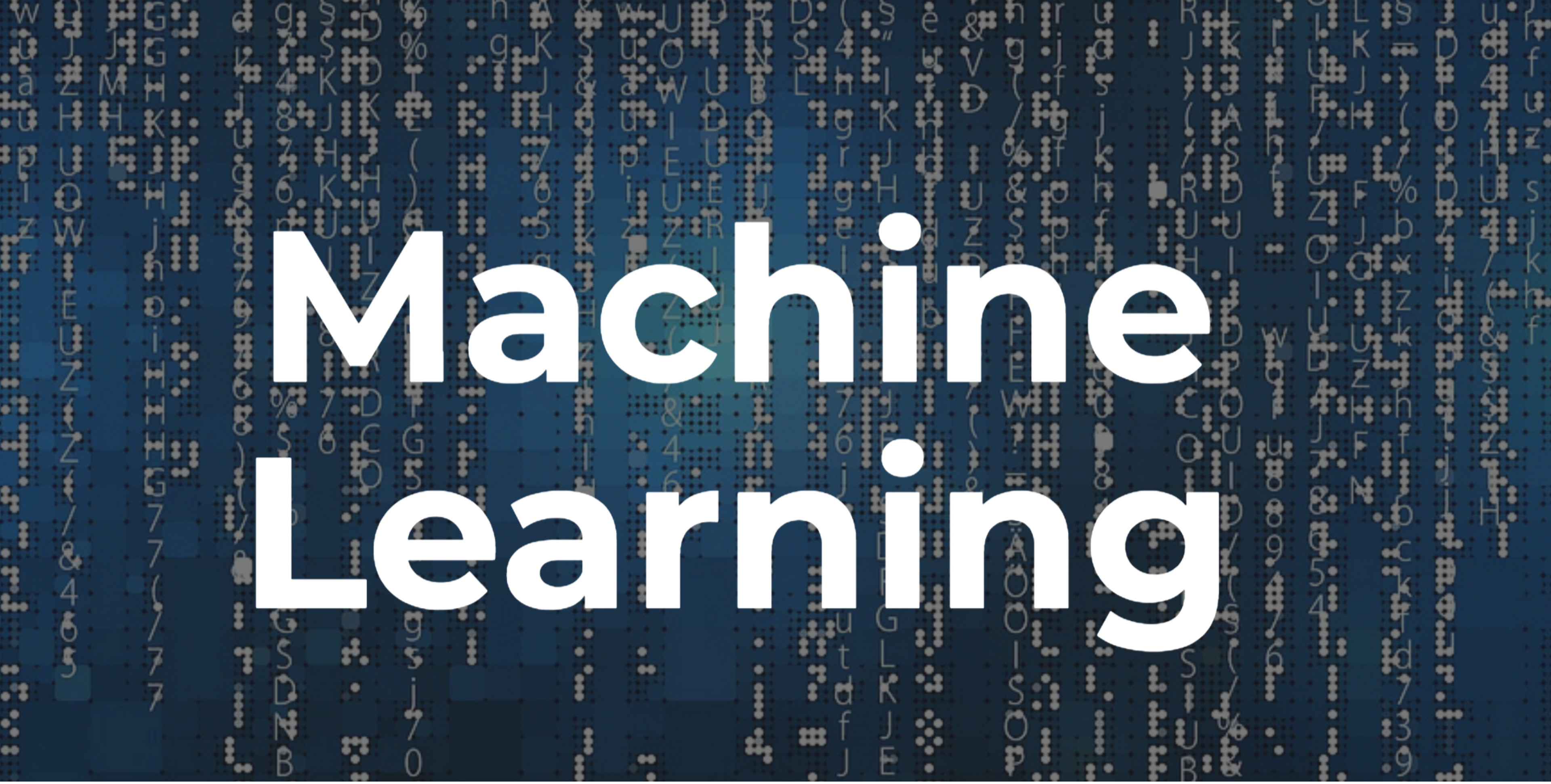


Machine Learning

CSCI 4622 Fall 2019

Prof. Claire Monteleoni

Today: Lecture 4

- Intro to Linear Classification
  - Perceptron algorithm

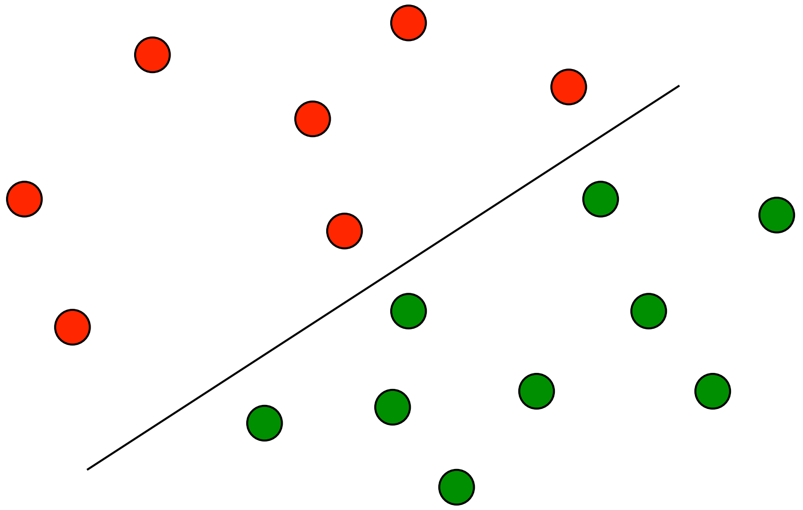
With credit to S. Dasgupta and T. Jaakkola

Linear classification

Given labeled data points, find a linear separator.

Describes the data

Generalizes well



Line in 2-d

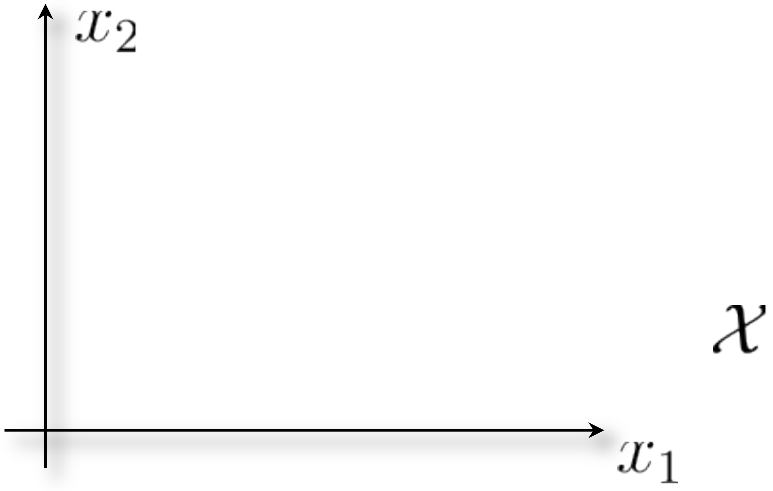
Plane in 3-d

Hyperplane in ***n***-d

Linear classifiers

- A linear classifier (through origin) with parameters $\underline{\theta}$ divides the space into positive and negative halves

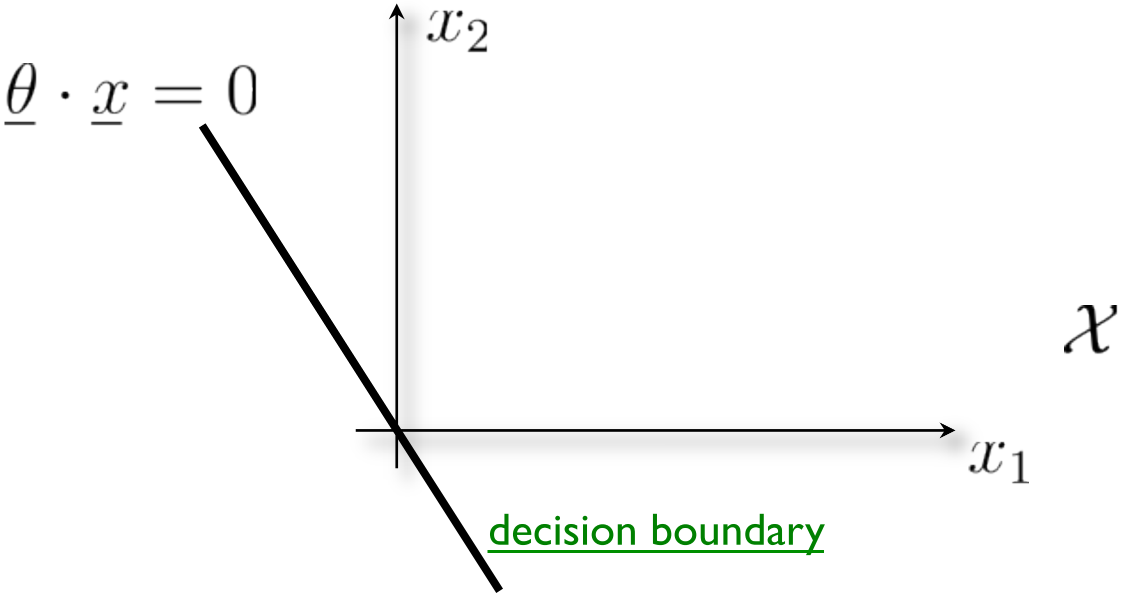
$$
\begin{aligned}
f(\underline{x};\underline{\theta}) &= \operatorname{sign}\big(\,\underline{\theta}\cdot\underline{x}\,\big) = \operatorname{sign}\big(\,\theta_1 x_1 + \ldots + \theta_d x_d\,\big) \\
&= \begin{cases} +1, & \text{if } \underline{\theta}\cdot\underline{x} > 0 \\ -1, & \text{if } \underline{\theta}\cdot\underline{x} \le 0 \end{cases}
\end{aligned}
$$



Linear classifiers

- A linear classifier (through origin) with parameters $\underline{\theta}$ divides the space into positive and negative halves

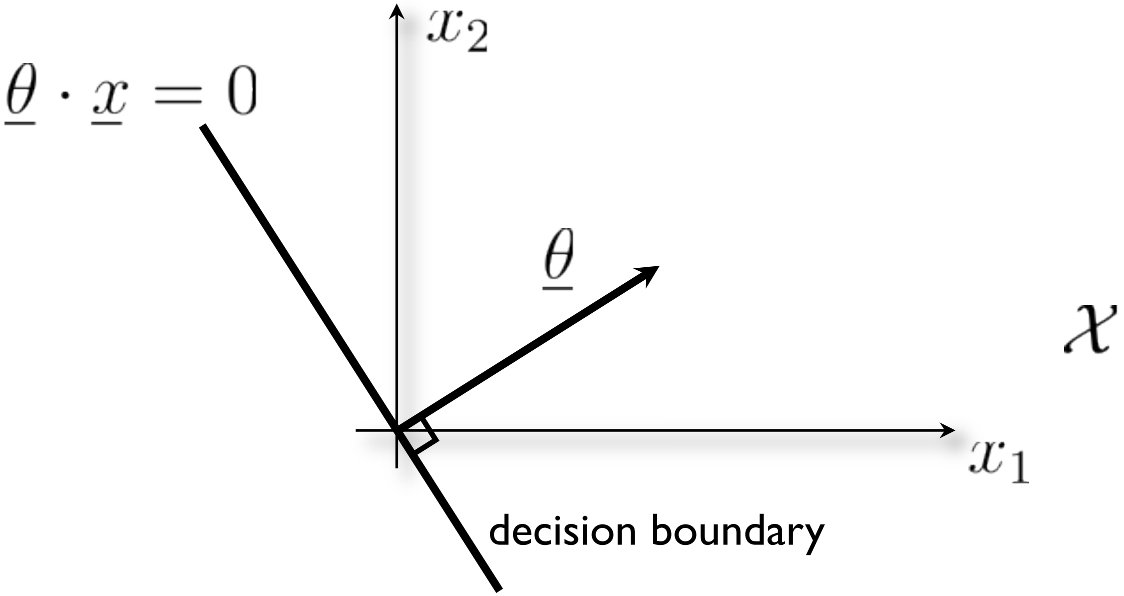
$$
\begin{aligned}
f(\underline{x};\underline{\theta}) &= \operatorname{sign}\big(\,\underline{\theta}\cdot\underline{x}\,\big) = \operatorname{sign}\big(\,\theta_1 x_1 + \ldots + \theta_d x_d\,\big) \\
&= \begin{cases} +1, & \text{if } \underline{\theta}\cdot\underline{x} > 0 \\ -1, & \text{if } \underline{\theta}\cdot\underline{x} \le 0 \end{cases}
\end{aligned}
$$



