

Learning From Data

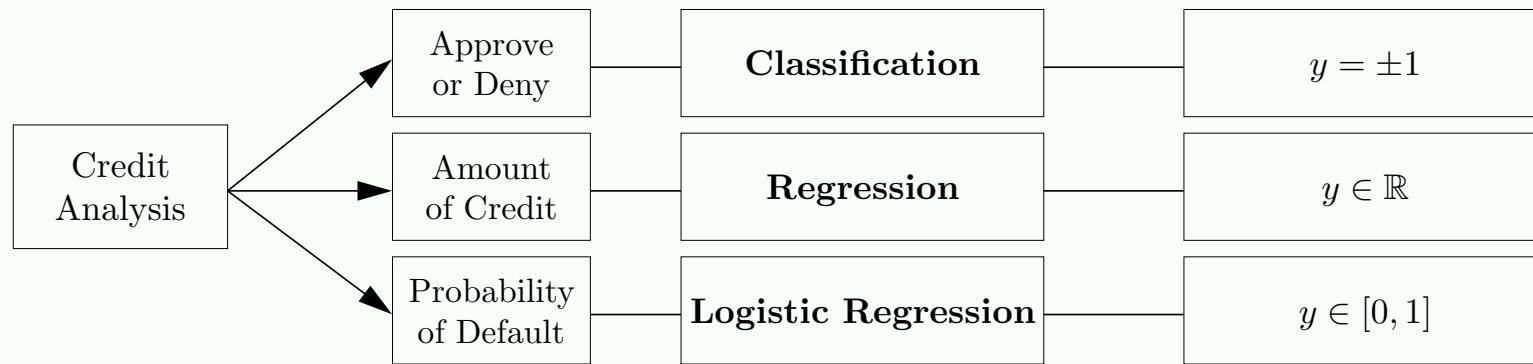
Lecture 8

Linear Classification and Regression

Linear Classification
Linear Regression

M. Magdon-Ismail
CSCI 4100/6100

Three Learning Problems



- Linear models are perhaps *the* fundamental model.
- The linear model is the first model to try.

