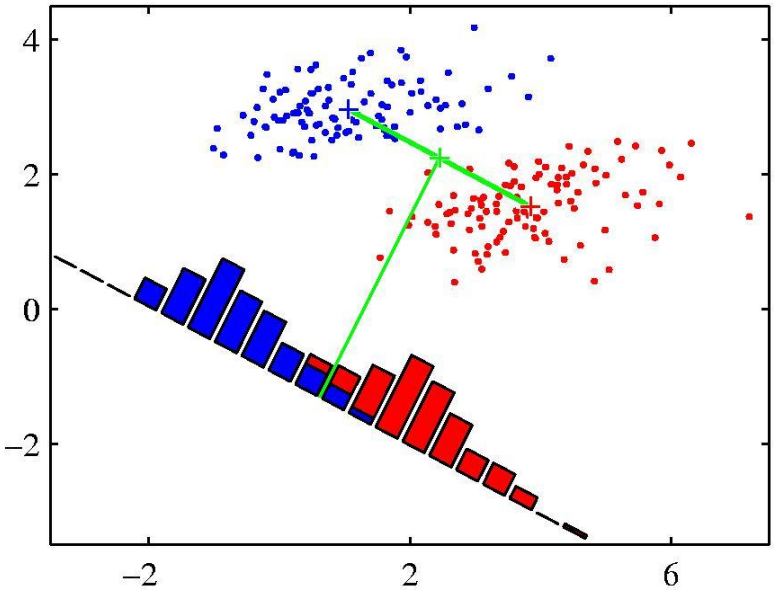
Fisher's Linear Discriminant

- **A simple linear discriminant function** is a projection of the data down to 1-D.
  - ✓ So choose the projection that gives the best separation of the classes. What do we mean by "best separation"?

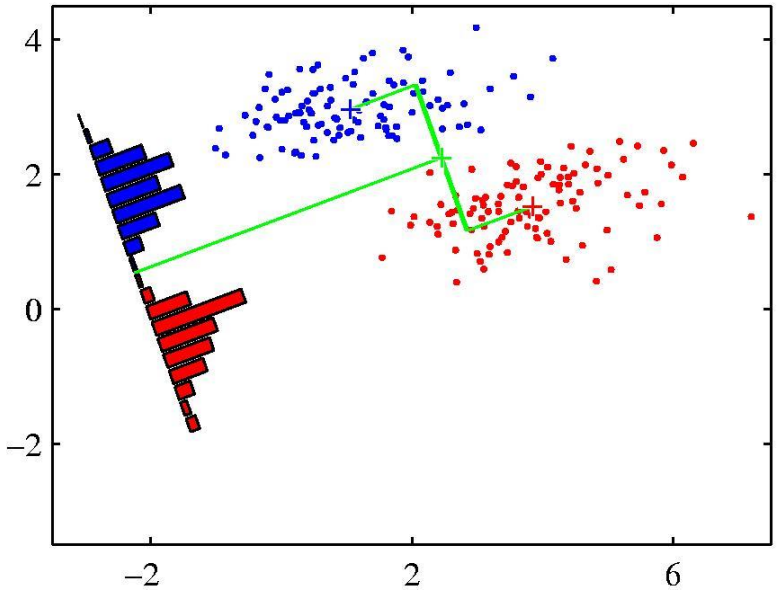
- **An obvious direction to choose** is the direction of the line joining the class means.
  - ✓ But if the main direction of variance in each class is not orthogonal to this line, this will not give good separation (see the next figure).

- **Fisher's method chooses the direction** that maximizes the ratio of between class variance to within class variance.
  - ✓ This is the direction in which the projected points contain the most information about class membership (under Gaussian assumptions)

Fisher's Linear Discriminant Function



When projected onto the line joining the class means, the classes are not well separated.



Fisher chooses a direction that makes the projected classes much tighter, even though their projected means are less far apart.

