

Supervised Learning (Chpt 2)

Rui Kuang

**Department of Computer Science and Engineering
University of Minnesota**

Key Concepts

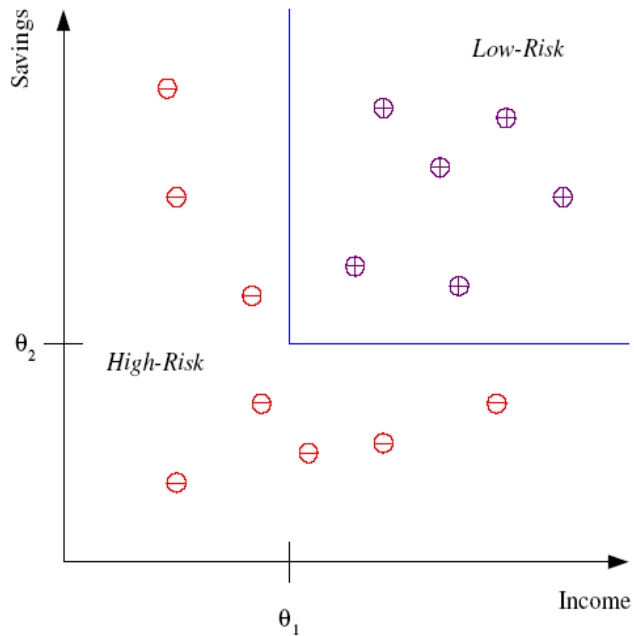
- Formulation of supervised learning: **classification** and **regression**
- First example of classification algorithms: **perceptron learning**
- Fundamental learning theory: **model complexity** and **VC dimension**
- Multiclass classifier: **K-nearest neighbors (KNN)**
- Line regression by **least squares**
- **Trade-off by model complexity** and **cross-validation**

Key Concepts

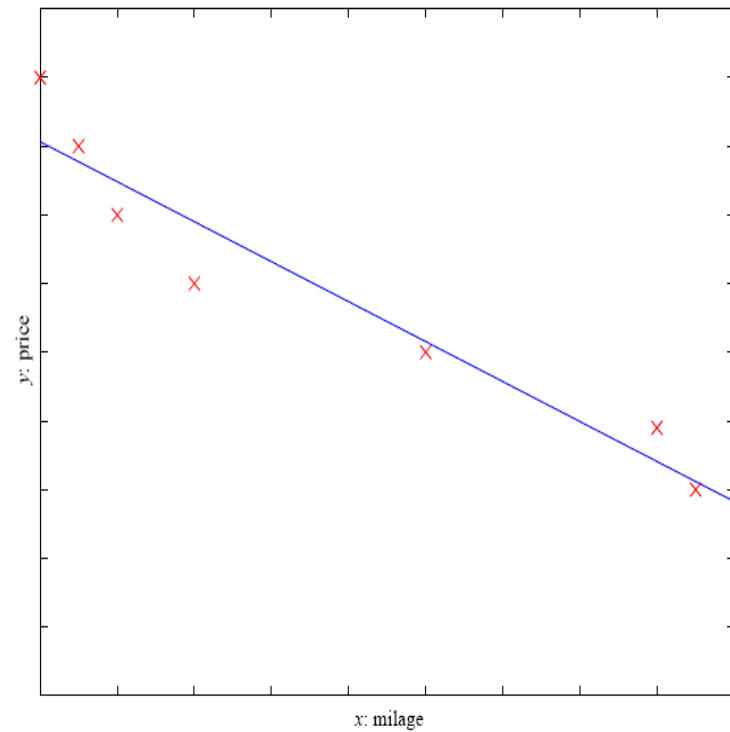
- Formulation of supervised learning: **classification** and **regression**
- First example of classification algorithms: **perceptron learning**
- Fundamental learning theory: **model complexity** and **VC dimension**
- Multiclass classifier: **K-nearest neighbors (KNN)**
- Line regression by **least squares**
- **Trade-off by model complexity** and **cross-validation**

Supervised Learning

■ Classification

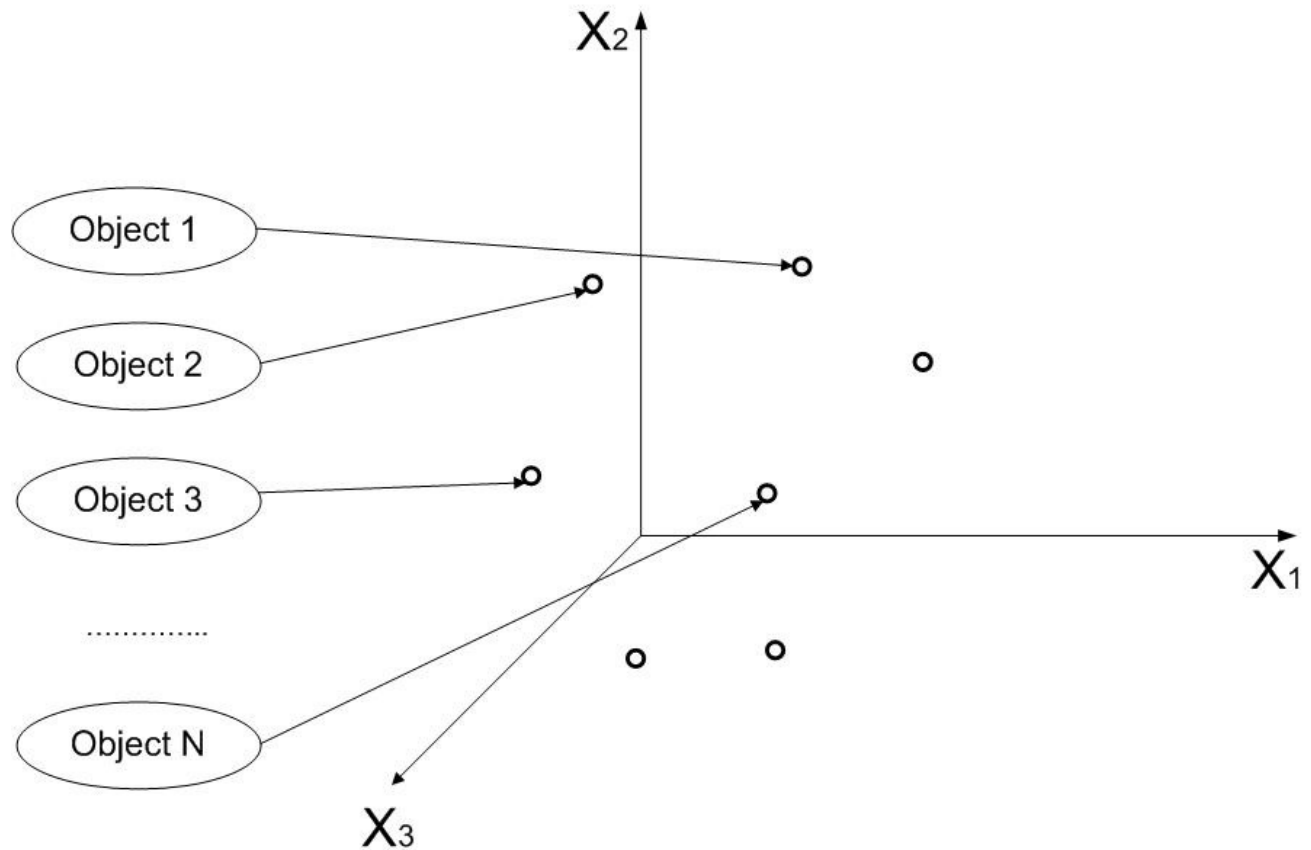


■ Regression



Input Feature Space

$$\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ \dots \\ x_D \end{bmatrix}$$