The Linear Signal

linear in \mathbf{x} : gives the line/hyperplane separator



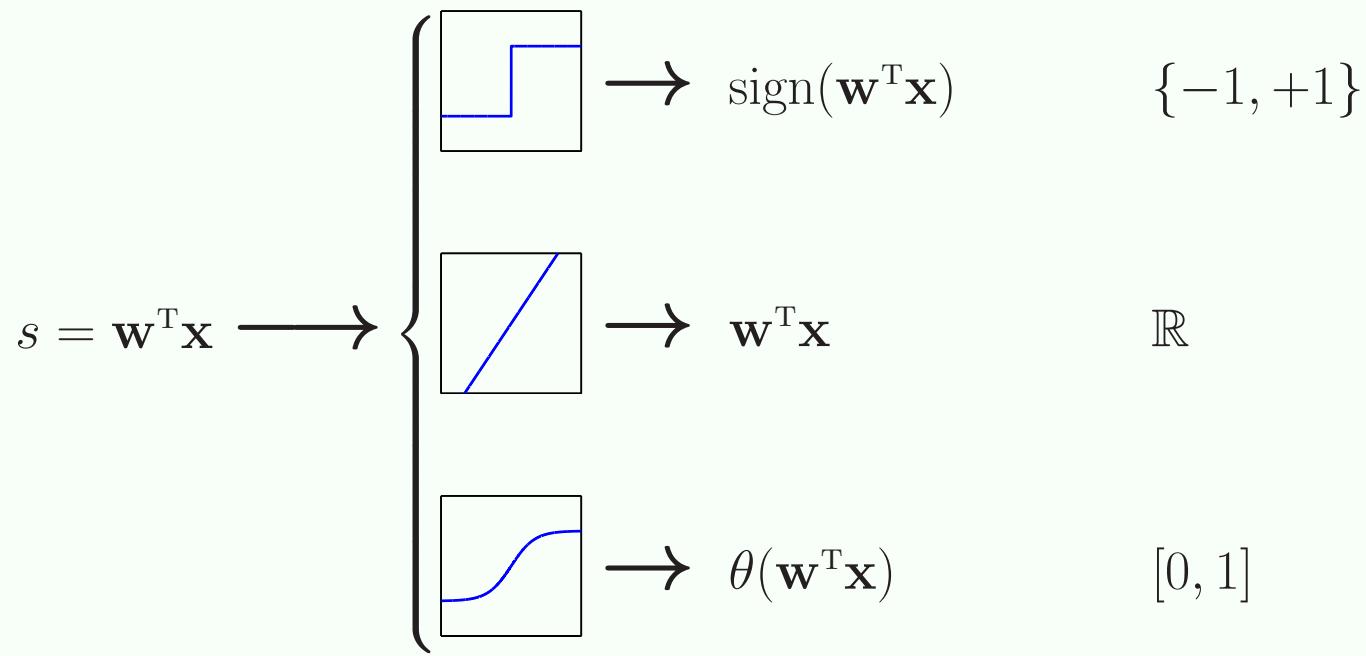
$$s = \mathbf{w}^T \mathbf{x}$$



linear in \mathbf{w} : makes the algorithms work

\mathbf{x} is the augmented vector: $\mathbf{x} \in \{1\} \times \mathbb{R}^d$

The Linear Signal



$$y = \theta(s)$$

Linear Classification

$$\mathcal{H}_{\text{lin}} = \{h(\mathbf{x}) = \text{sign}(\mathbf{w}^T \mathbf{x})\}$$

1. $E_{\text{in}} \approx E_{\text{out}}$ because $d_{\text{VC}} = d + 1$,

$$E_{\text{out}}(h) \leq E_{\text{in}}(h) + O\left(\sqrt{\frac{d}{N} \log N}\right).$$

2. If the data is linearly separable, PLA will find a separator $\implies E_{\text{in}} = 0$.

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mathbf{x}_* y_*$$

↑
misclassified data point

$$E_{\text{in}} = 0 \implies E_{\text{out}} \approx 0$$

(f is well approximated by a linear fit).

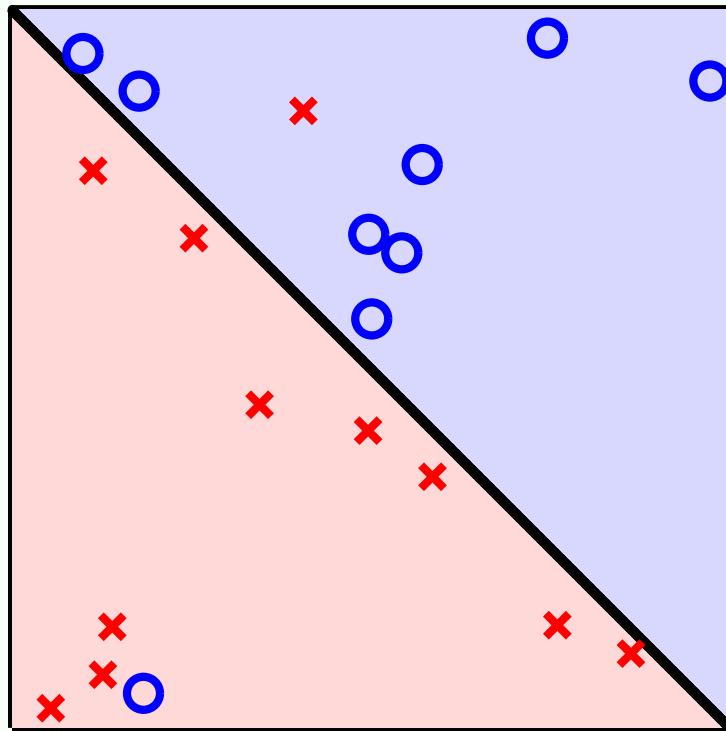
What if the data is not separable ($E_{\text{in}} = 0$ is not possible)?

pocket algorithm

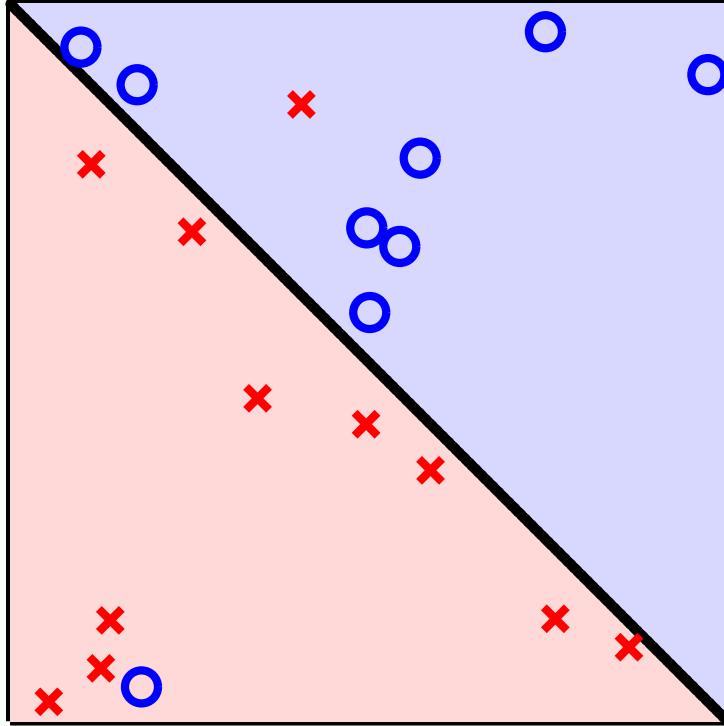
How to ensure $E_{\text{in}} \approx 0$ is possible?

select good features

Non-Separable Data



The Pocket Algorithm



Minimizing E_{in} is a *hard* combinatorial problem.