Linear classifiers

- A linear classifier (through origin) with parameters $\underline{\theta}$ divides the space into positive and negative halves

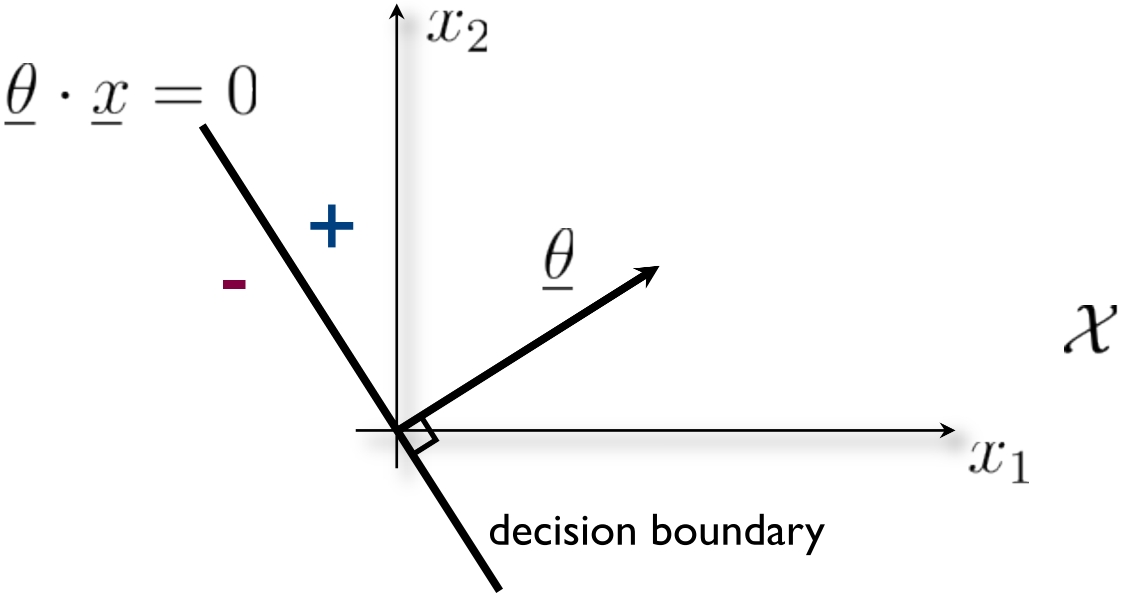
$$\begin{aligned} f(\underline{x};\underline{\theta}) &= \operatorname{sign}\big(\,\underline{\theta}\cdot\underline{x}\,\big) = \operatorname{sign}\big(\,\theta_1 x_1 + \ldots + \theta_d x_d\,\big) \\ &= \begin{cases} +1, & \text{if } \underline{\theta}\cdot\underline{x} > 0 \\ -1, & \text{if } \underline{\theta}\cdot\underline{x} \leq 0 \end{cases} \end{aligned}$$



Linear classifiers

- A linear classifier (through origin) with parameters $\underline{\theta}$ divides the space into positive and negative halves

$$\begin{aligned} f(\underline{x};\underline{\theta}) &= \operatorname{sign}\big(\,\underline{\theta}\cdot\underline{x}\,\big) = \operatorname{sign}\big(\,\theta_1 x_1 + \ldots + \theta_d x_d\,\big) \\ &= \begin{cases} +1, & \text{if } \underline{\theta}\cdot\underline{x} > 0 \\ -1, & \text{if } \underline{\theta}\cdot\underline{x} \leq 0 \end{cases} \end{aligned}$$



Linear classifiers

- The **magnitude** of $(\vec{\theta} \cdot \vec{x})$ is the distance between *x* and the decision boundary.

- The **sign** of $(\vec{\theta} \cdot \vec{x})$ is the classification label that $\vec{\theta}$ assigns to *x*.

- Note: while we will refer to $\vec{\theta}$ as the classifier, the function that classifies *x* is:

$$f(\vec{x}) = \mathrm{SIGN}(\vec{\theta} \cdot \vec{x})$$

(1958)
F. Rosenblatt

The perceptron: a probabilistic model for information storage and organization in the brain

Psychological Review 65:386–408

If we are eventually to understand the capability of higher organisms for perceptual recognition, generalization, recall, and thinking, we must first have answers to three fundamental questions:

1. How is information about the physical world sensed, or detected, by the biological system?
2. In what form is information stored, or remembered?
3. How does information contained in storage, or in memory, influence recognition and behavior?

The first of these questions is in the province of sensory physiology, and is the only one for which appreciable understanding has been achieved. This article will be concerned primarily with the second and third questions, which are still subject to a vast amount of speculation, and where the few relevant facts currently supplied by neurophysiology have not yet been integrated into an acceptable theory.

With regard to the second question

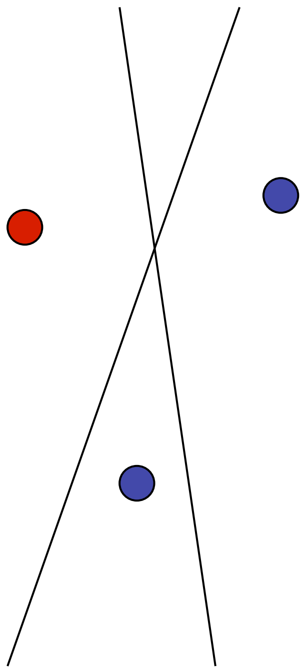
and the stored pattern. According to this hypothesis, if one understood the code or "wiring diagram" of the nervous system, one should, in principle, be able to discover exactly what an organism remembers by reconstructing the original sensory patterns from the "memory traces" which they have left, much as we might develop a photographic negative, or translate the pattern of electrical charges in the "memory" of a digital computer. This hypothesis is appealing in its simplicity and ready intelligibility, and a large family of theoretical brain models has been developed around the idea of a coded, representational memory (**2**, **3**, **9**, **14**). The alternative approach, which stems from the tradition of British empiricism, hazards the guess that the images of stimuli may never really be recorded at all, and that the central nervous system simply acts as an intricate switching network, where retention takes the form of new connections, or pathways, between centers of activity. In many of the more recent developments of

Learning from one data point at a time

Instead of receiving a batch of data ahead of time,
Learner receives one data point at a time.

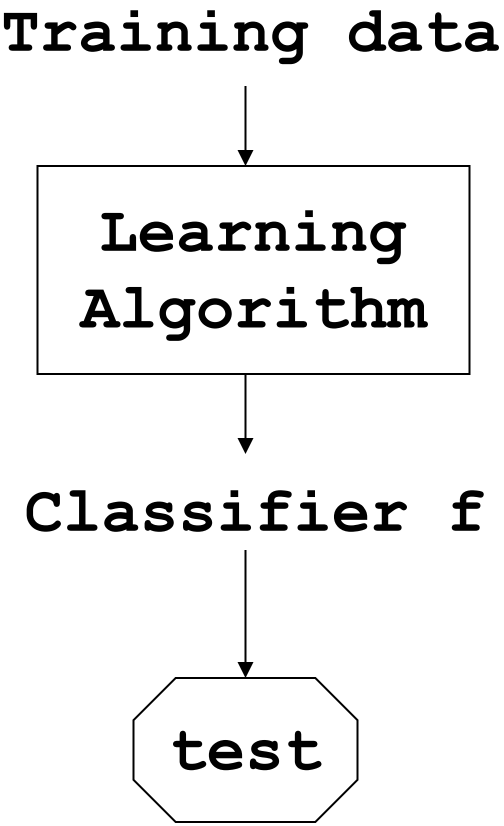
Can also cycle through a batch of data this way.

E.g. learning a linear classifier:



Batch vs. Online Learning

Batch learning



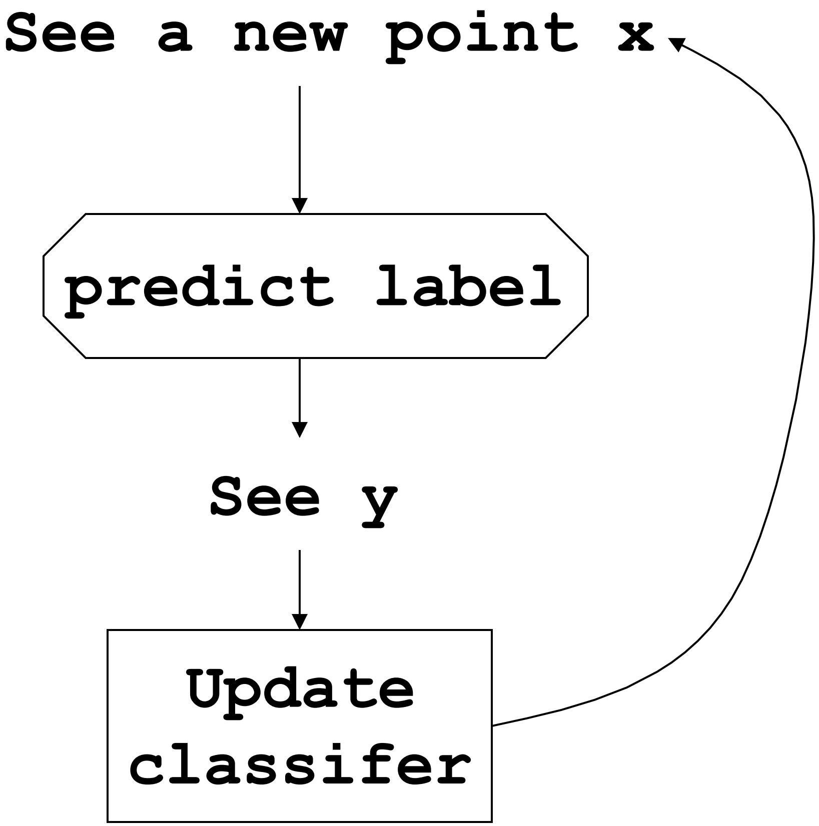
Training data

Learning Algorithm

Classifier f

test

Online learning



See a new point x

predict label

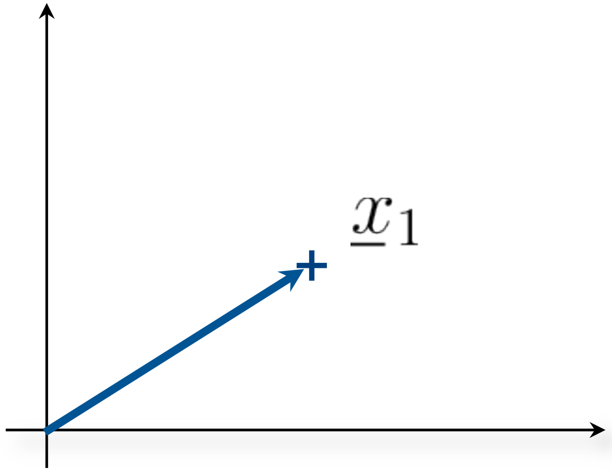
See y

Update classifer

Perceptron algorithm

- Iterative updates based on mistakes

$$\underline{\theta}_0 = 0$$

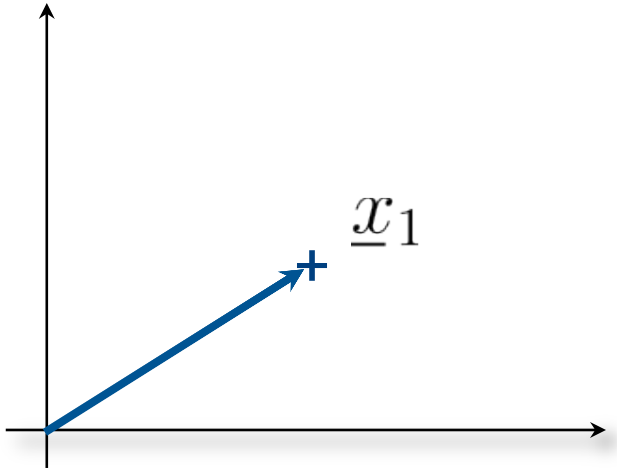


Perceptron algorithm

- Iterative updates based on mistakes

$$\underline{\theta}_0 = 0$$

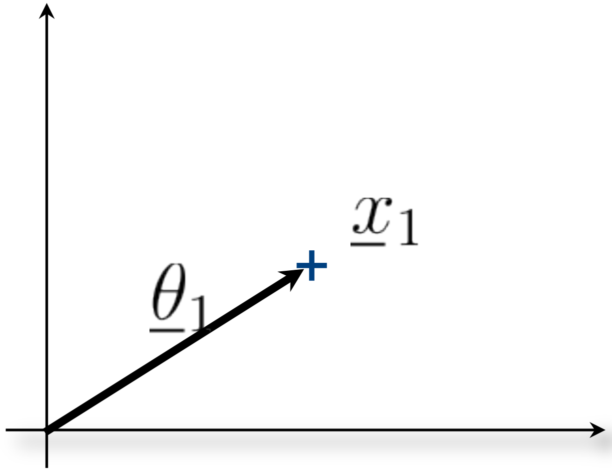
$$\underline{\theta}_1 = \underline{\theta}_0 + 1\,\underline{x}_1$$