Supervised Learning

■ Classification

Data:

$$\mathcal{X} = \{\mathbf{x}^t, r^t\}_{t=1}^N$$

Output: $r^t = \begin{cases} 1 & \text{if } \mathbf{x} \text{ is positive} \\ 0 / -1 & \text{if } \mathbf{x} \text{ is negative} \end{cases}$

(Class label)

■ Regression

$$\mathcal{X} = \{\mathbf{x}^t, r^t\}_{t=1}^N$$

$$r^t \in \mathfrak{R}$$

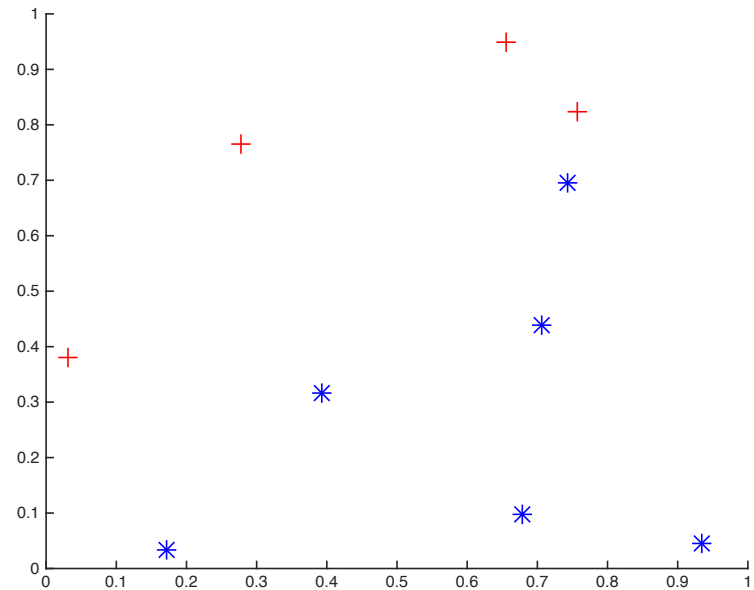
(Response)

Remember the notations on page xxiii-page xxiv in EA book

Classification

Data: $\mathcal{X} = \{\mathbf{x}^t, r^t\}_{t=1}^N$ Output: $r = \begin{cases} 1 & \text{if } \mathbf{x} \text{ is positive} \\ 0 / -1 & \text{if } \mathbf{x} \text{ is negative} \end{cases}$

x_1	x_2	r
0.934	0.046	-1
0.679	0.097	-1
0.758	0.823	1
0.743	0.695	-1
0.392	0.317	-1
0.655	0.950	1
0.171	0.034	-1
0.706	0.439	-1
0.032	0.382	1
0.277	0.766	1





Learning a Class from Examples

- Class C of a “family car”

- Prediction: Is car x a family car?

- Knowledge extraction: What do people expect from a family car?

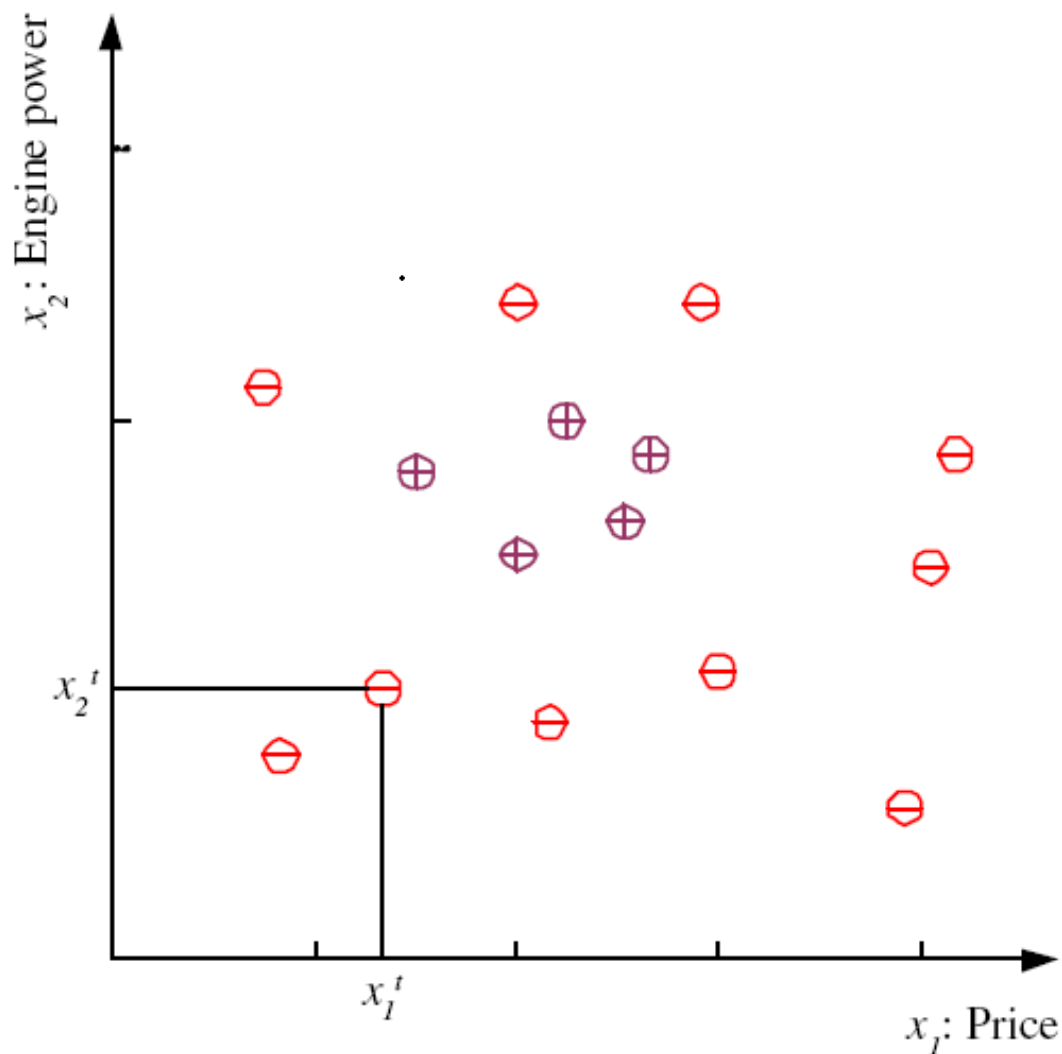
- Output:

- Positive (+) and negative (–) examples

- Input representation:

- x_1 : price, x_2 : engine power

Training set \mathcal{X}

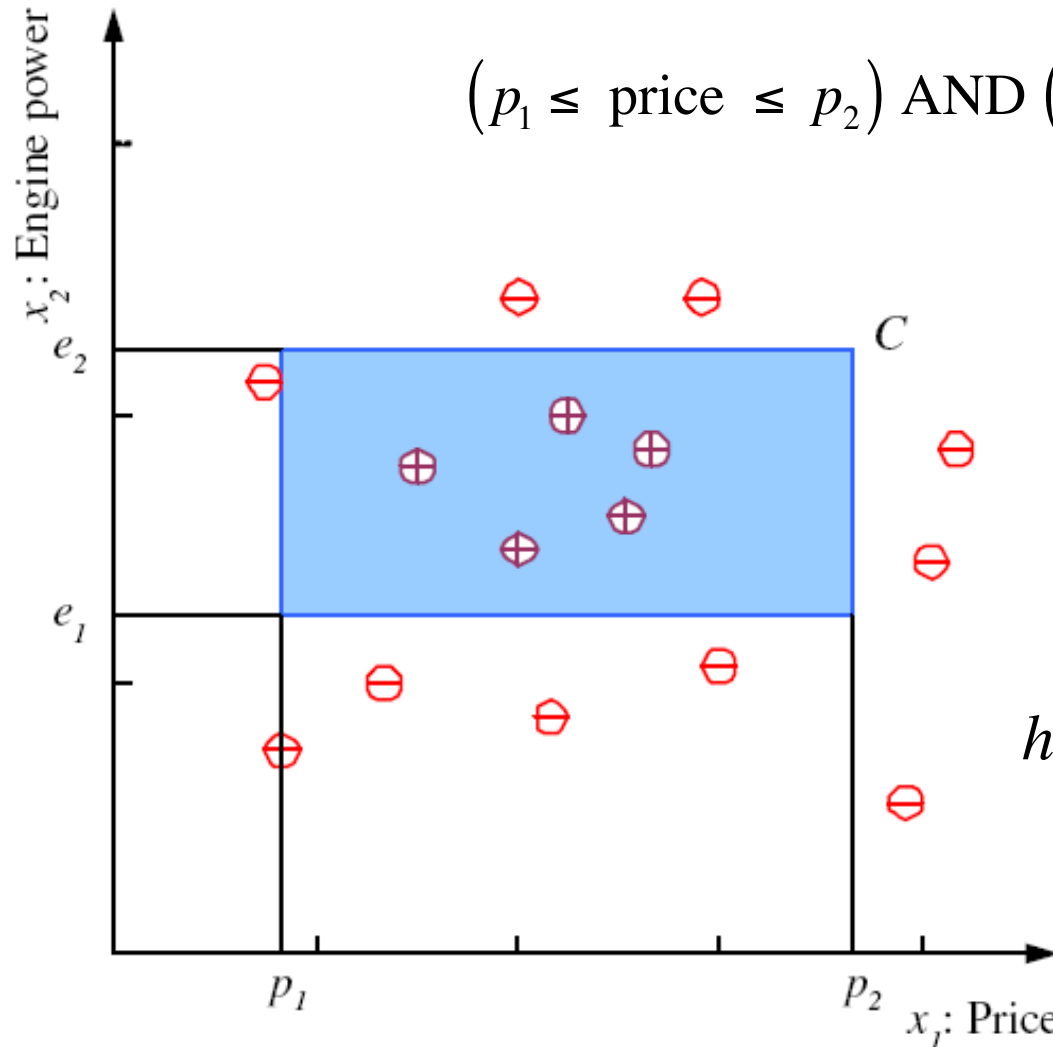


$$\mathcal{X} = \{\mathbf{x}^t, r^t\}_{t=1}^N$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$r = \begin{cases} 1 & \text{if } \mathbf{x} \text{ is positive} \\ 0 & \text{if } \mathbf{x} \text{ is negative} \end{cases}$$

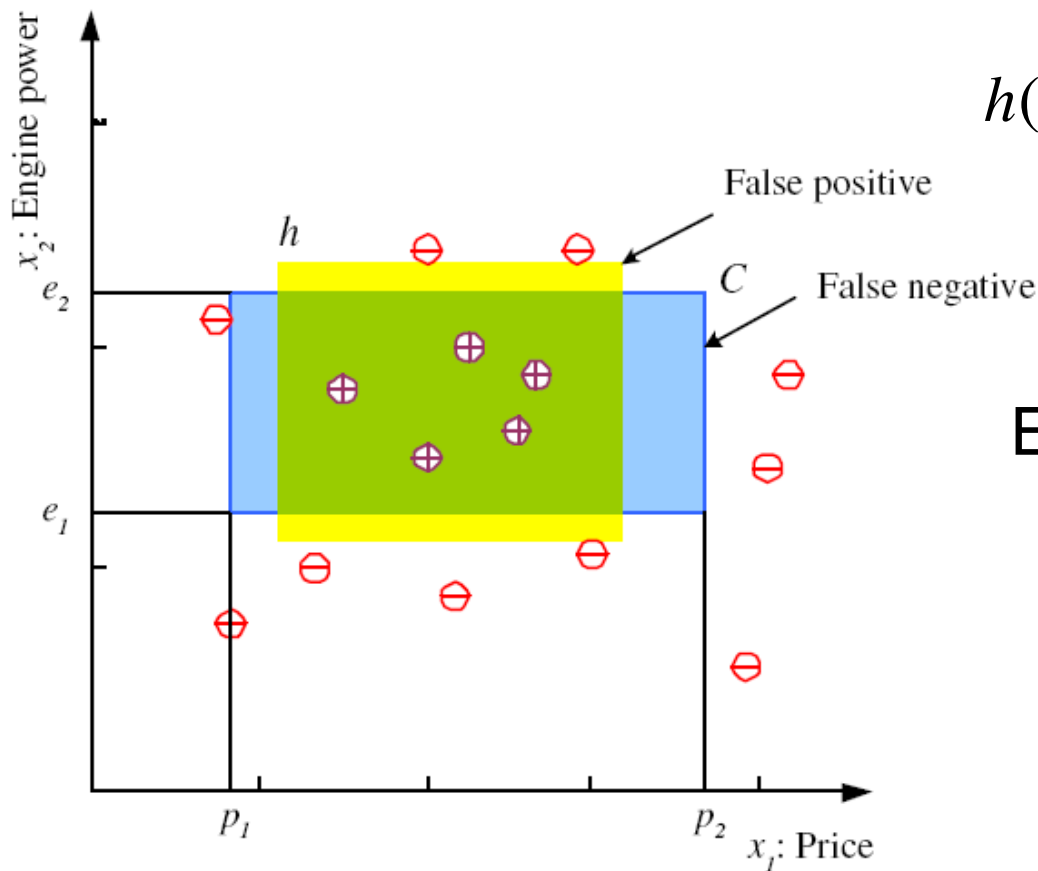
Class in a Rectangle



$$h(\mathbf{x}) = \begin{cases} 1 & \text{if } h \text{ says } \mathbf{x} \text{ is positive} \\ 0 & \text{if } h \text{ says } \mathbf{x} \text{ is negative} \end{cases}$$