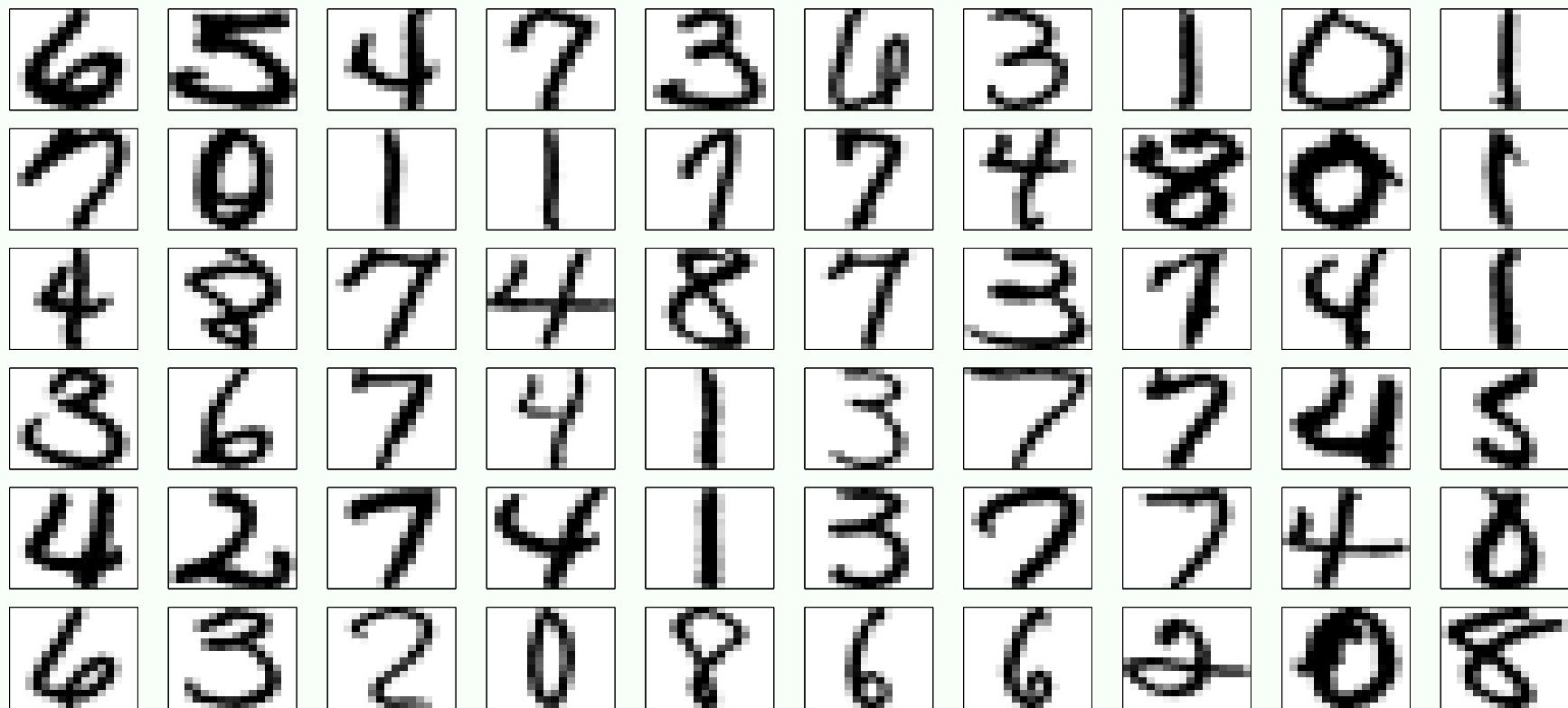
The Pocket Algorithm

- Run PLA
- At each step keep the best E_{in} (and \mathbf{w}) so far.

(Its not rocket science, but it works.)