Perceptron algorithm

- Iterative updates based on mistakes

$$\underline{\theta}_0 = 0$$
$$\underline{\theta}_1 = \underline{\theta}_0 + 1\,\underline{x}_1$$

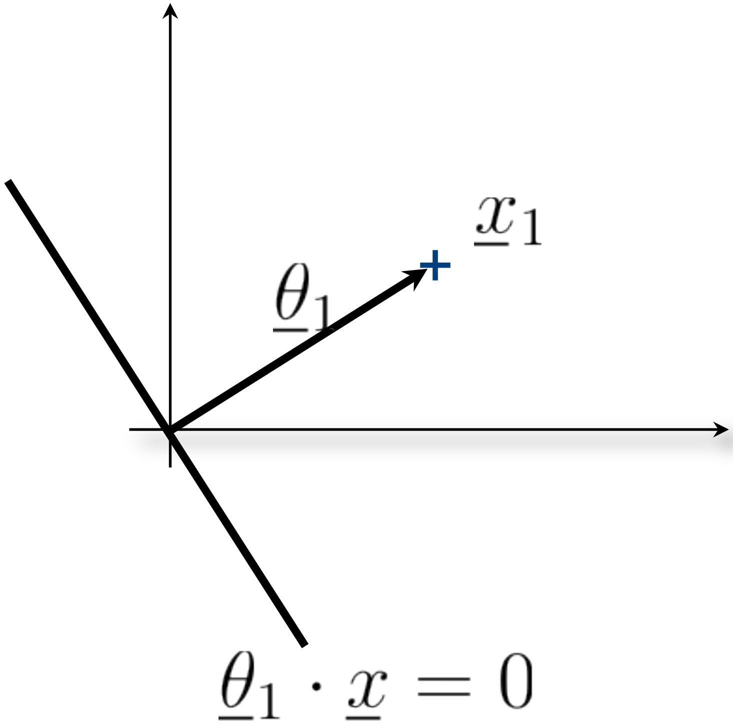


Perceptron algorithm

- Iterative updates based on mistakes

$$\underline{\theta}_0 = 0$$

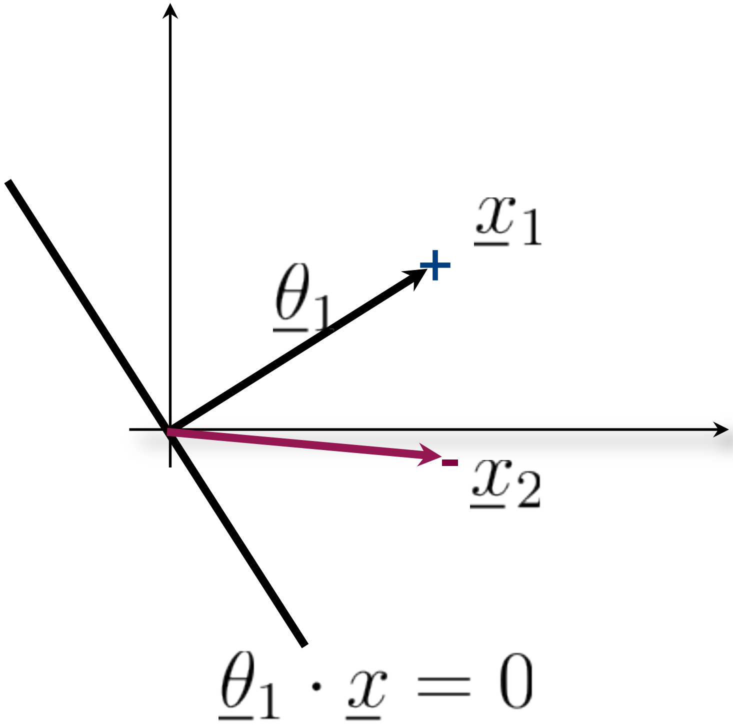
$$\underline{\theta}_1 = \underline{\theta}_0 + 1\,\underline{x}_1$$



Perceptron algorithm

- Iterative updates based on mistakes

$$\underline{\theta}_0 = 0$$
$$\underline{\theta}_1 = \underline{\theta}_0 + 1\,\underline{x}_1$$