Hypothesis class \mathcal{H}

Consider \mathcal{H} : the set of all rectangles

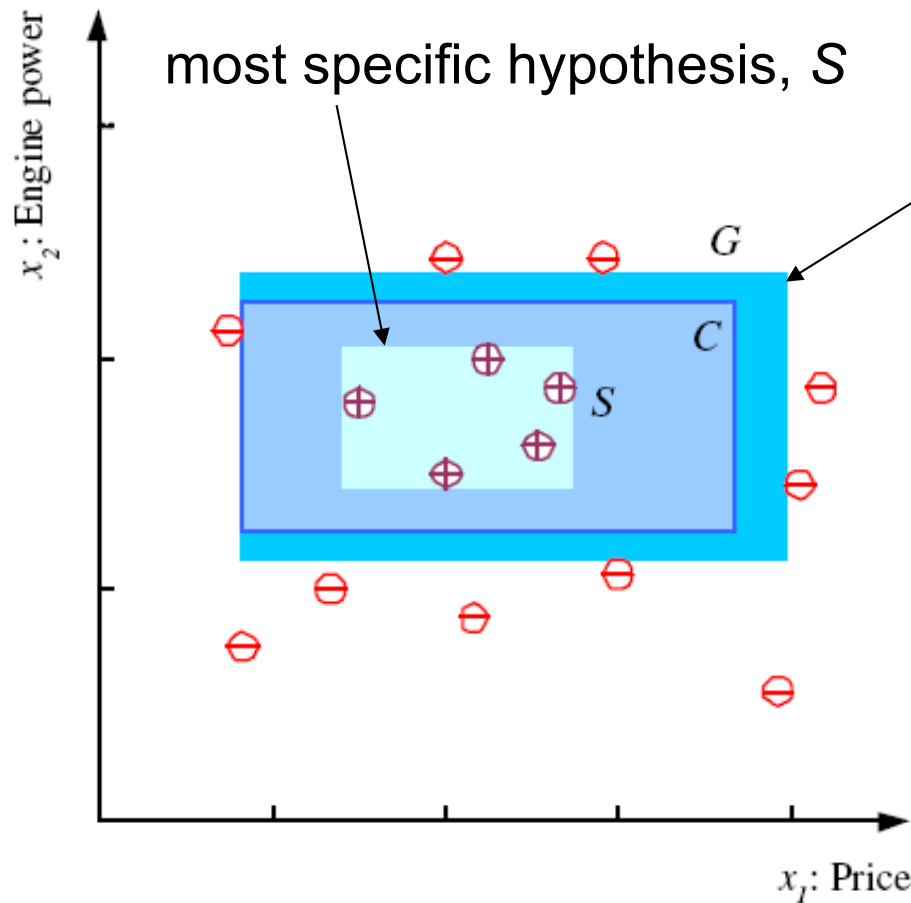


$$h(\mathbf{x}) = \begin{cases} 1 & \text{if } h \text{ says } \mathbf{x} \text{ is positive} \\ 0 & \text{if } h \text{ says } \mathbf{x} \text{ is negative} \end{cases}$$

Error of h on \mathcal{X}

$$E(h | \mathcal{X}) = \sum_{t=1}^N 1(h(\mathbf{x}^t) \neq r^t)$$

Version Space



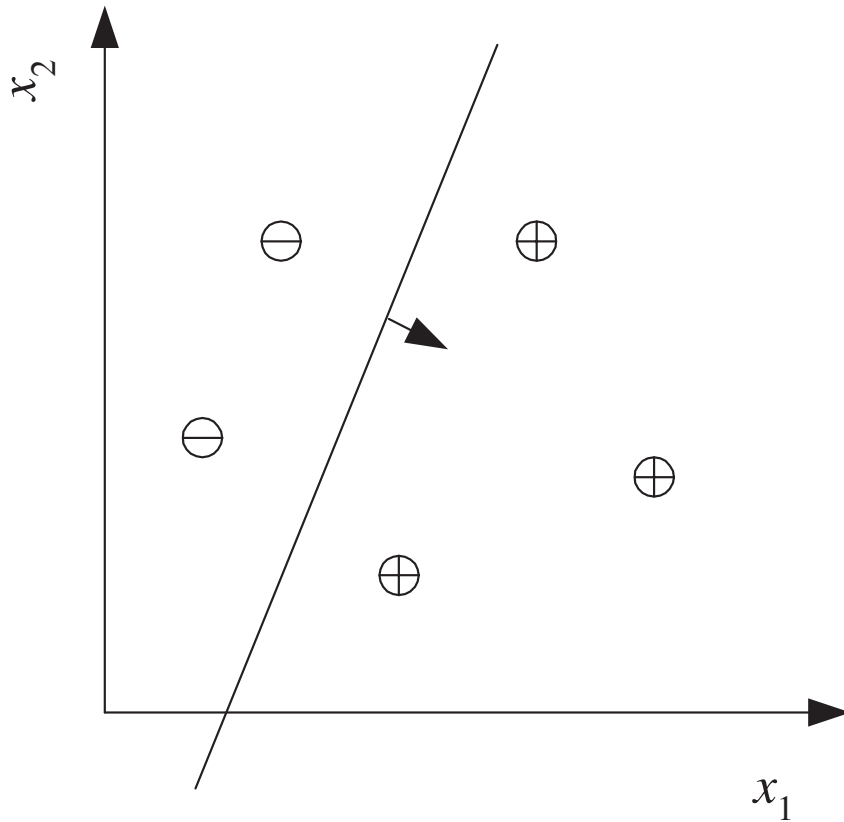
$h \in H$, between S and G is consistent and make up the version space (Mitchell, 1997)



Key Concepts

- Formulation of supervised learning: **classification** and **regression**
- **First example of classification algorithms: perceptron learning**
- Fundamental learning theory: **model complexity** and **VC dimension**
- Multiclass classifier: **K-nearest neighbors (KNN)**
- Line regression by **least squares**
- **Trade-off by model complexity and cross-validation**

Linear Classifier



$h(x) = \langle w, x \rangle + b$ is a linear classifier

$h(x) > 0$ positive
 $h(x) < 0$ negative

$h \in H, H?$



Perceptron Learning

- Perceptron algorithm, Rosenblatt, 1957.
- Initialization:

$$\mathbf{w} = \mathbf{0}$$

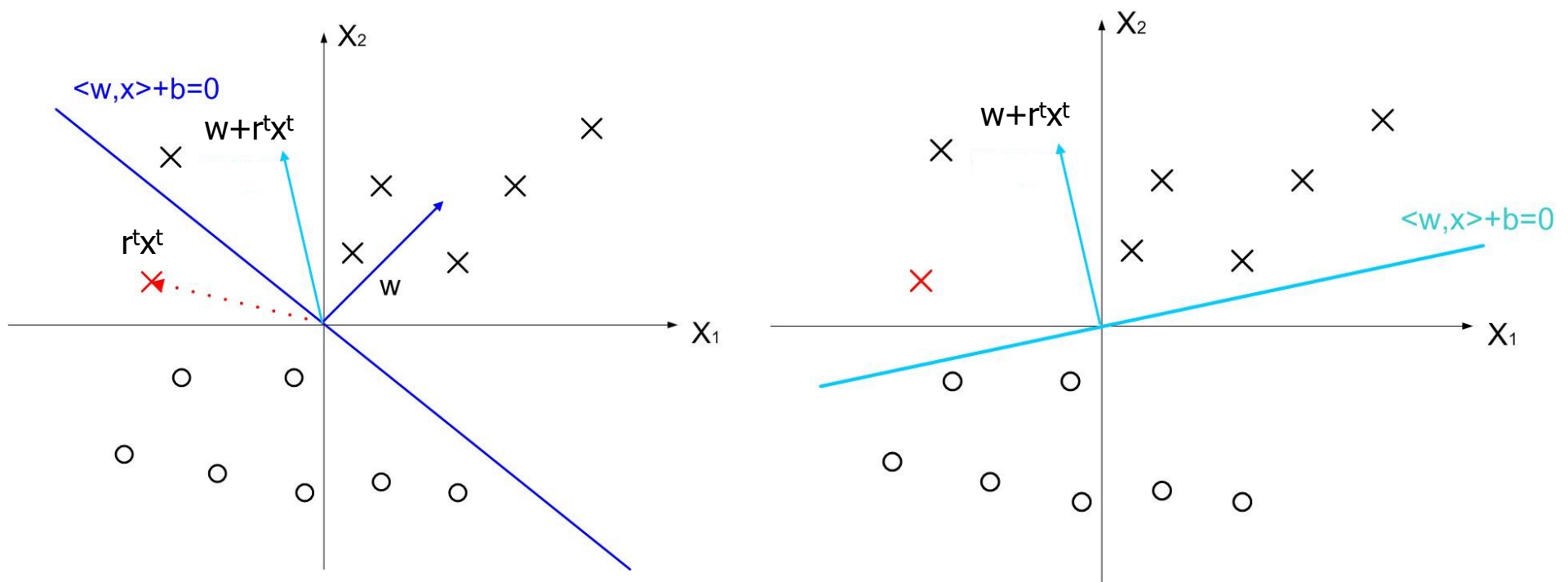
- Iterate until converge (no mistake)