Fisher's Linear Discriminants (I)

- What linear transformation is best for discrimination?

$$y = \mathbf{w}^T \mathbf{x}$$

$$\mathbf{w} \propto \mathbf{m}_2 - \mathbf{m}_1$$

- The projection onto the vector separating the class means seems sensible:

$$s_1^2 = \sum_{n \varepsilon C_1} (y_n - m_1)^2$$

- But we also want small variance within each class:

$$s_2^2 = \sum_{n \varepsilon C_2} (y_n - m_2)^2$$

- Fisher's objective function is:

$$J(\mathbf{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}$$

← between

← within

Fisher's Linear Discriminants (II)

$$J(\mathbf{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2} = \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}}$$

$$\mathbf{S}_B = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^T$$

$$\mathbf{S}_W = \sum_{n \in C_1} (\mathbf{x}_n - \mathbf{m}_1)(\mathbf{x}_n - \mathbf{m}_1)^T + \sum_{n \in C_2} (\mathbf{x}_n - \mathbf{m}_2)(\mathbf{x}_n - \mathbf{m}_2)^T$$

$$\textit{Optimal solution}: \quad \mathbf{w} \propto \mathbf{S}_W^{-1}(\mathbf{m}_2 - \mathbf{m}_1)$$

$$(\mathbf{w}^{\mathrm{T}} \mathbf{S}_{\mathrm{B}} \mathbf{w}) \mathbf{S}_{\mathrm{W}} \mathbf{w} = (\mathbf{w}^{\mathrm{T}} \mathbf{S}_{\mathrm{W}} \mathbf{w}) \mathbf{S}_{\mathrm{B}} \mathbf{w}$$

Fisher's Linear Discriminants (III)

$$E = \frac{1}{2}\sum_{n=1}^{N}\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + w_0 - t_n\right)^2 \qquad \sum_{n=1}^{N} t_n = N_1\frac{N}{N_1} - N_2\frac{N}{N_2} = 0$$

for two classes

Set its derivatives w.r.t. $\mathbf{w}$ and $\mathbf{w}_0$ to 0, then we will have

$$\sum_{n=1}^{N}\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + w_0 - t_n\right) = 0$$

$$\sum_{n=1}^{N}\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + w_0 - t_n\right)\mathbf{x}_n = 0$$

$$\mathbf{m} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n = \frac{1}{N}(N_1\mathbf{m}_1 + N_2\mathbf{m}_2) \qquad \left(\mathbf{S}_{\mathrm{W}} + \frac{N_1N_2}{N}\mathbf{S}_{\mathrm{B}}\right)\mathbf{w} = N(\mathbf{m}_1 - \mathbf{m}_2)$$

$$w_0 = -\mathbf{w}^{\mathrm{T}}\mathbf{m} \qquad \mathbf{w} \propto \mathbf{S}_{\mathrm{W}}^{-1}(\mathbf{m}_2 - \mathbf{m}_1)$$

K-Case Classification

- **Fisher's linear discriminants** can be extended to K-case classification.

$$J(\mathbf{w}) = \mathrm{Tr}\left\{(\mathbf{W}\mathbf{S}_{\mathrm{W}}\mathbf{W}^{\mathrm{T}})^{-1}(\mathbf{W}\mathbf{S}_{\mathrm{B}}\mathbf{W}^{\mathrm{T}})\right\}$$

Outlines

- Three Approaches to Linear Classification
- Approach I: Discriminant Functions
- Least Square Classification
- Fisher's Linear Discriminants
- Perceptrons
- Approach II: Probabilistic Generative Models
- Approach III: Probabilistic Discriminative Models
- Bayesian Information Criterion

Perceptrons

- **"Perceptrons" describes a whole family of learning machines**
  - ✓ a layer of fixed non-linear basis functions followed by a simple linear discriminant function.
  - ✓ introduced in the late 1950's
  - ✓ a simple online learning procedure

$$y(\mathbf{x}) = f\left(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x})\right) \qquad f(a) = \begin{cases} +1, & a \geqslant 0 \\ -1, & a < 0 \end{cases}$$