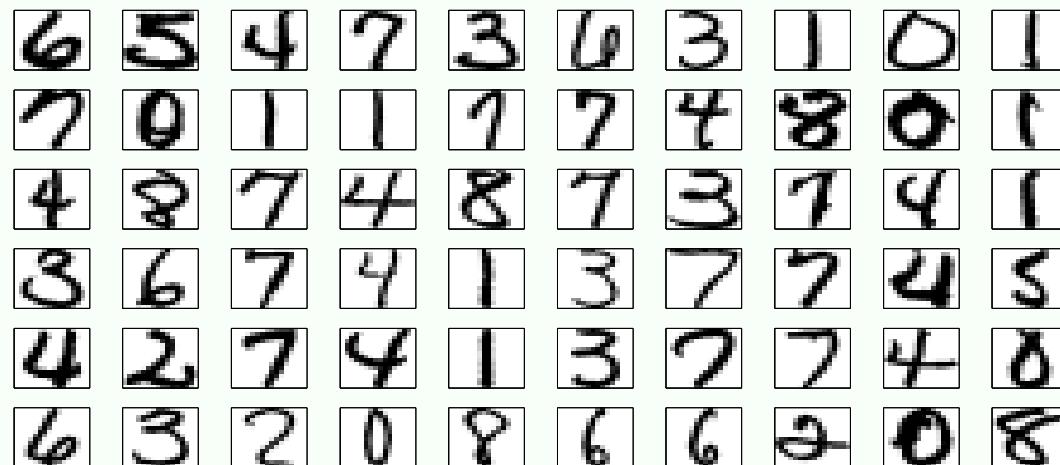
(Other approaches: linear regression, logistic regression, linear programming . . .)

Digits Data



Each digit is a 16×16 image.

Digits Data



Each digit is a 16×16 image.

```

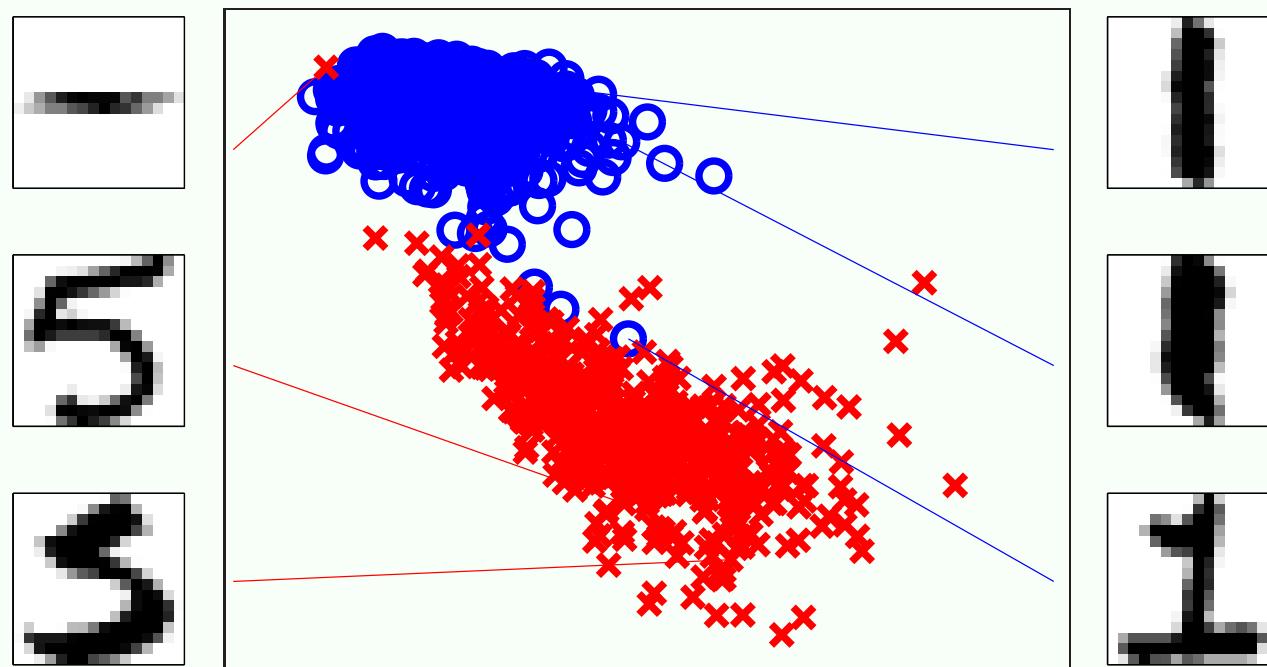
[-1 -1 -1 -1 -1 -1 -1 -0.63 0.86 -0.17 -1 -1 -1 -1 -1 -1 -1 -1 -0.99 0.3 1 0.31 -1 -1 -1 -1 -1 -1 -1 -0.41 1 0.99 -0.57 -1 -1 -1 -1 -1 -1 -1 -1 -0.68 0.83 1
0.56 -1 -1 -1 -1 -1 -1 -1 -1 -1 -0.94 0.54 1 0.78 -0.72 -1 -1 -1 -1 -1 -1 -1 -0.1 1 0.92 -0.44 -1 -1 -1 -1 -1 -1 -1 -0.26 0.95 1 -0.16 -1 -1 -1 -0.99 -0.71 -0.83 -1
-1 -1 -1 -1 -0.8 0.91 1 0.3 -0.96 -1 -1 -0.55 0.49 1 0.88 0.09 -1 -1 -1 -1 0.28 1 0.88 -0.8 -1 -0.9 0.14 0.97 1 1 1 0.99 -0.74 -1 -1 -0.95 0.84 1 0.32 -1 -1 0.35 1 0.65 -0.10 -0.18 1 0.98
-0.72 -1 -1 -0.63 1 1 0.07 -0.92 0.11 0.96 0.30 -0.88 -1 -0.07 1 0.64 -0.99 -1 -1 -0.67 1 1 0.75 0.34 1 0.70 -0.94 -1 -1 0.54 1 0.02 -1 -1 -1 -0.90 0.79 1 1 1 1 0.53 0.18 0.81 0.83 0.97
0.86 -0.63 -1 -1 -1 -1 -0.45 0.82 1 1 1 1 1 1 1 1 0.13 -1 -1 -1 -1 -1 -1 -0.48 0.81 1 1 1 1 1 1 0.21 -0.94 -1 -1 -1 -1 -1 -1 -0.97 -0.42 0.30 0.82 1 0.48 -0.47 -0.99 -1 -1 -1 -1]