for each example (\mathbf{x}^t, r^t) :

$$\text{if } (\langle \mathbf{w}, \mathbf{x}^t \rangle * r^t \leq 0)$$

$$\mathbf{w} = \mathbf{w} + r^t \mathbf{x}^t$$

Perceptron Learning

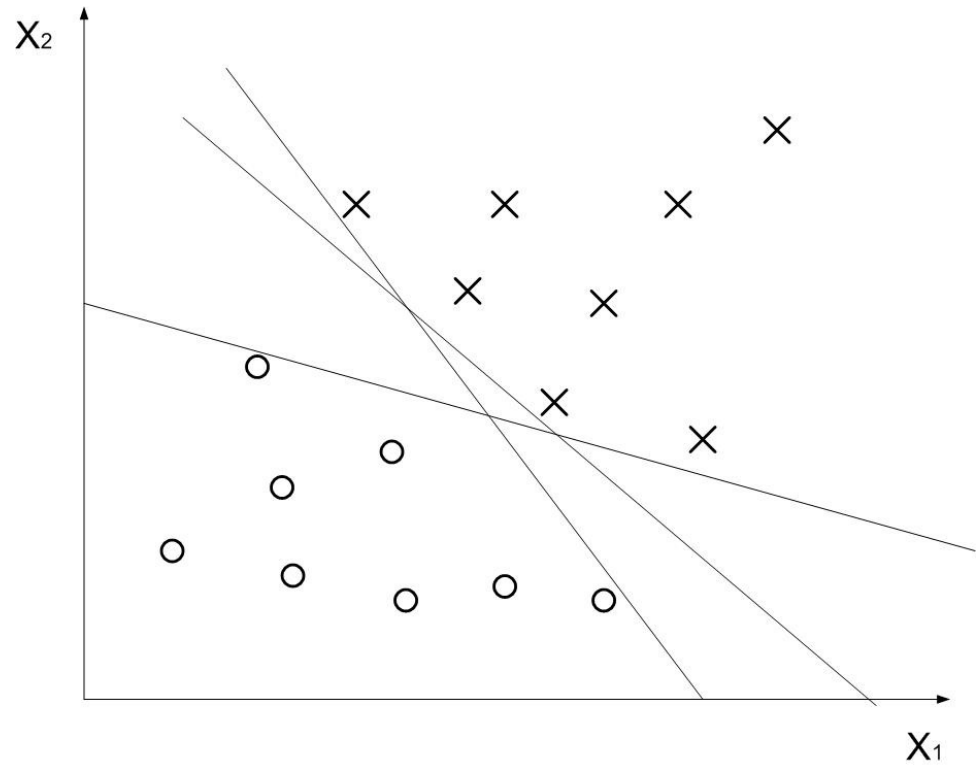
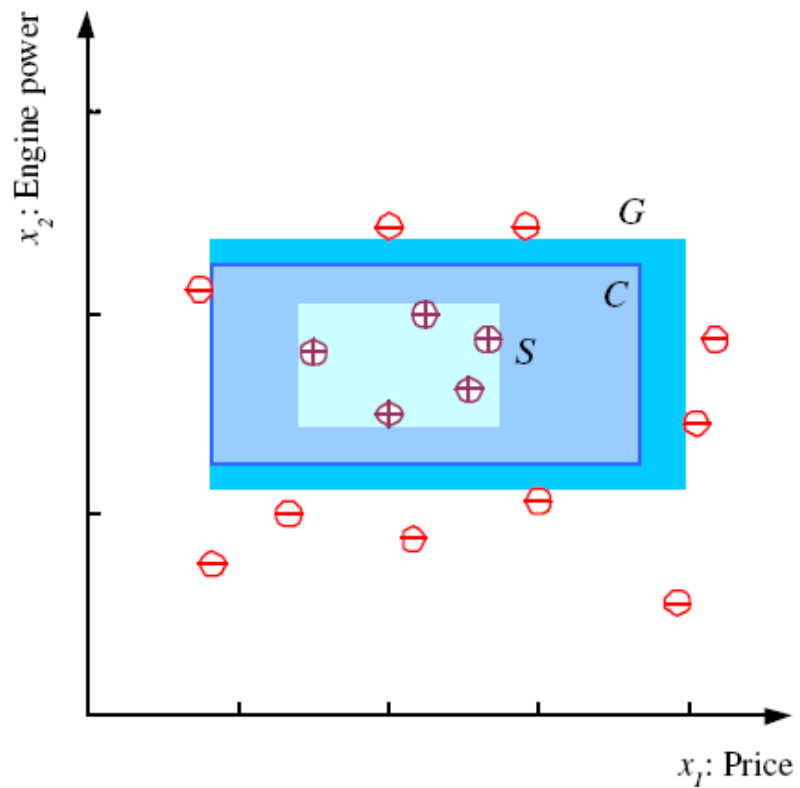