Perceptron Training

Perceptron criterion:

$$E_{\mathrm{P}}(\mathbf{w}) = -\sum_{n \in \mathcal{M}} \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}_n t_n$$

$t_n = \{1, -1\}$:

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E_{\mathrm{P}}(\mathbf{w}) = \mathbf{w}^{(\tau)} + \eta \boldsymbol{\phi}_n t_n$$

Set the learning rate η as 1, then we will have

$$-\mathbf{w}^{(\tau+1)\mathrm{T}} \boldsymbol{\phi}_n t_n = -\mathbf{w}^{(\tau)\mathrm{T}} \boldsymbol{\phi}_n t_n - (\boldsymbol{\phi}_n t_n)^{\mathrm{T}} \boldsymbol{\phi}_n t_n < -\mathbf{w}^{(\tau)\mathrm{T}} \boldsymbol{\phi}_n t_n$$

which indicates the convergence of perceptron training

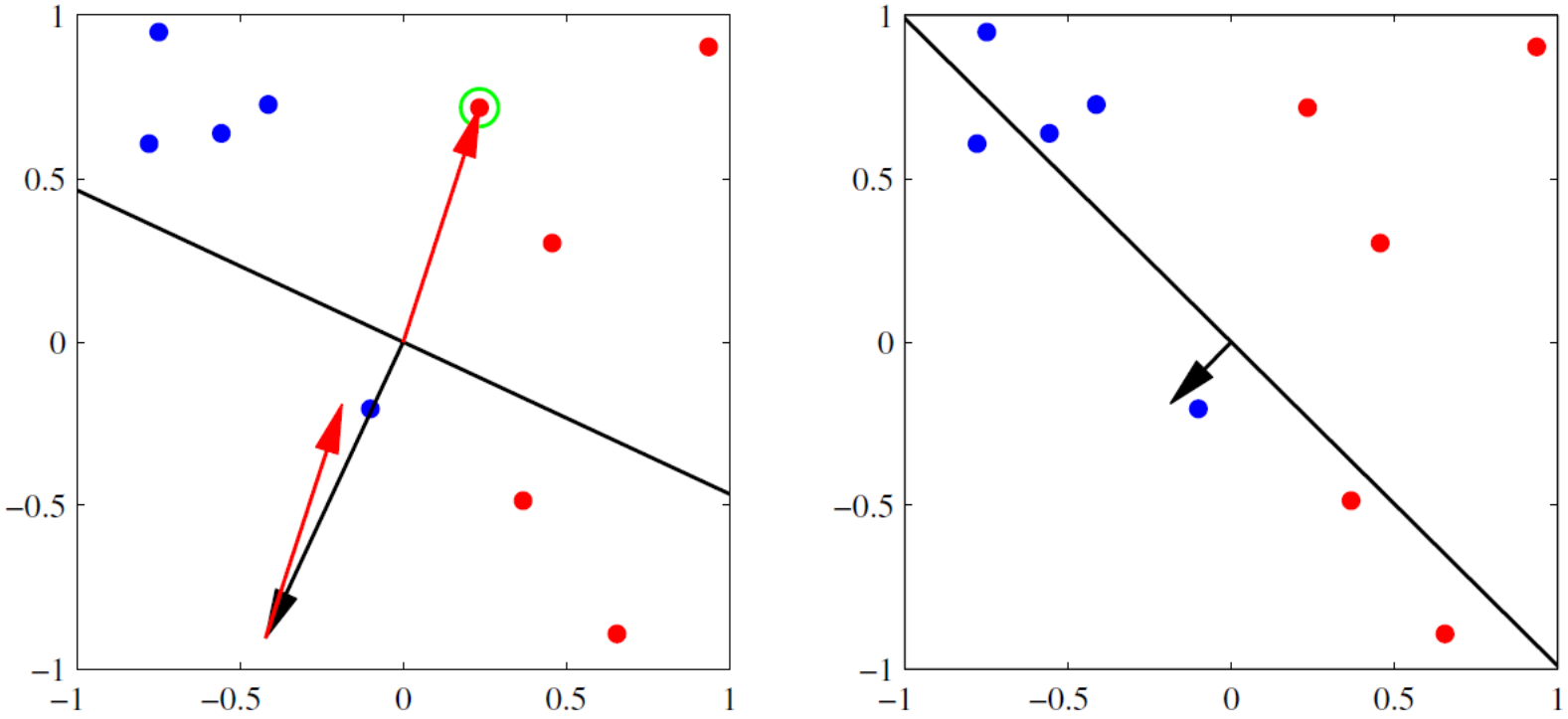