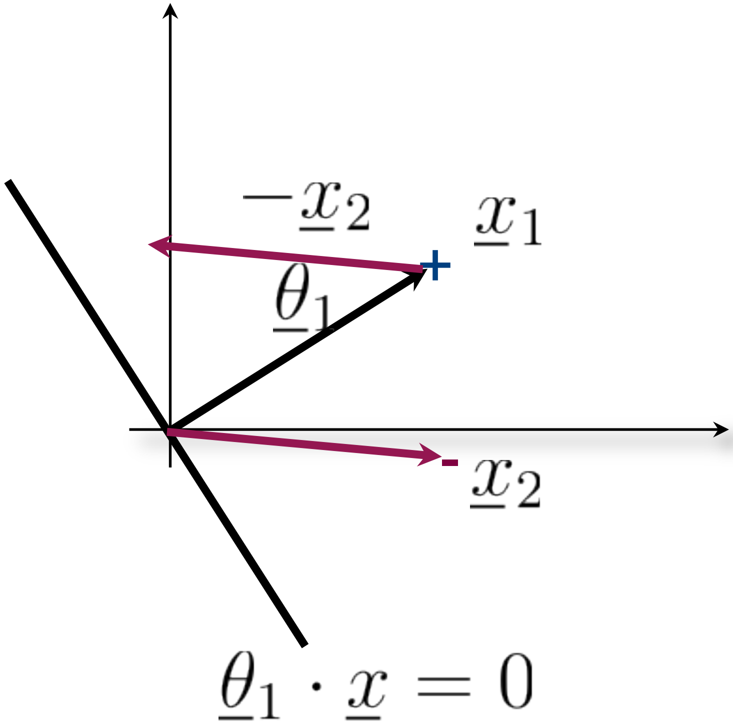
Perceptron algorithm

- Iterative updates based on mistakes

$$\underline{\theta}_0 = 0$$

$$\underline{\theta}_1 = \underline{\theta}_0 + 1\,\underline{x}_1$$

$$\underline{\theta}_2 = \underline{\theta}_1 + (-1)\,\underline{x}_2$$



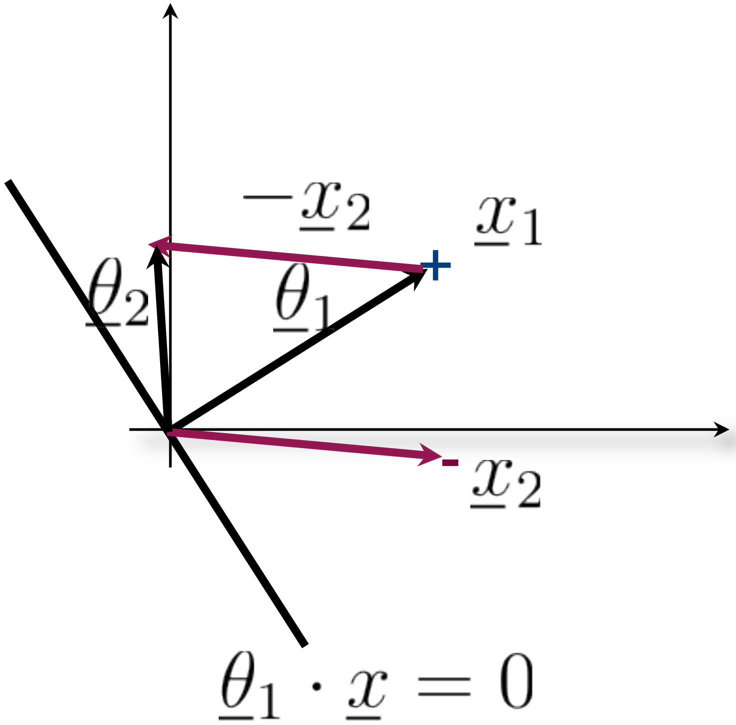
Perceptron algorithm

- Iterative updates based on mistakes

$$\underline{\theta}_0 = 0$$

$$\underline{\theta}_1 = \underline{\theta}_0 + 1\,\underline{x}_1$$

$$\underline{\theta}_2 = \underline{\theta}_1 + (-1)\,\underline{x}_2$$



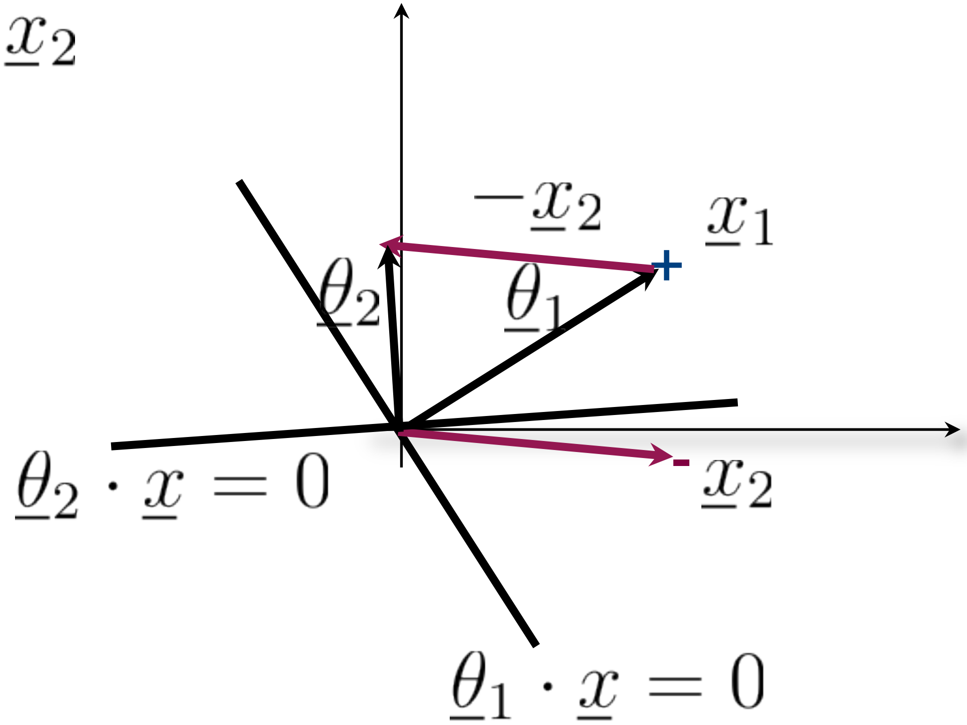
Perceptron algorithm

- Iterative updates based on mistakes

$$\underline{\theta}_0 = 0$$

$$\underline{\theta}_1 = \underline{\theta}_0 + 1\,\underline{x}_1$$

$$\underline{\theta}_2 = \underline{\theta}_1 + (-1)\,\underline{x}_2$$



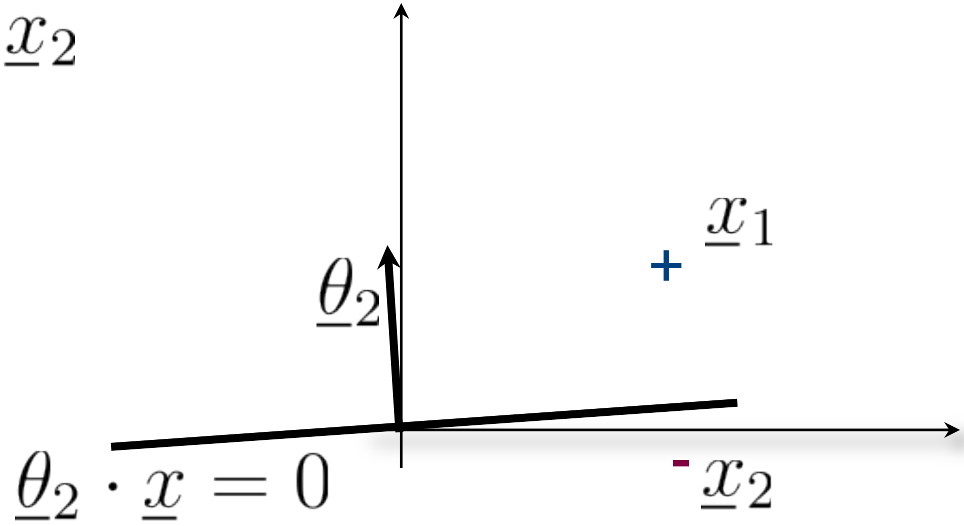
Perceptron algorithm

- Iterative updates based on mistakes

$$\underline{\theta}_0 = 0$$

$$\underline{\theta}_1 = \underline{\theta}_0 + 1\,\underline{x}_1$$

$$\underline{\theta}_2 = \underline{\theta}_1 + (-1)\,\underline{x}_2$$

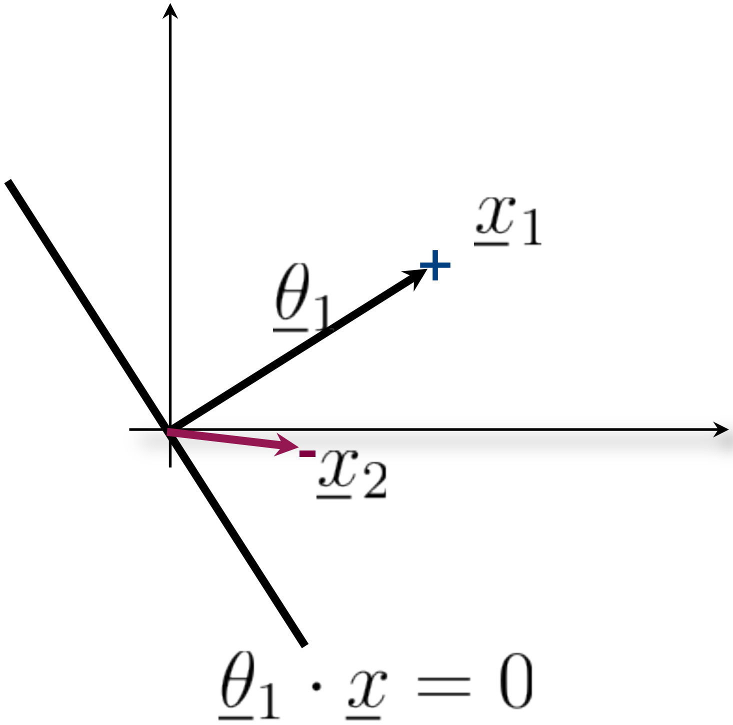


Perceptron algorithm (take 2)

- Iterative updates based on mistakes

$$\underline{\theta}_0 = 0$$

$$\underline{\theta}_1 = \underline{\theta}_0 + 1\,\underline{x}_1$$



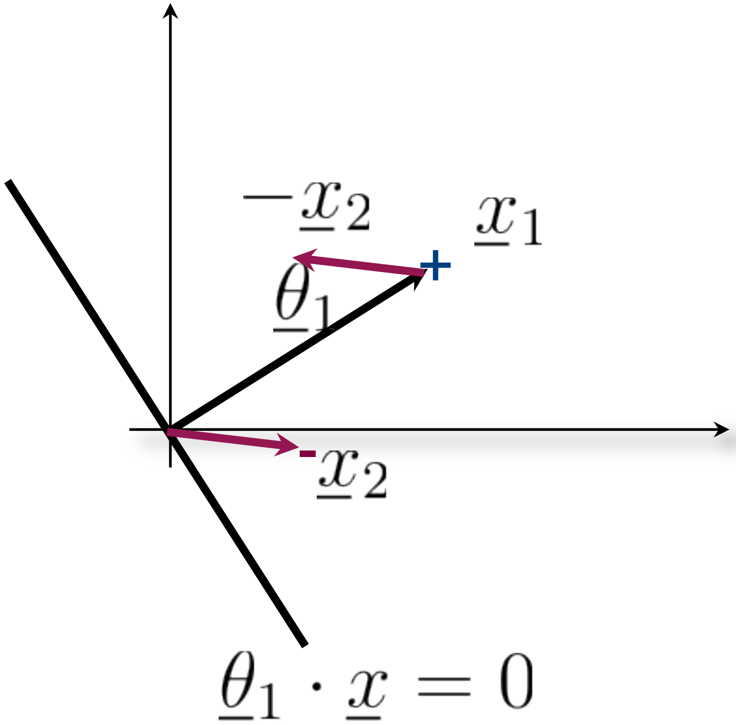
Perceptron algorithm (take 2)

- Iterative updates based on mistakes

$$\underline{\theta}_0 = 0$$

$$\underline{\theta}_1 = \underline{\theta}_0 + 1\, \underline{x}_1$$

$$\underline{\theta}_2 = \underline{\theta}_1 + (-1)\, \underline{x}_2$$



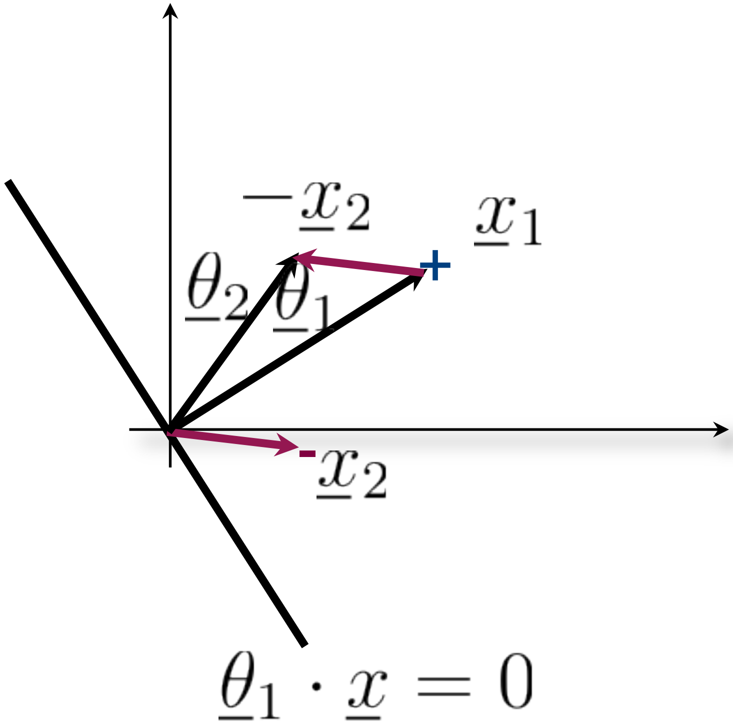
Perceptron algorithm (take 2)

- Iterative updates based on mistakes

$$\underline{\theta}_0 = 0$$

$$\underline{\theta}_1 = \underline{\theta}_0 + 1\,\underline{x}_1$$

$$\underline{\theta}_2 = \underline{\theta}_1 + (-1)\,\underline{x}_2$$



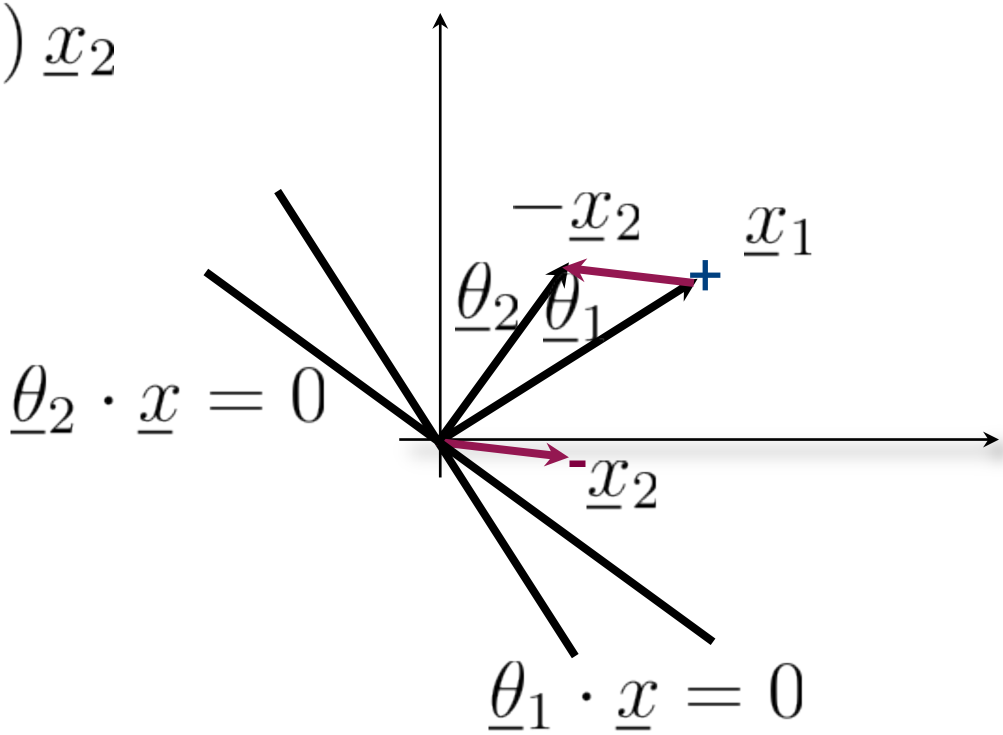
Perceptron algorithm (take 2)

- Iterative updates based on mistakes

$$\underline{\theta}_0 = 0$$

$$\underline{\theta}_1 = \underline{\theta}_0 + 1\,\underline{x}_1$$

$$\underline{\theta}_2 = \underline{\theta}_1 + (-1)\,\underline{x}_2$$



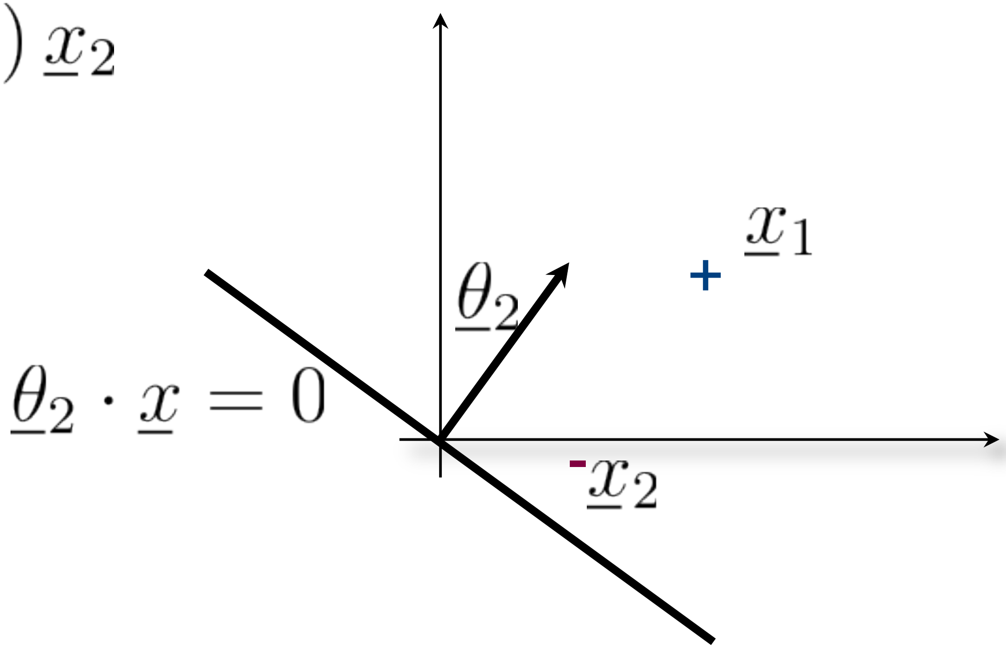
Perceptron algorithm (take 2)