Key Concepts

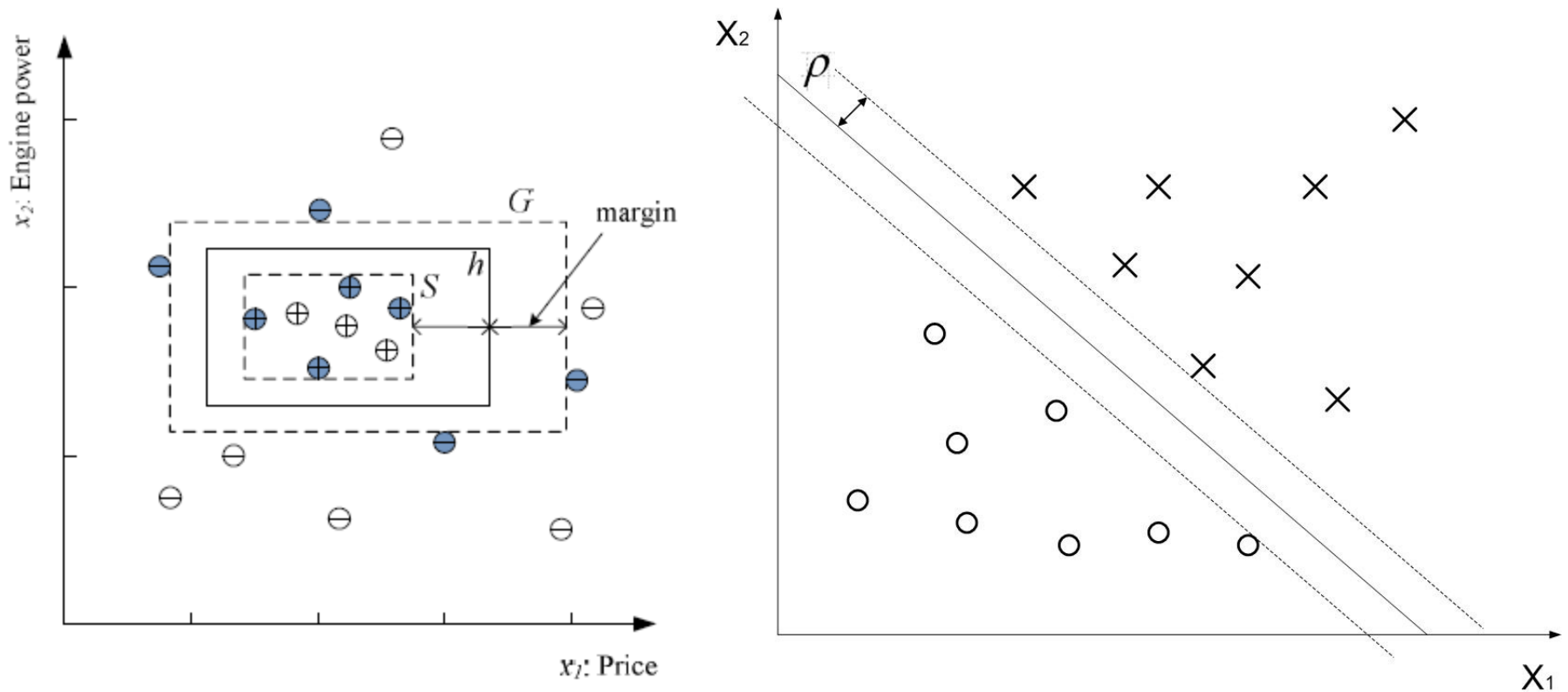
- Formulation of supervised learning: **classification** and **regression**
- First example of classification algorithms: **perceptron learning**
- **Fundamental learning theory: model complexity and VC dimension**
- Multiclass classifier: **K-nearest neighbors (KNN)**
- Line regression by **least squares**
- **Trade-off by model complexity and cross-validation**

Best in the Version Space



Margin

- Choose h with largest margin
- Why?





Model Capacity

- Different models have different capacity meaning the ability to handle more complex data.
- How to measure model capacity?
- The maximum number of data points that can be classified perfectly in any labeling.

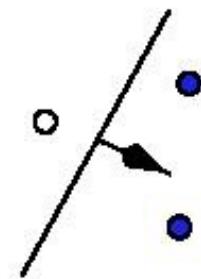
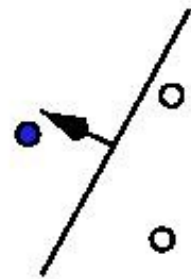
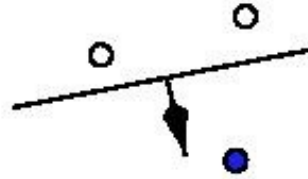
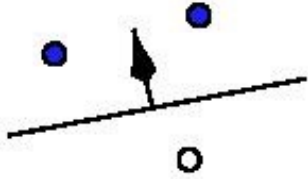
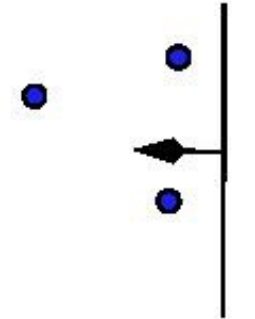
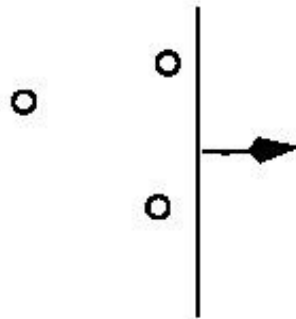
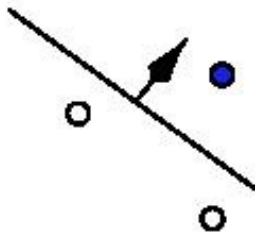
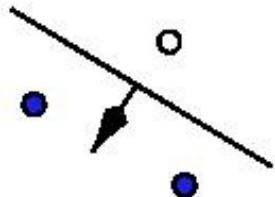
VC (Vapnik Chervonenkis) Dimension

- N points can be labeled in 2^N ways as +/-
- In a particular arrangement, \mathcal{H} shatters N if there exists $h \in \mathcal{H}$ consistent for any of the 2^N ways:

$$VC(\mathcal{H}) = N$$

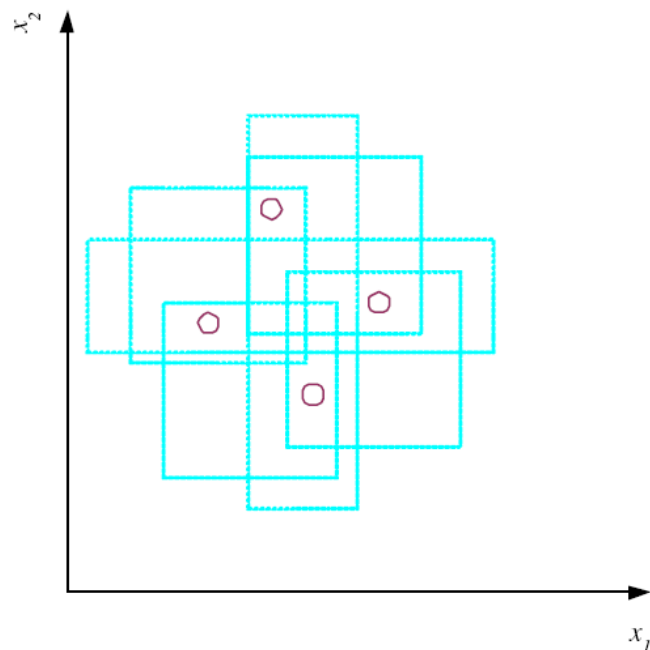
VC Dimension

How many points can be shattered by a line?



VC (Vapnik Chervonenkis) Dimension

- How about axis-aligned rectangles?





VC Summary

- The capacity of function is measured by the number of data points that can be shattered by the function.
- VC dimension can be motivated by the proof of No-Free-Lunch theorem for PAC learning theory (section 2.3 EA book).
- Rectangle classifier in 2-D space: 4.
- A line : 3.
- More ...