Simplified Perceptron Training

- **Pick training cases using any policy that ensures that every training case will keep getting picked**
  - ✓ If the output is correct, leave its weights alone.
  - ✓ If the output is -1 but should be 1, add the feature vector to the weight vector.
  - ✓ If the output is 1 but should be -1, subtract the feature vector from the weight vector

$$\boldsymbol{w}^{new} = \boldsymbol{w}^{old} - 0.5(y_n - tn)\boldsymbol{x}_n$$

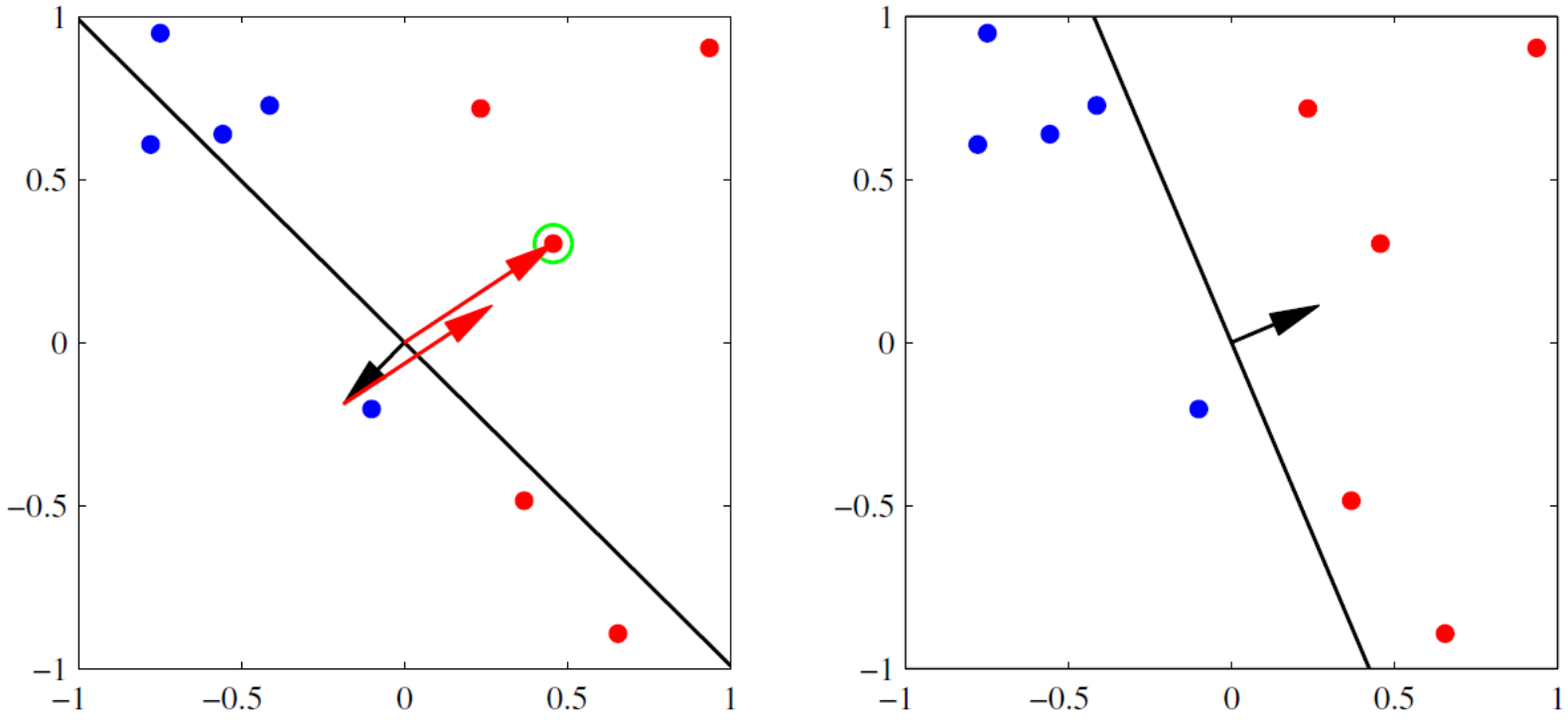
- **This is guaranteed to find a set of weights that gets the right answer on the whole training set if any such a set exists. There is no need to choose a learning rate**.