```

$$\left. \begin{array}{l} \mathbf{x} = (1, x_1, \dots, x_{256}) \leftarrow \text{input} \\ \mathbf{w} = (w_0, w_1, \dots, w_{256}) \leftarrow \text{linear model} \end{array} \right\} d_{\text{VC}} = 257$$

Intensity and Symmetry Features

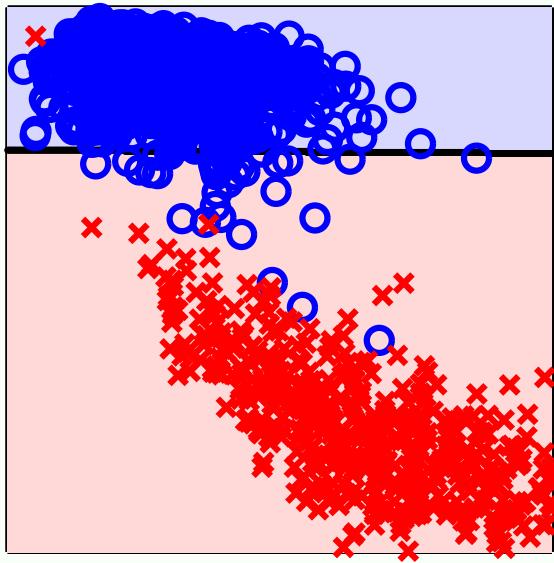
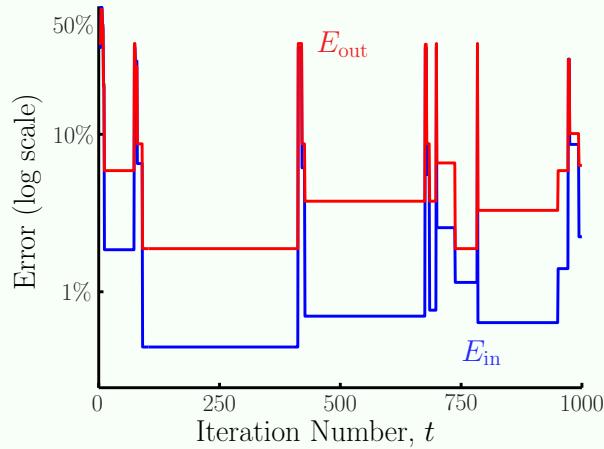
feature: an important property of the input that you think is useful for classification.
(dictionary.com: a prominent or conspicuous part or characteristic)



$$\left. \begin{array}{l} \mathbf{x} = (1, x_1, x_2) \leftarrow \text{input} \\ \mathbf{w} = (w_0, w_1, w_2) \leftarrow \text{linear model} \end{array} \right\} d_{VC} = 3$$