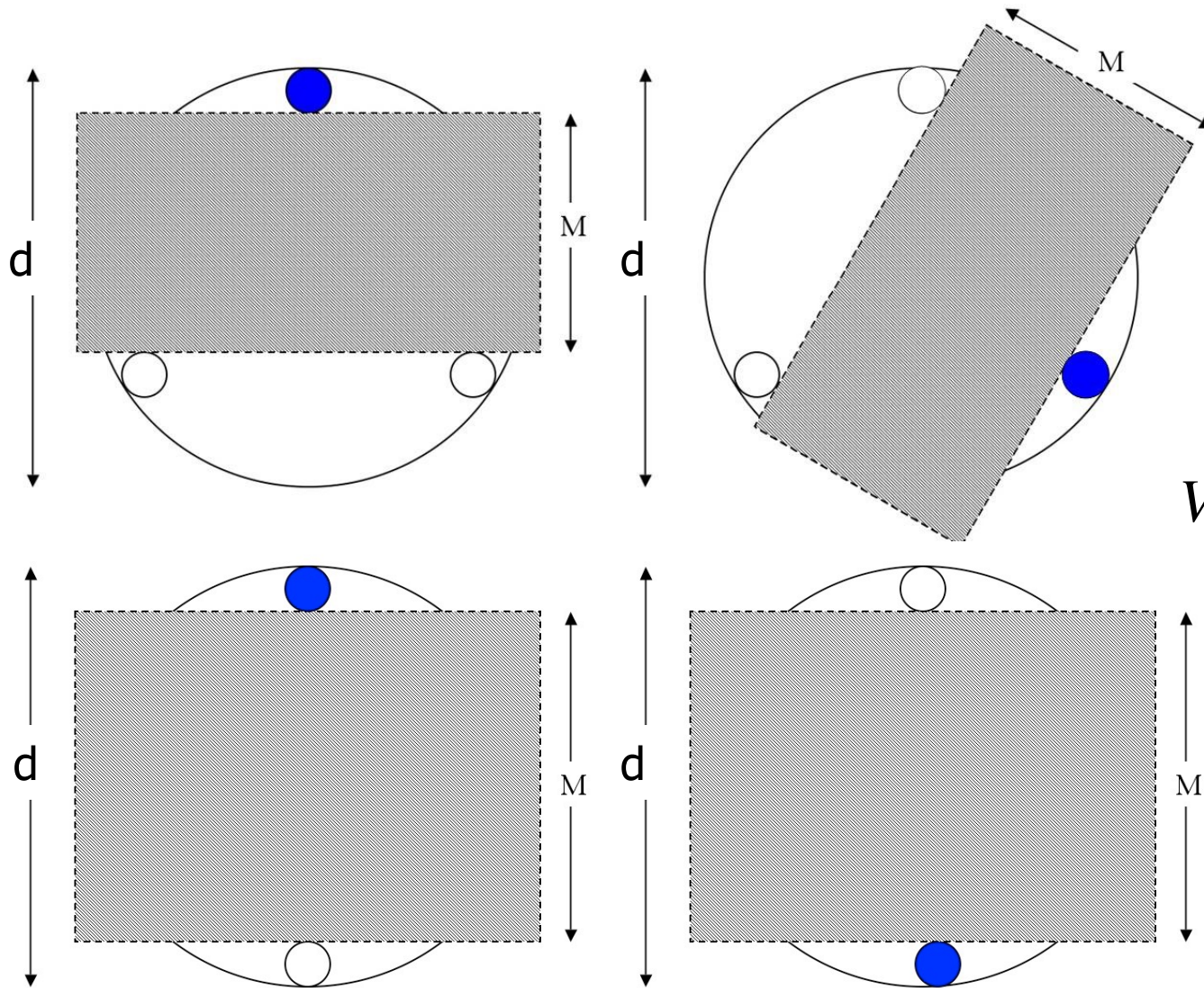


VC Dimension

- More generally, in \mathbb{R}^D space, what is the VC of a hyperplane?
- What is the VC of a triangle classifier?
- Is an algorithm that can shatter only 4 or 3 data points useful?

- How easy it is to determine the VC dimension for the hypothesis class?

VC Dimension: Why Large Margin



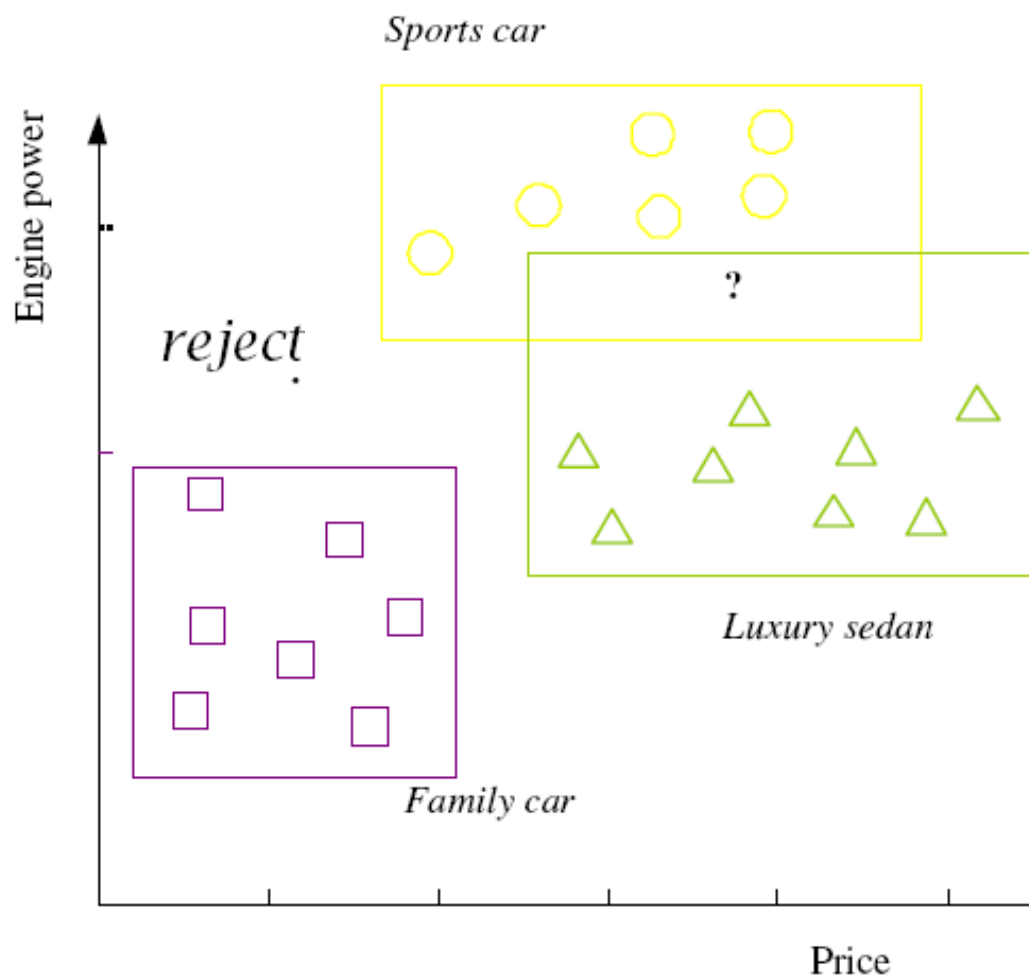
$$VC \leq \min(\text{ceil}[\frac{d^2}{M^2}], D) + 1$$