- Iterative updates based on mistakes

$$\underline{\theta}_0 = 0$$

$$\underline{\theta}_1 = \underline{\theta}_0 + 1\,\underline{x}_1$$

$$\underline{\theta}_2 = \underline{\theta}_1 + (-1)\,\underline{x}_2$$

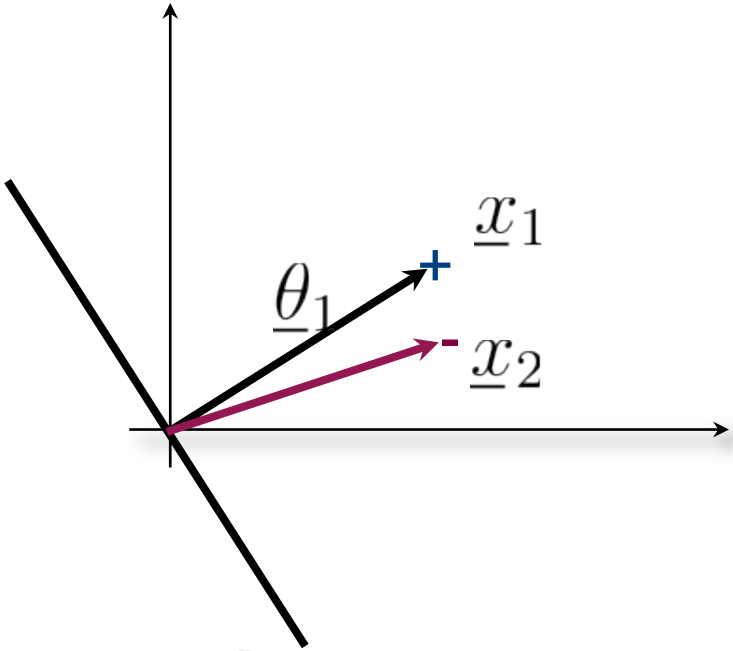


Perceptron algorithm (take 3)

- Iterative updates based on mistakes

$$\underline{\theta}_0 = 0$$

$$\underline{\theta}_1 = \underline{\theta}_0 + 1\,\underline{x}_1$$



$$\underline{\theta}_1 \cdot \underline{x} = 0$$

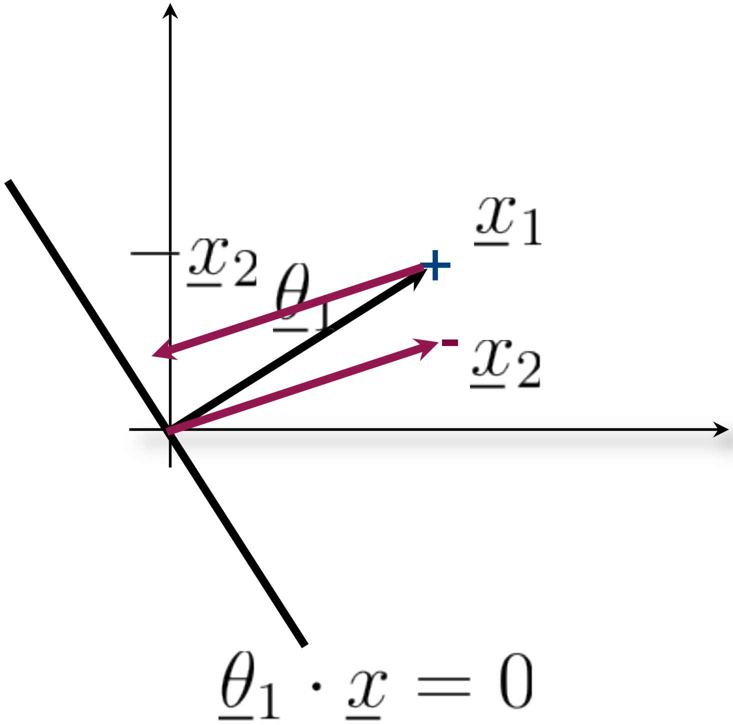
Perceptron algorithm (take 3)

- Iterative updates based on mistakes

$$\underline{\theta}_0 = 0$$

$$\underline{\theta}_1 = \underline{\theta}_0 + 1\,\underline{x}_1$$

$$\underline{\theta}_2 = \underline{\theta}_1 + (-1)\,\underline{x}_2$$



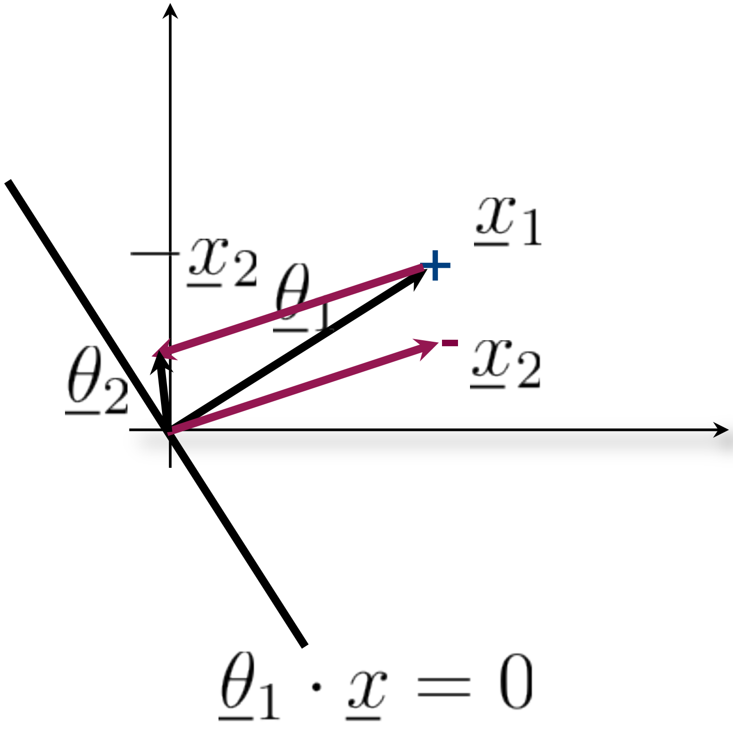
Perceptron algorithm (take 3)

- Iterative updates based on mistakes

$$\underline{\theta}_0 = 0$$

$$\underline{\theta}_1 = \underline{\theta}_0 + 1\,\underline{x}_1$$

$$\underline{\theta}_2 = \underline{\theta}_1 + (-1)\,\underline{x}_2$$



$$\underline{\theta}_1 \cdot \underline{x} = 0$$

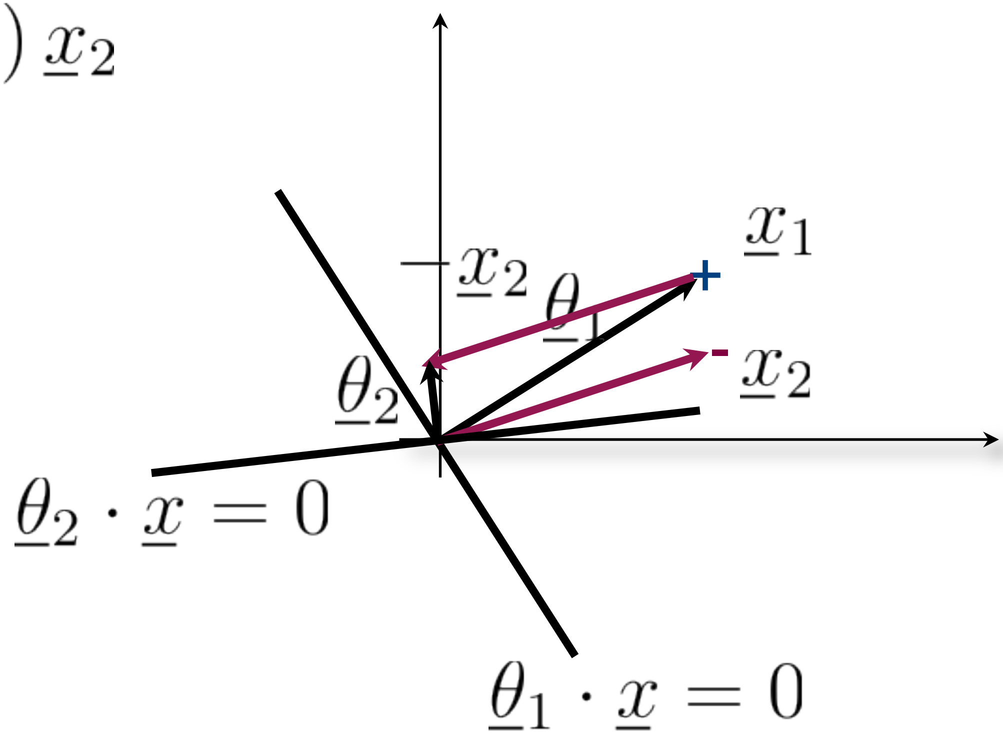
Perceptron algorithm (take 3)

- Iterative updates based on mistakes

$$\underline{\theta}_0 = 0$$

$$\underline{\theta}_1 = \underline{\theta}_0 + 1\,\underline{x}_1$$

$$\underline{\theta}_2 = \underline{\theta}_1 + (-1)\,\underline{x}_2$$



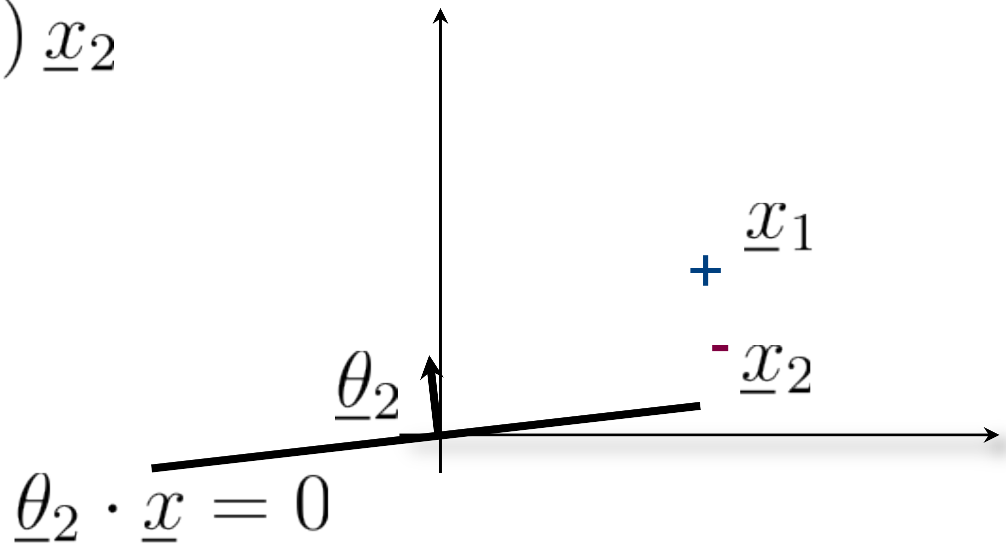
Perceptron algorithm (take 3)

- Iterative updates based on mistakes

$$\underline{\theta}_0 = 0$$

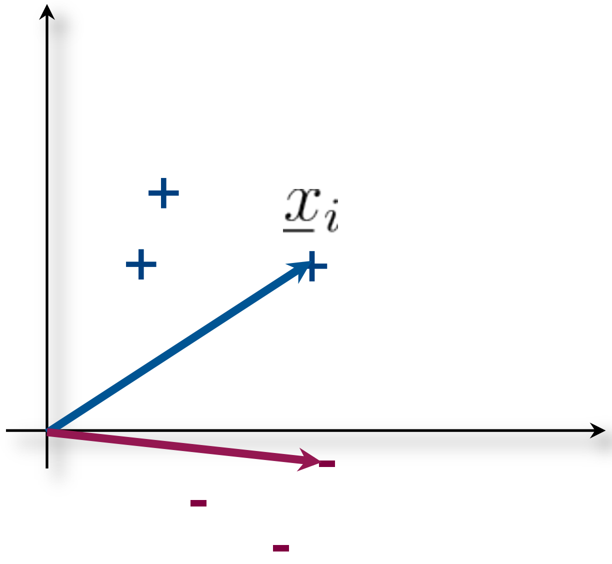
$$\underline{\theta}_1 = \underline{\theta}_0 + 1\,\underline{x}_1$$

$$\underline{\theta}_2 = \underline{\theta}_1 + (-1)\,\underline{x}_2$$



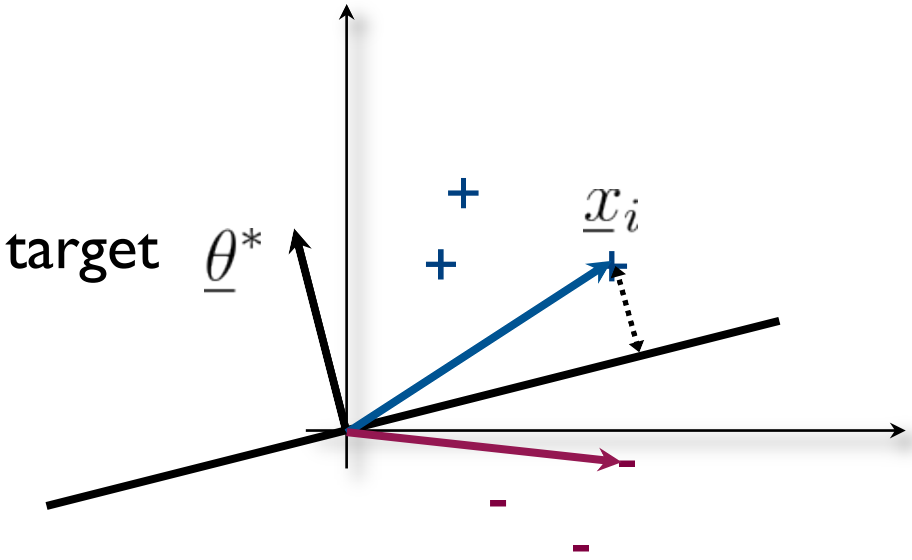
"Margin"