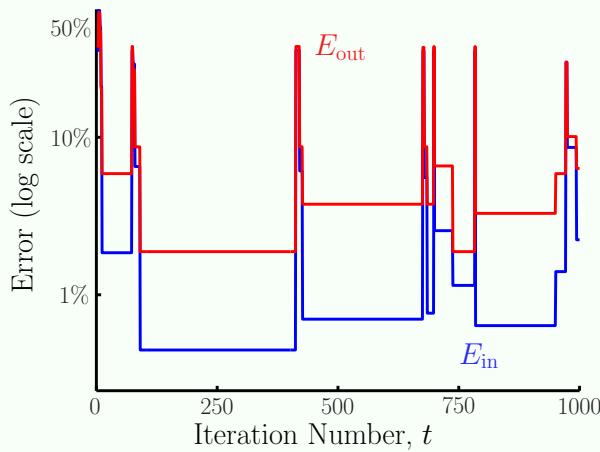
PLA on Digits Data

PLA

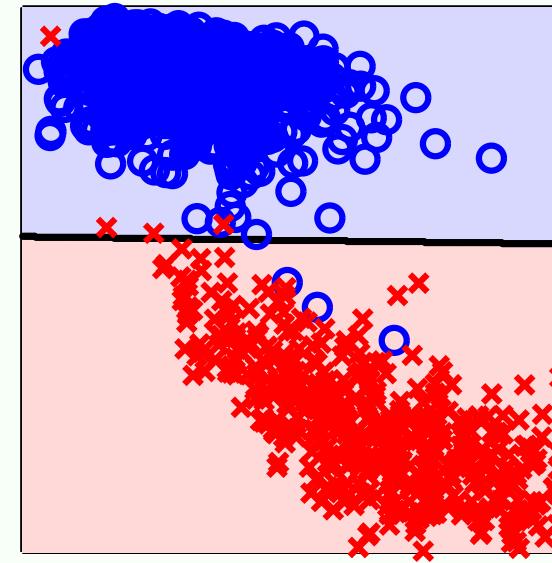
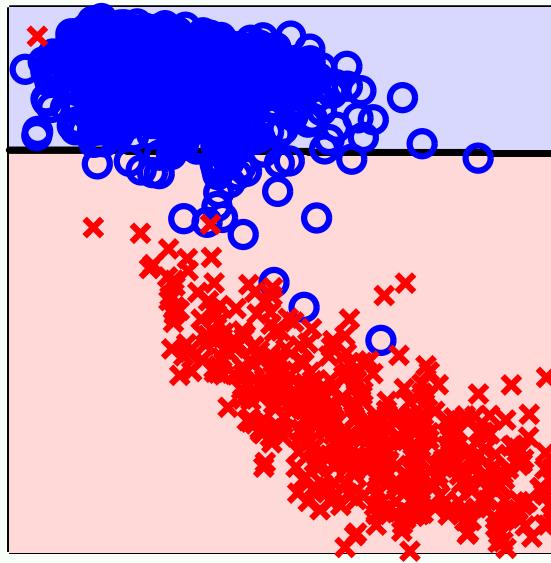
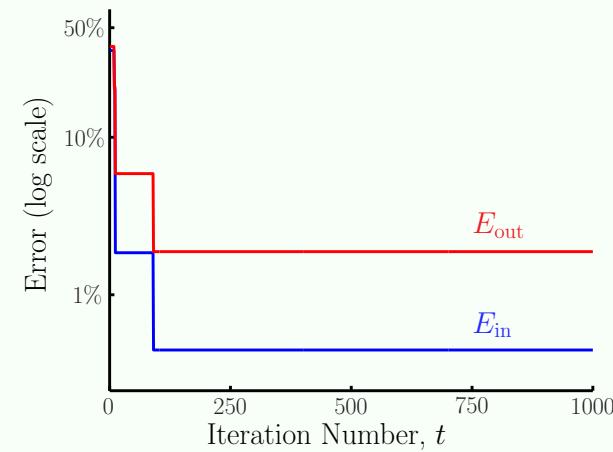


Pocket on Digits Data

PLA



Pocket



Linear Regression

age	32 years
gender	male
salary	40,000
debt	26,000
years in job	1 year
years at home	3 years
...	...