Perceptron Training Procedure



decision boundary: black line; w: black arrow; mismatching data: green circle
Δw: red arrow;

Perceptron Training Procedure



decision boundary: black line; w: black arrow; mismatching data: green circle
Δw: red arrow;

What Perceptrons Cannot Learn

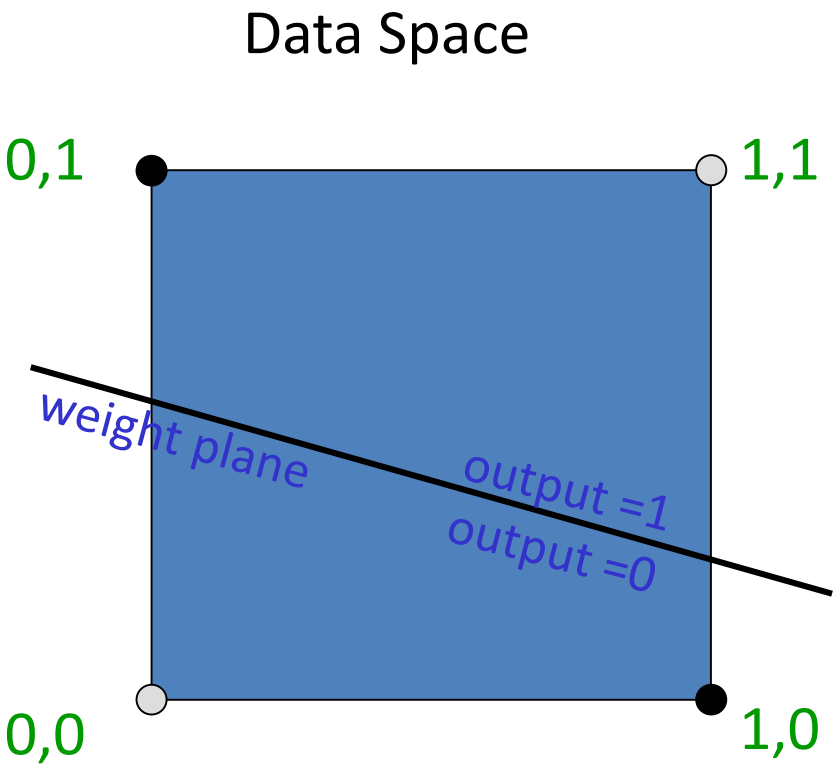
The adaptive part of a perceptron cannot even tell if two single bit features have the same value!

Same: (1,1) → 1; (0,0) → 1
Different: (1,0) → 0; (0,1) → 0

The four feature-output pairs give four inequalities that are impossible to satisfy:

$$w_1 + w_2 \geq \theta, \quad 0 \geq \theta$$

$$w_1 < \theta, \qquad w_2 < \theta$$

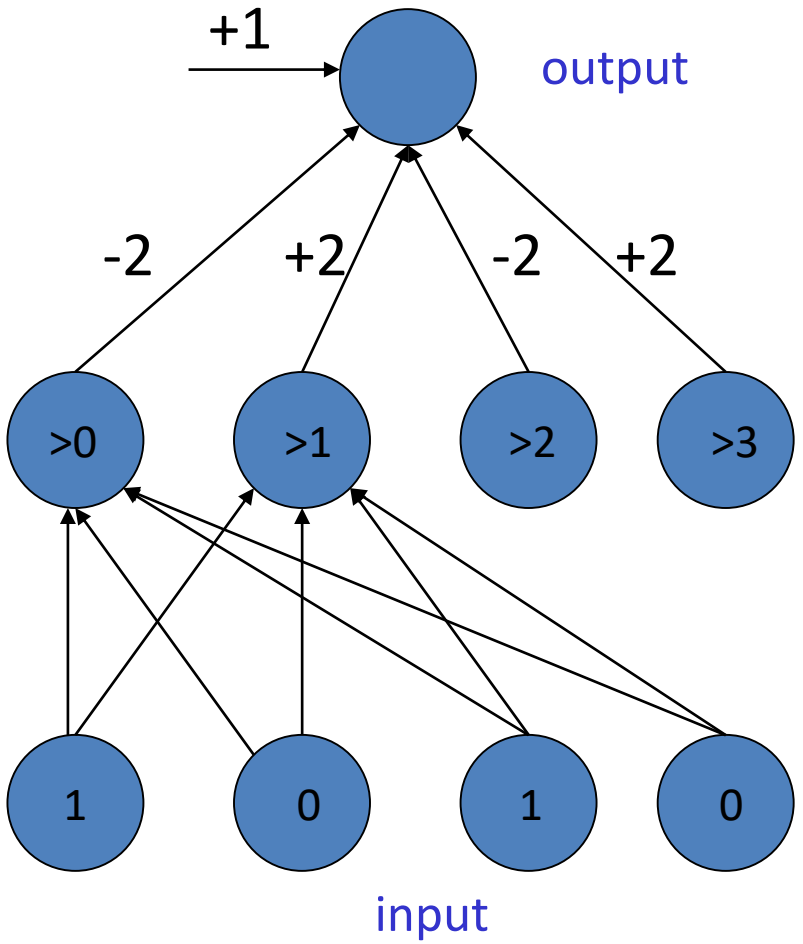


The positive and negative cases cannot be separated by a plane

The N-bit Even Parity Task

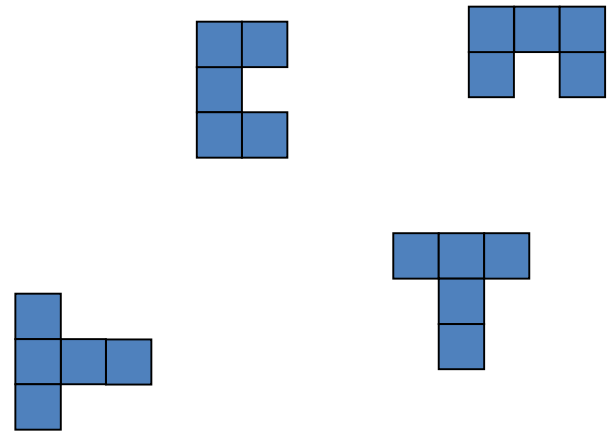
- There is a simple solution that requires N hidden units.
  - ✓ Each hidden unit computes whether more than M of the inputs are on.
  - ✓ This is a linearly separable problem.
- There are many variants of this solution.
  - ✓ It can be learned.
  - ✓ It generalizes well if:

$$2^N >> N^2$$

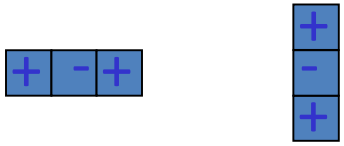


Distinguishing Patterns

- What kind of features are required to distinguish two different patterns of 5 pixels independent of position and orientation?
  - ✓ Do we need to replicate T and C templates across all positions and orientations?
  - ✓ Looking at pairs of pixels will not work
  - ✓ Looking at triples will work if we assume that each input image only contains one object.



Replicate the following two feature detectors in all positions



If any of these equal their threshold of 2, it's a C. If not, it's a T.