- We can get a handle on convergence by assuming that there exists a target classifier, θ*, with good properties
- One such property is margin, i.e., how close the separating boundary is to the points



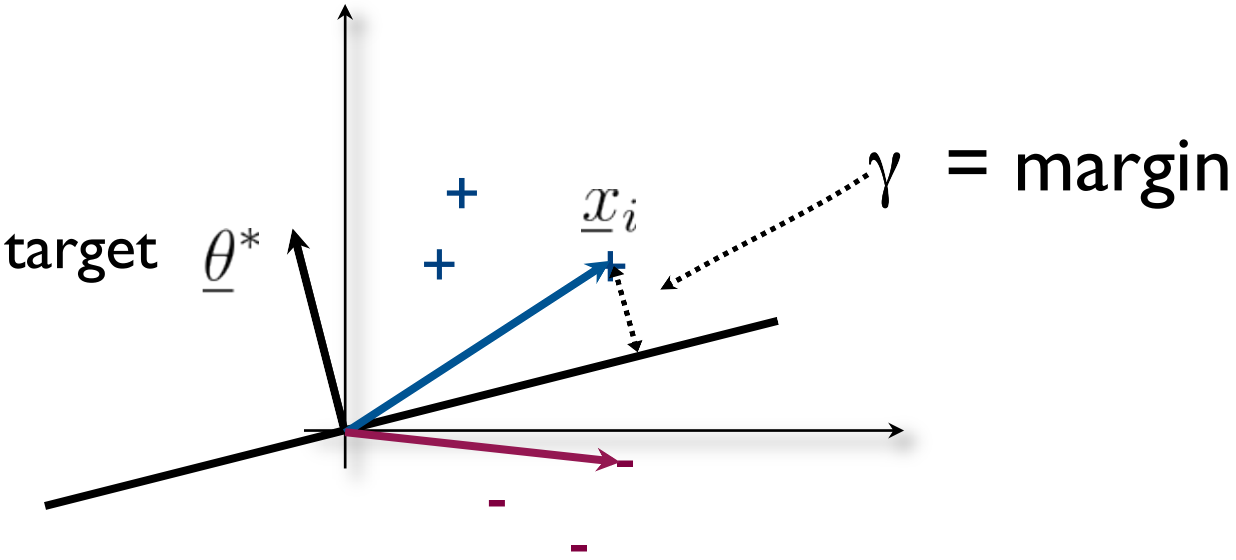
"Margin"

- We can get a handle on convergence by assuming that there exists a target classifier θ* with good properties
- One such property is margin, i.e., how close the separating boundary is to the points



“Margin”

- We can get a handle on convergence by assuming that there exists a target classifier θ* with good properties
- One such property is margin, i.e., how close the separating boundary is to the points



Perceptron

Perceptron: If $y_t(v_t \, . \, x_t \; - \; \tau) < 0$ Filtering rule

$v_{t+1} = v_t + \eta \, y_t \, x_t$ Update step

NOTE: Additive updates. $X=R^d$. Algorithm credited to [Rosenblatt '58].

We will now assume $\eta= 1, \tau= 0$.

Perceptron: If $y_t(v_t \, . \, X_t) < 0$ Filtering rule

$v_{t+1} = v_t + y_t \, x_t$ Update step

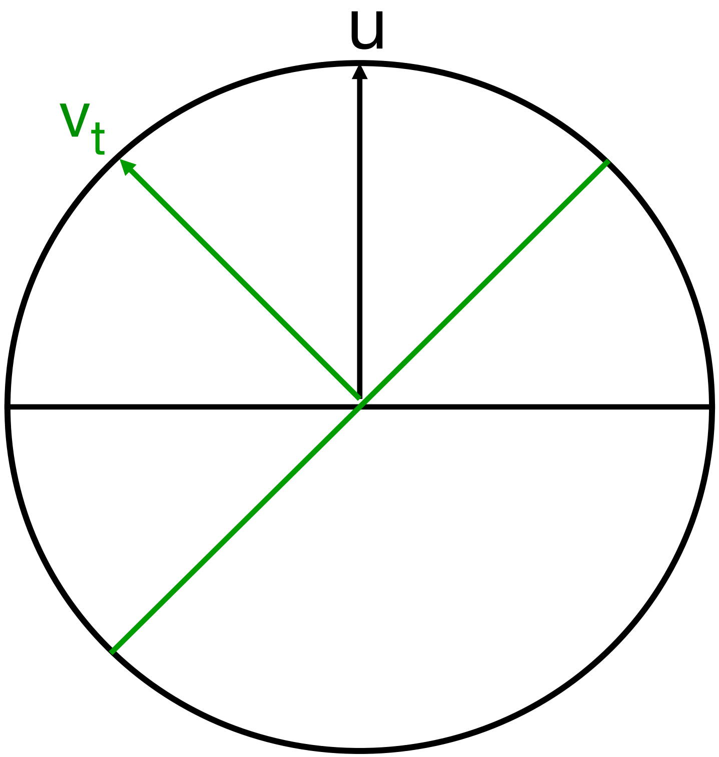
Note: here we use the notation v = θ, u = θ*

Problem framework

$$x_t \in \mathbb{R}^d, \ \|x\| \leq R, \ y_t \in \{-1, +1\}$$

Separability: there exists some perfect classifier, u, such that:

$$u : y_t(u \cdot x_t) > 0 \ \ \forall t, \ \ \|u\| = 1$$



Assume u is through origin.

→ Always possible, by increasing dimension by 1.

Current hypothesis: v_t

(Called θ on previous slides)

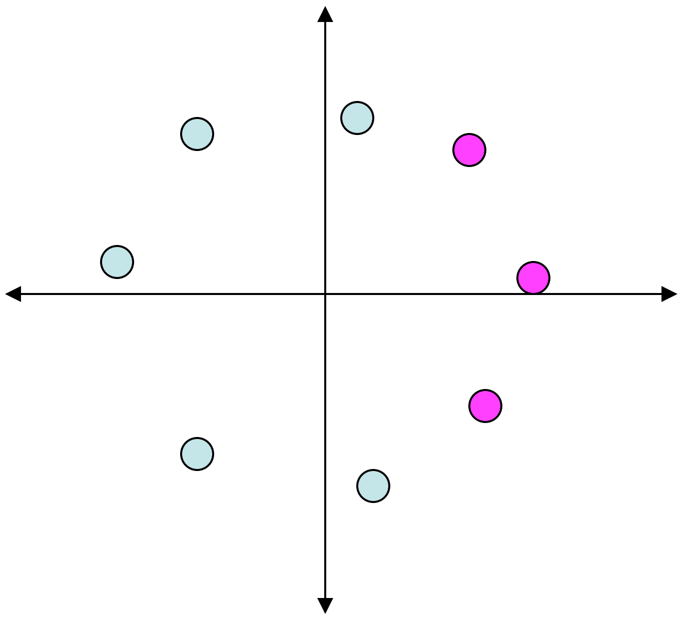
Preprocessing step

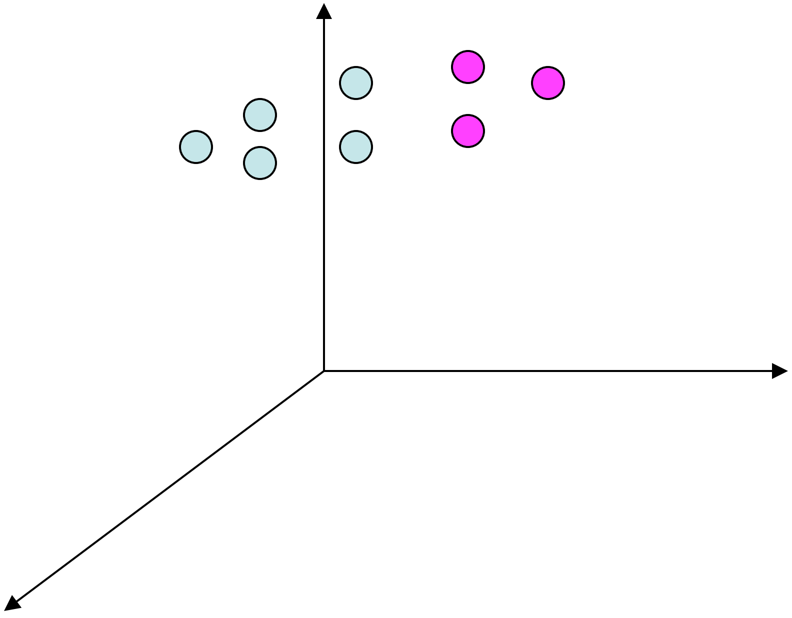
Points (x,y), where input space $X = R^d$, label space $Y = \{+1,-1\}$

Add an extra feature to x, and set it to 1:

$x^0 = (x,1)$ in R^{d+1}

Then: points (x,y) linearly separable $\Leftrightarrow$ points (x^0, y) linearly separable by a hyperplane through the origin





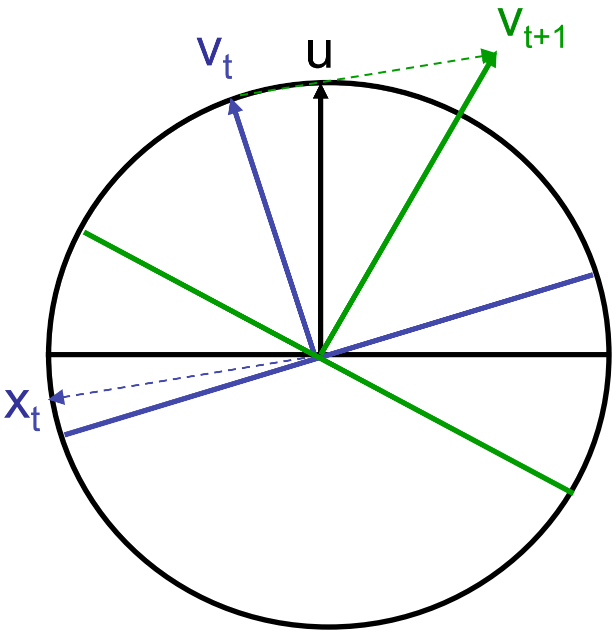
Perceptron

Perceptron: If $y_t(v_t \,.\, x_t) < 0$ Filtering rule

$v_{t+1} = v_t + y_t\, x_t$ Update step

NOTE: Additive updates. Algorithm credited to [Rosenblatt '58].

Example: data is separable by u:



Perceptron

Analyses of standard Perceptron:

Linearly separable (through origin) data, uniform distribution:

- $\tilde{O}(d/\varepsilon^2)$ mistakes (to reach error ε) [Baum ’89].
- $\Omega(1/\varepsilon^2)$ mistakes [Dasgupta, Kalai & Monteleoni, ’05].
- $\Omega(1/\varepsilon^2)$ labels for active learning [Dasgupta, Kalai & Monteleoni, ’05].

Margin assumption: no distribution assumption except linearly separable (through origin), with margin γ

$$y_t(u \,.\, x_t) \geq \gamma \text{ for all } t.$$

- $O(1/\gamma^2)$ mistakes to reach zero error [Novikoff ‘62].

Perceptron analysis with margin

Margin assumption: no distribution assumption except separable (through origin), AND: $y_t(u \,.\, x_t) \geq \gamma$ for all t.

- $O(1/\gamma^2)$ mistakes to reach zero error [Novikoff '62].

Proof: Let $||u|| = 1$. Let (x, y) be a mistake, i.e. $y\,(v_t \,.\, x) < 0$, $||x|| \leq R$.

Lemma 1: $u \,.\, v_{t+1} \geq u \,.\, v_t + \gamma$.