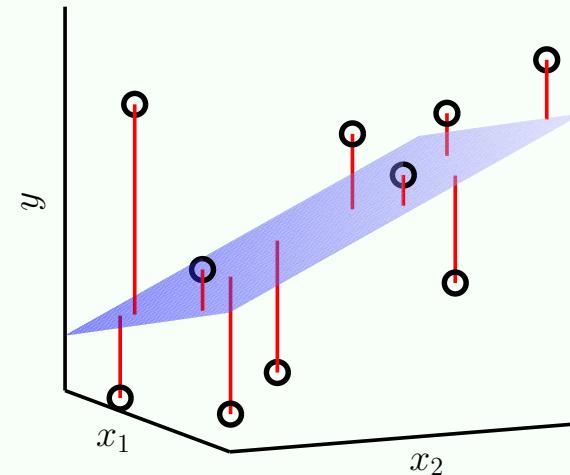
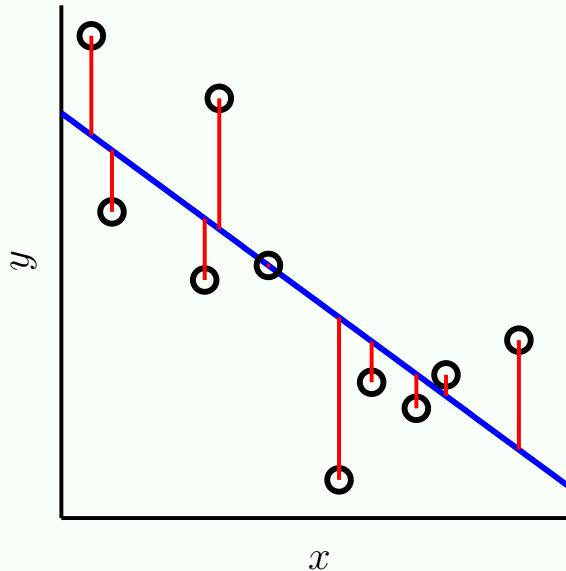
Classification: Approve/Deny

Regression: Credit Line (dollar amount)

regression $\equiv y \in \mathbb{R}$

$$h(\mathbf{x}) = \sum_{i=0}^d w_i x_i = \mathbf{w}^T \mathbf{x}$$

Least Squares Linear Regression



$$y = f(\mathbf{x}) + \epsilon$$

← noisy target $P(y|\mathbf{x})$

in-sample error

$$E_{\text{in}}(h) = \frac{1}{N} \sum_{n=1}^N (h(\mathbf{x}_n) - y_n)^2$$

out-of-sample error

$$E_{\text{out}}(h) = \mathbb{E}_{\mathbf{x}}[(h(\mathbf{x}) - y)^2]$$

$$h(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$$

Using Matrices for Linear Regression

$$X = \underbrace{\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_N \end{bmatrix}}_{\text{data matrix, } N \times (d+1)}$$

$$\mathbf{y} = \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}}_{\text{target vector}}$$

$$\hat{\mathbf{y}} = \underbrace{\begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_N \end{bmatrix}}_{\text{in-sample predictions}} = \begin{bmatrix} \mathbf{w}^T \mathbf{x}_1 \\ \mathbf{w}^T \mathbf{x}_2 \\ \vdots \\ \mathbf{w}^T \mathbf{x}_N \end{bmatrix} = X\mathbf{w}$$

$$\begin{aligned}
 E_{\text{in}}(\mathbf{w}) &= \frac{1}{N} \sum_{n=1}^N (\hat{y}_n - y_n)^2 \quad \text{想想L-P範數的意思} \\
 &= \frac{1}{N} \| \hat{\mathbf{y}} - \mathbf{y} \|_2^2 \\
 &= \frac{1}{N} \| X\mathbf{w} - \mathbf{y} \|_2^2 \quad \text{How??} \\
 &= \frac{1}{N} (\mathbf{w}^T X^T X \mathbf{w} - 2\mathbf{w}^T X^T \mathbf{y} + \mathbf{y}^T \mathbf{y})
 \end{aligned}$$

Linear Regression Solution

[Learning from Data] - P. 85

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N} (\mathbf{w}^T \mathbf{X}^T \mathbf{X} \mathbf{w} - 2\mathbf{w}^T \mathbf{X}^T \mathbf{y} + \mathbf{y}^T \mathbf{y})$$

Vector Calculus: To minimize $E_{\text{in}}(\mathbf{w})$, set $\nabla_{\mathbf{w}} E_{\text{in}}(\mathbf{w}) = \mathbf{0}$.

$$\nabla_{\mathbf{w}} (\mathbf{w}^T \mathbf{A} \mathbf{w}) = (\mathbf{A} + \mathbf{A}^T) \mathbf{w}, \quad \nabla_{\mathbf{w}} (\mathbf{w}^T \mathbf{b}) = \mathbf{b}.$$

$$\mathbf{A} = \mathbf{X}^T \mathbf{X} \text{ and } \mathbf{b} = \mathbf{X}^T \mathbf{y}:$$

$$\nabla_{\mathbf{w}} E_{\text{in}}(\mathbf{w}) = \frac{2}{N} (\mathbf{X}^T \mathbf{X} \mathbf{w} - \mathbf{X}^T \mathbf{y})$$