Proof: $u \,.\, v_{t+1} = u \,.\, (v_t + y\,x) = u \,.\, v_t + y\,(u \,.\, x) \geq u \,.\, v_t + \gamma$

(by definition of margin, γ).

Lemma 2: $||v_{t+1}||^2 \leq ||\,v_t||^2 + R^2$

Proof: $||v_{t+1}||^2 = ||v_t + y\,x||^2 = ||\,v_t||^2 + 2y(v_t \,.\, x) + ||x||^2$

$\leq ||\,v_t||^2 + 2y(v_t \,.\, x) + R^2$

$< ||\,v_t||^2 + R^2$

because v_t makes a mistake on (x, y), i.e. $2y(v_t \,.\, x) < 0$.

Perceptron analysis with margin

Proof continued:

Let $||u|| = 1$. Let (x, y) be a mistake, i.e. $y\,(v_t \,.\, x) < 0$, $||x|| \leq R$.

Lemma 1: $u \,.\, v_{t+1} \geq u \,.\, v_t + \gamma$.

Lemma 2: $||v_{t+1}||^2 \leq ||\, v_t||^2 + R^2$.

Finally, after M mistakes:

a. $u \,.\, v_{M+1} \geq M\,\gamma$, by Lemma 1 (expanding the recurrence).

b. $||v_{M+1}||^2 \leq M\,R^2$, by Lemma 2. So $||v_{M+1}|| \leq M^{½}\,R$.

Since u is a unit vector, $u \,.\, v_t \leq ||\, v_t||$ by Cauchy-Schwartz: $|u \,.\, v| \leq ||u||\; ||v||$

So, $u \,.\, v_{M+1} \leq ||\, v_{M+1}||$.

Using a. and b. for LHS and RHS respectively,

$$M\,\gamma \leq u \,.\, v_{M+1} \leq ||\, v_{M+1}|| \leq M^{½}\,R$$

$$M\,\gamma \leq M^{½}\,R$$

$$M^{½} \leq R / \gamma, \text{ and } M \leq (R / \gamma\,)^2 \quad \square$$

Fisher’ s IRIS data

Four features

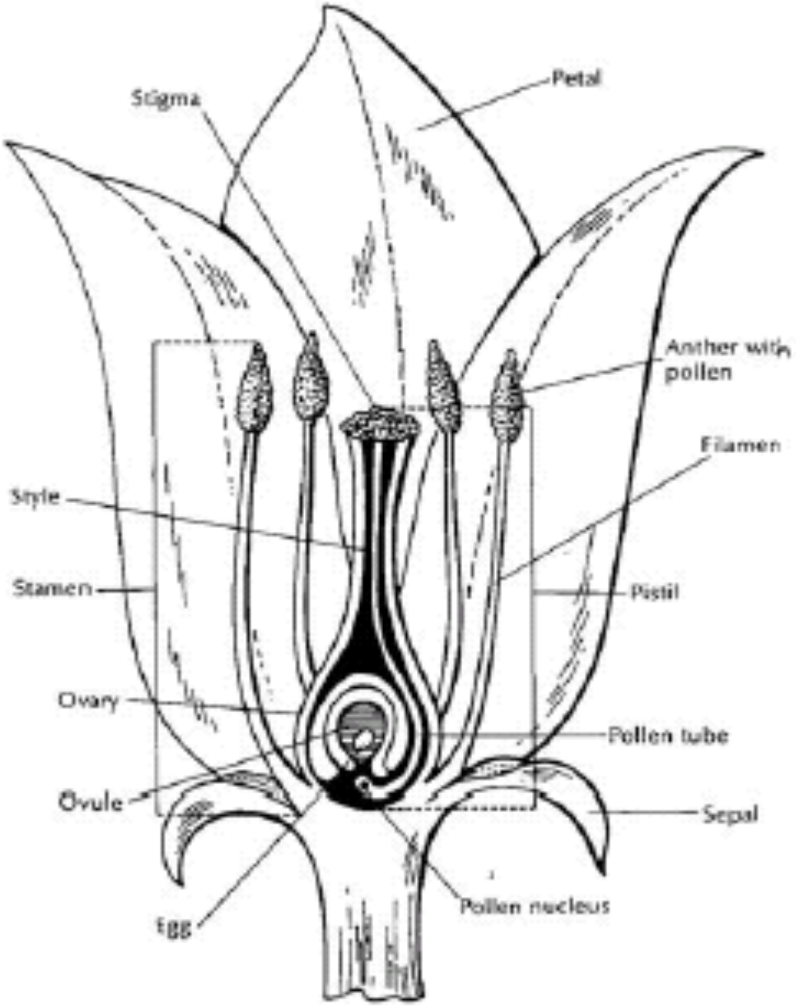
- sepal length
- sepal width
- petal length
- petal width

Three classes (species of iris)

- setosa
- versicolor
- virginica

50 instances of each

Parts of a Flower



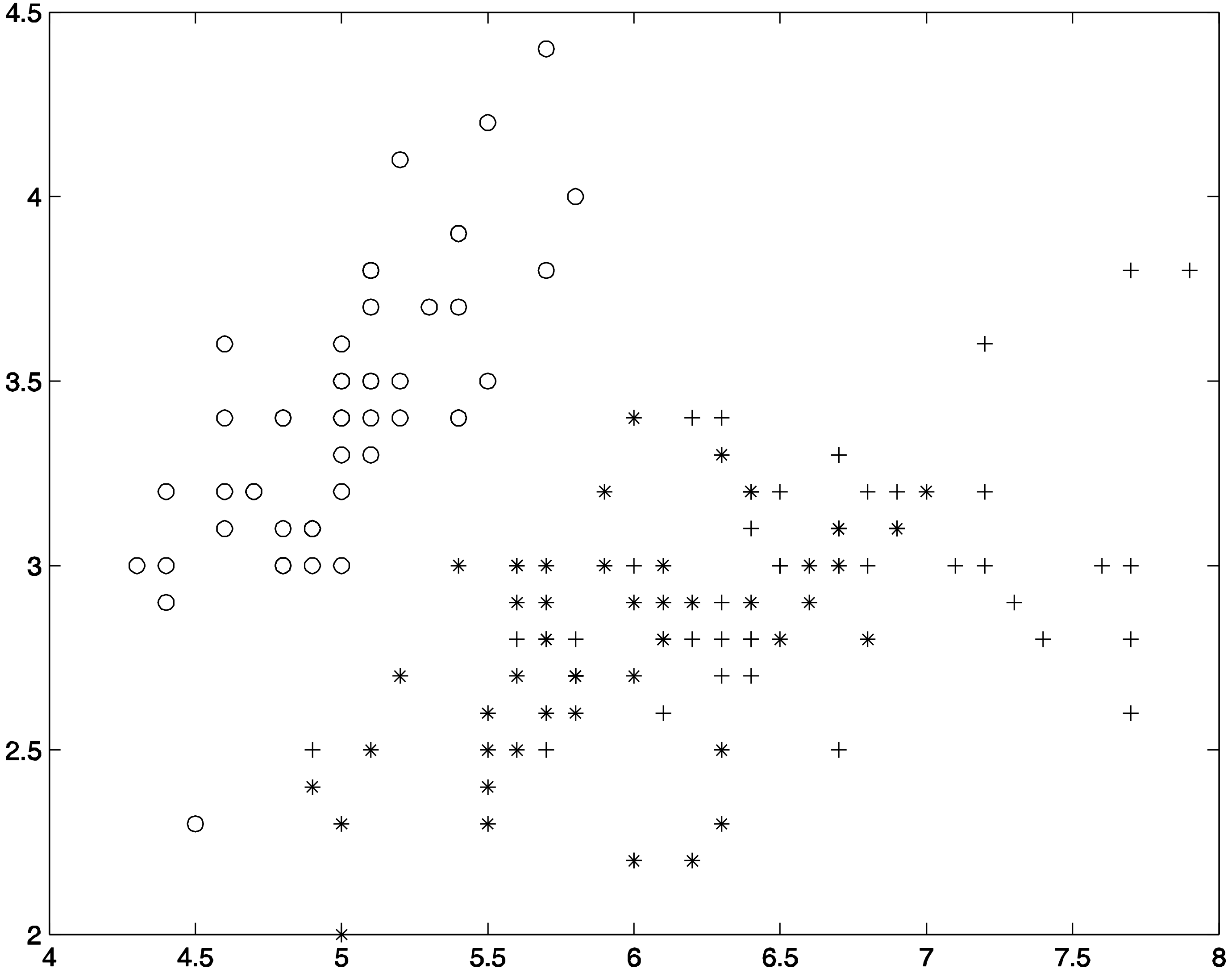
Reproduced from the Annals of Eugenics, v. 7, p. 179-188 (1936) with permission of Cambridge University Press.

THE USE OF MULTIPLE MEASUREMENTS IN TAXONOMIC PROBLEMS

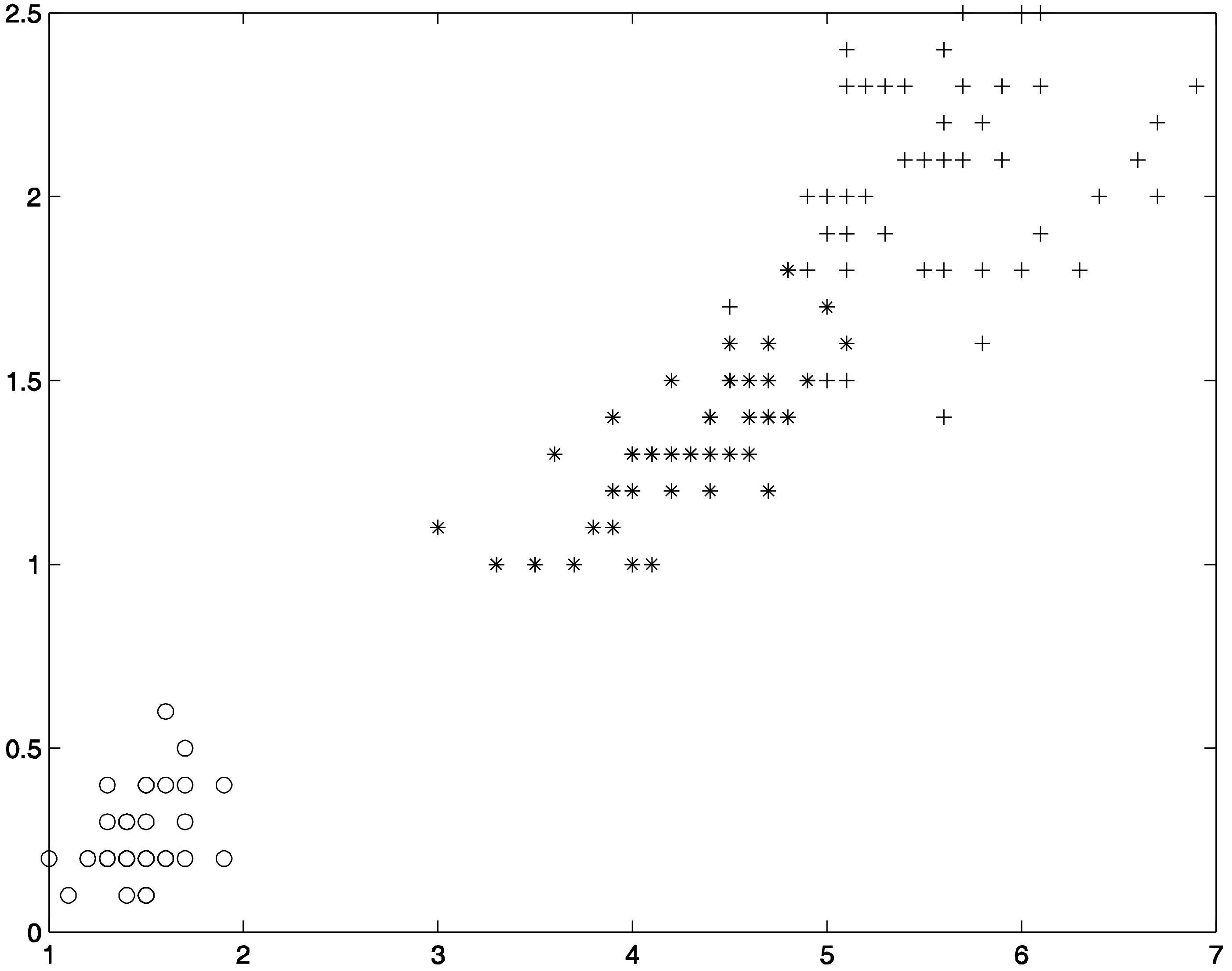
By R. A. FISHER, Sc.D., F.R.S.