Setting $\nabla E_{\text{in}}(\mathbf{w}) = \mathbf{0}$:

$$\mathbf{X}^T \mathbf{X} \mathbf{w} = \mathbf{X}^T \mathbf{y} \qquad \longleftarrow \text{normal equations}$$

$$\boxed{\mathbf{w}_{\text{lin}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}} \qquad \longleftarrow \text{when } \mathbf{X}^T \mathbf{X} \text{ is invertible}$$

Linear Regression Algorithm

P 86 - Learn from Data

Linear Regression Algorithm:

1. Construct the matrix X and the vector y from the data set $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)$, where each \mathbf{x} includes the $x_0 = 1$ coordinate,

$$X = \underbrace{\begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_N \end{bmatrix}}_{\text{data matrix}}, \quad y = \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}}_{\text{target vector}}.$$

2. Compute the pseudo inverse X^\dagger of the matrix X . If $X^T X$ is invertible,

$$X^\dagger = (X^T X)^{-1} X^T$$

3. Return $\mathbf{w}_{\text{lin}} = X^\dagger y$.

Generalization

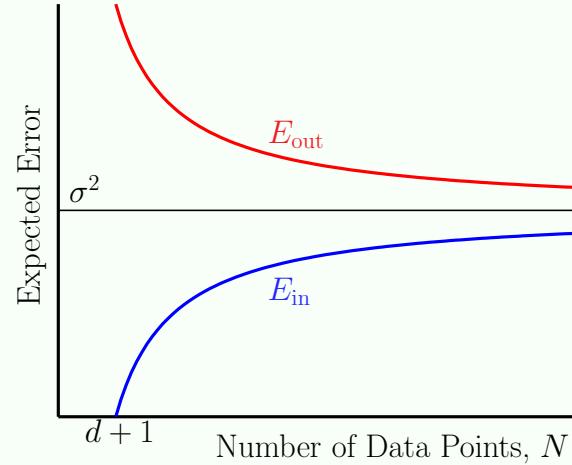
The linear regression algorithm gets the smallest possible E_{in} in *one step*.

Generalization is also good.

One can obtain a regression version of d_{VC} .

There are other bounds, for example:

$$\mathbb{E}[E_{\text{out}}(h)] = \mathbb{E}[E_{\text{in}}(h)] + O\left(\frac{d}{N}\right)$$



Linear Regression for Classification

Linear regression can learn *any* real valued target function.

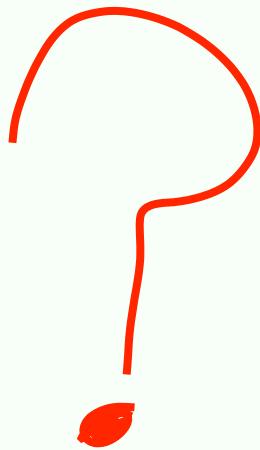
For example $y_n = \pm 1$.

(± 1 are real values!)

Use linear regression to get \mathbf{w} with $\mathbf{w}^T \mathbf{x}_n \approx y_n = \pm 1$

Then $\text{sign}(\mathbf{w}^T \mathbf{x}_n)$ will likely agree with $y_n = \pm 1$.

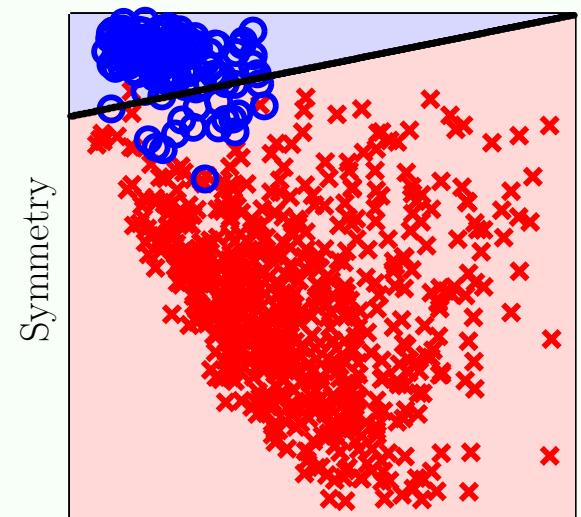
These can be good initial weights for classification.



Example.

Classifying 1 from not 1

(multiclass \rightarrow 2 class)



Average Intensity