I. Discriminant functions

When two or more populations have been measured in several characters, $x_1, \ldots, x_s$, special interest attaches to certain linear functions of the measurements by which the populations are best discriminated. At the author's suggestion use has already been made of this fact in craniometry (*a*) by Mr E. S. Martin, who has applied the principle to the sex differences in measurements of the mandible, and (*b*) by Miss Mildred Barnard, who showed how to obtain from a series of dated series the particular compound of cranial measurements showing most distinctly a progressive or secular trend. In the present paper the application of the same principle will be illustrated on a taxonomic problem; some questions connected with the precision of the processes employed will also be discussed.

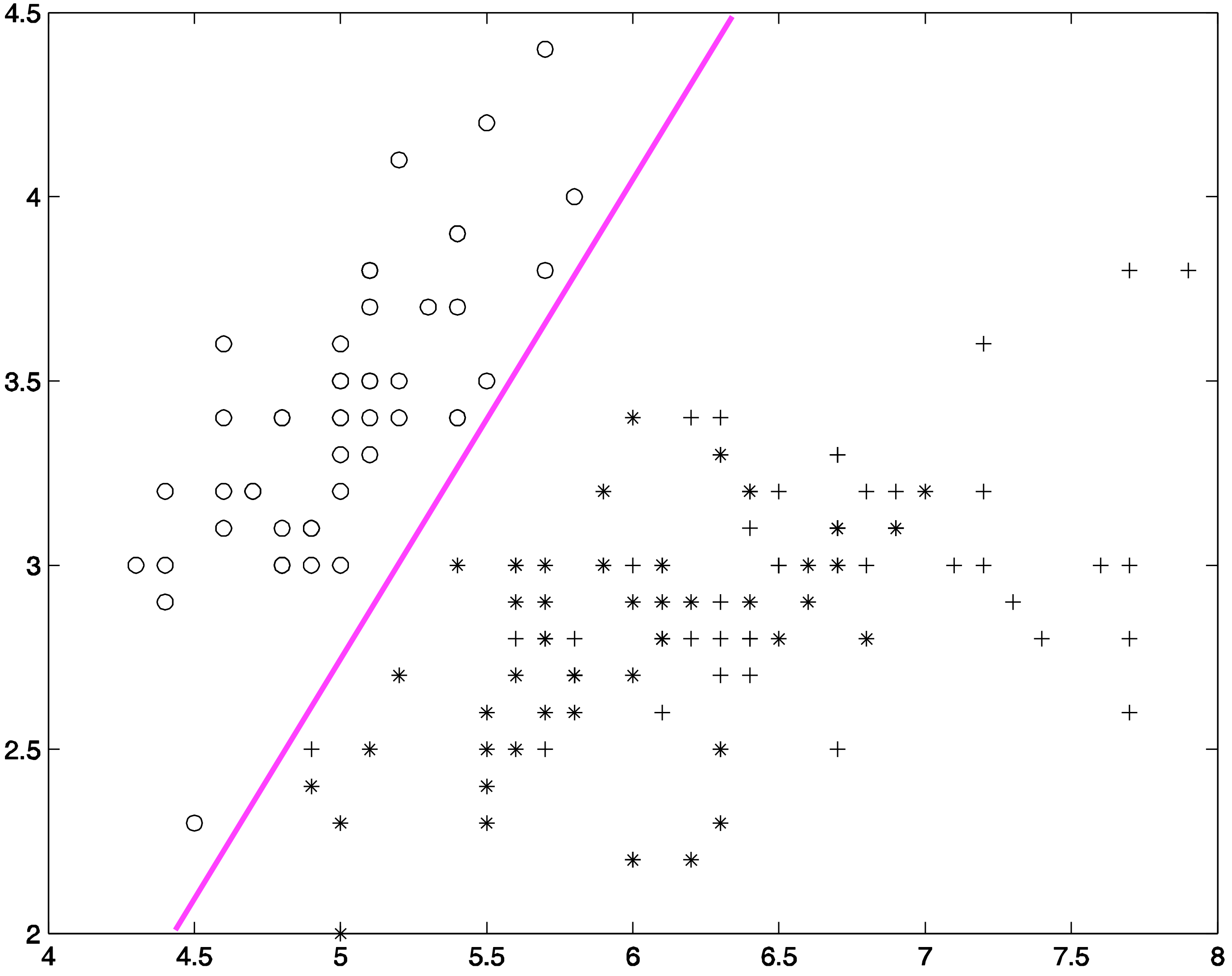


Features 1 and 2 (sepal width/length)



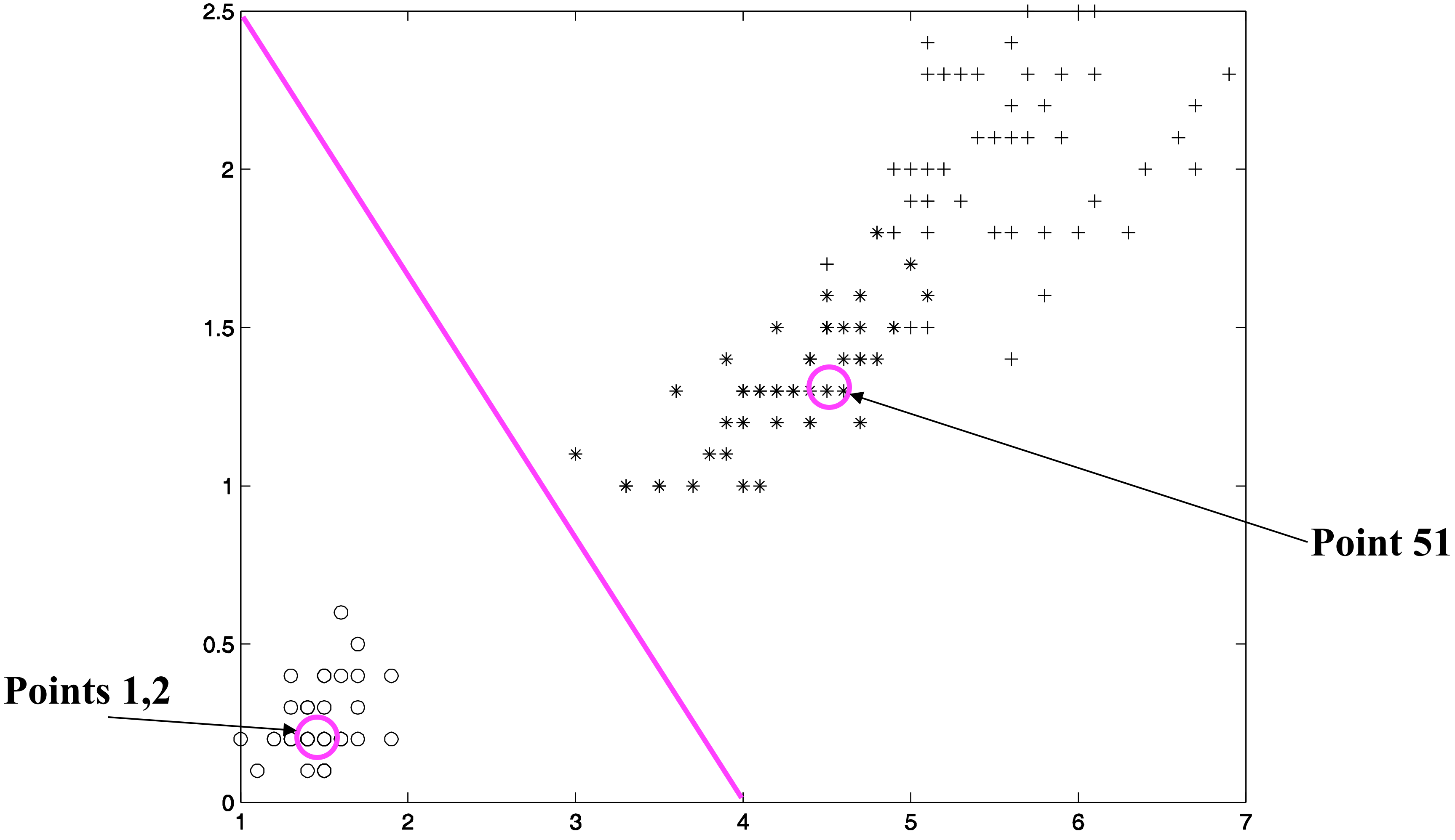
Features 3 and 4 (petal width/length)

Features 1 and 2; goal: separate setosa from other two



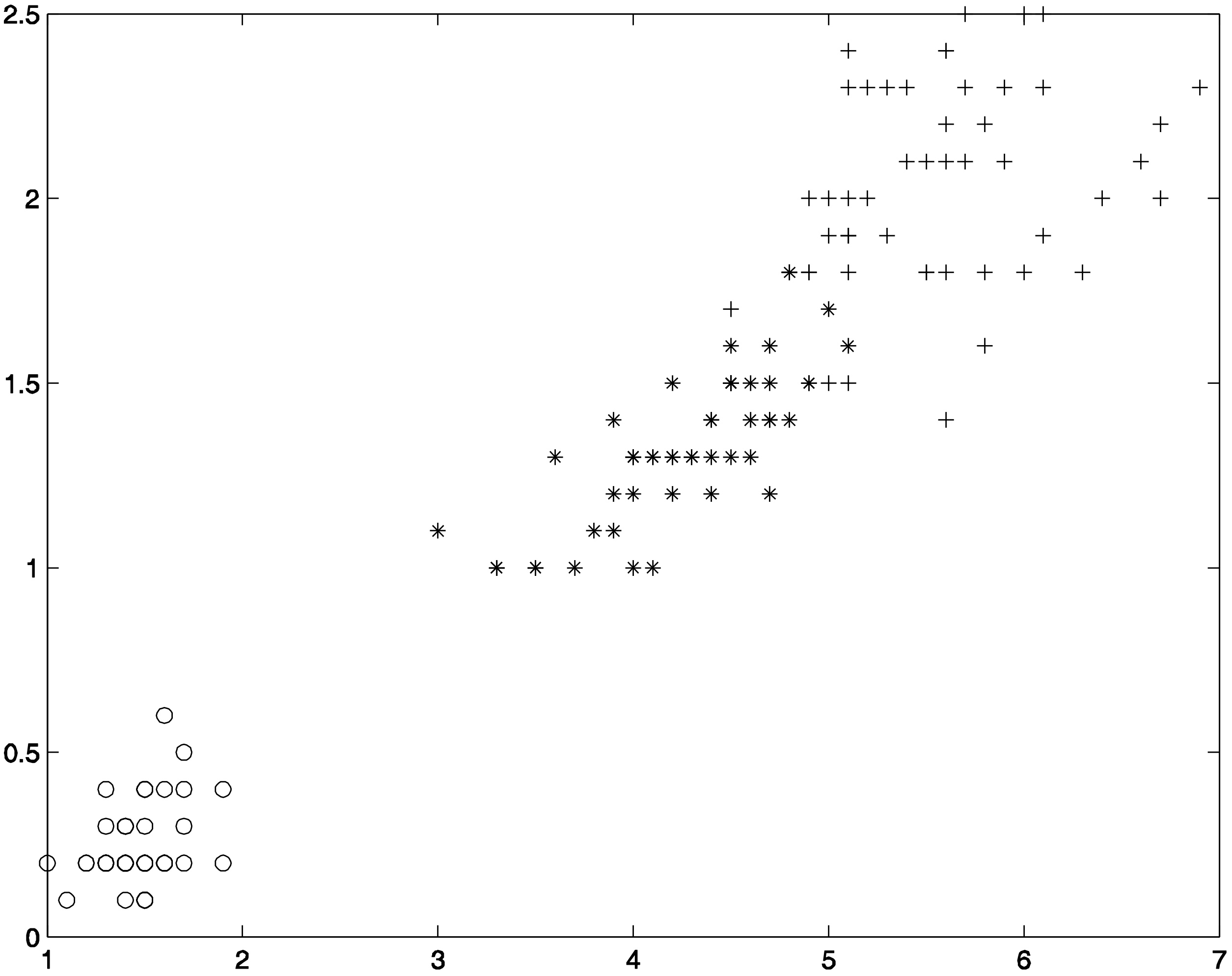
1500 updates (different permutation: 900)

Features 3 and 4; goal: separate setosa from other two



Iteration 1 [1,51] Iteration 2 [1,2] Iteration 3 [ ]

Features 3 and 4; goal: separate versicolor from other two



What if the data is not linearly separable?