Key Concepts

- Formulation of supervised learning: **classification** and **regression**
- First example of classification algorithms: **perceptron learning**
- Fundamental learning theory: **model complexity** and **VC dimension**
- Multiclass classifier: **K-nearest neighbors (KNN)**
- Line regression by **least squares**
- **Trade-off by model complexity** and **cross-validation**

Multiple Classes, C_i $i=1, \dots, K$



$$\mathcal{X} = \{\mathbf{x}^t, \mathbf{r}^t\}_{t=1}^N$$

$$\mathbf{r}^t = C_i \text{ if } \mathbf{x}^t \in C_i$$

or

$$\mathbf{r}_i^t = \begin{cases} 1 & \text{if } \mathbf{x}^t \in C_i \\ 0 & \text{if } \mathbf{x}^t \in C_j, j \neq i \end{cases}$$

Train hypotheses

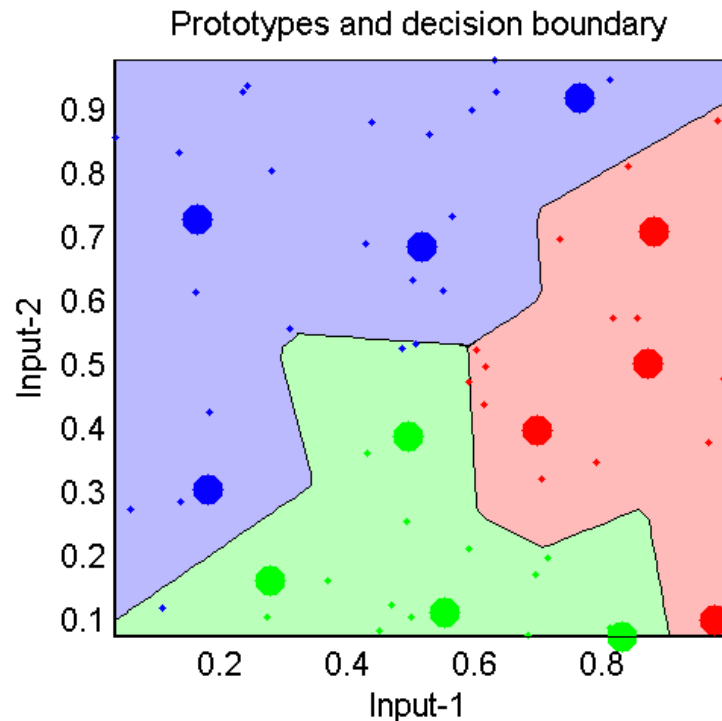
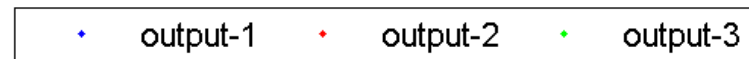
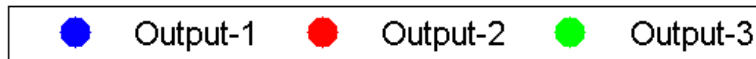
$h_i(\mathbf{x}), i = 1, \dots, K:$

$$h_i(\mathbf{x}^t) = \begin{cases} 1 & \text{if } \mathbf{x}^t \in C_i \\ 0 & \text{if } \mathbf{x}^t \in C_j, j \neq i \end{cases}$$

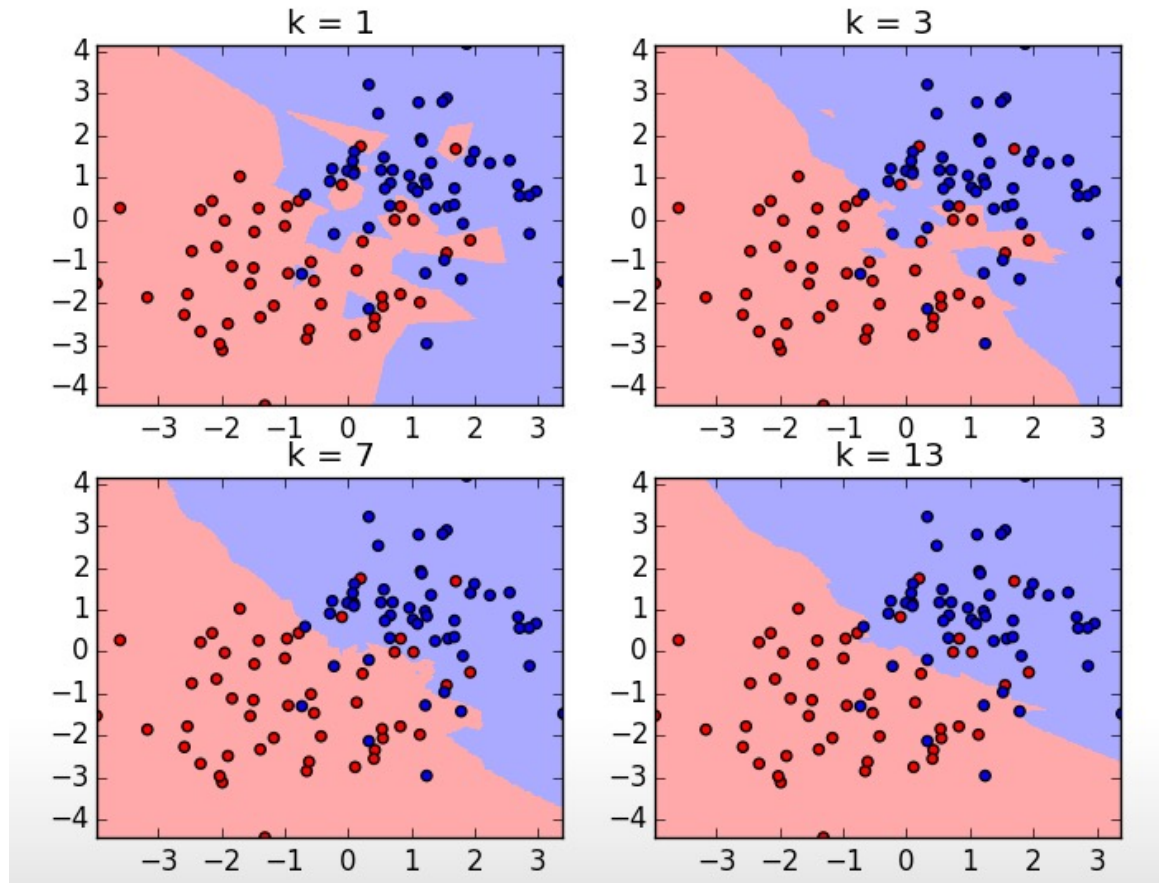
KNN Classification

■ K nearest neighbors


$$h_i(\mathbf{x}) = \left| \left\{ (\mathbf{x}^t, \mathbf{r}^t) \mid r_i^t = 1 \ \& \ \mathbf{x}^t \in N_{\mathbf{x}}^{(k)} \right\} \right|$$



How to Choose K for KNN?



- What is the VC dimension of KNN?
- Is VC proportional to the # of parameters (appeared complexity)?



Key Concepts

- Formulation of supervised learning: **classification** and **regression**
- First example of classification algorithms: **perceptron learning**
- Fundamental learning theory: **model complexity** and **VC dimension**
- Multiclass classifier: **K-nearest neighbors (KNN)**
- **Line regression by least squares**
- **Trade-off by model complexity and cross-validation**

Regression

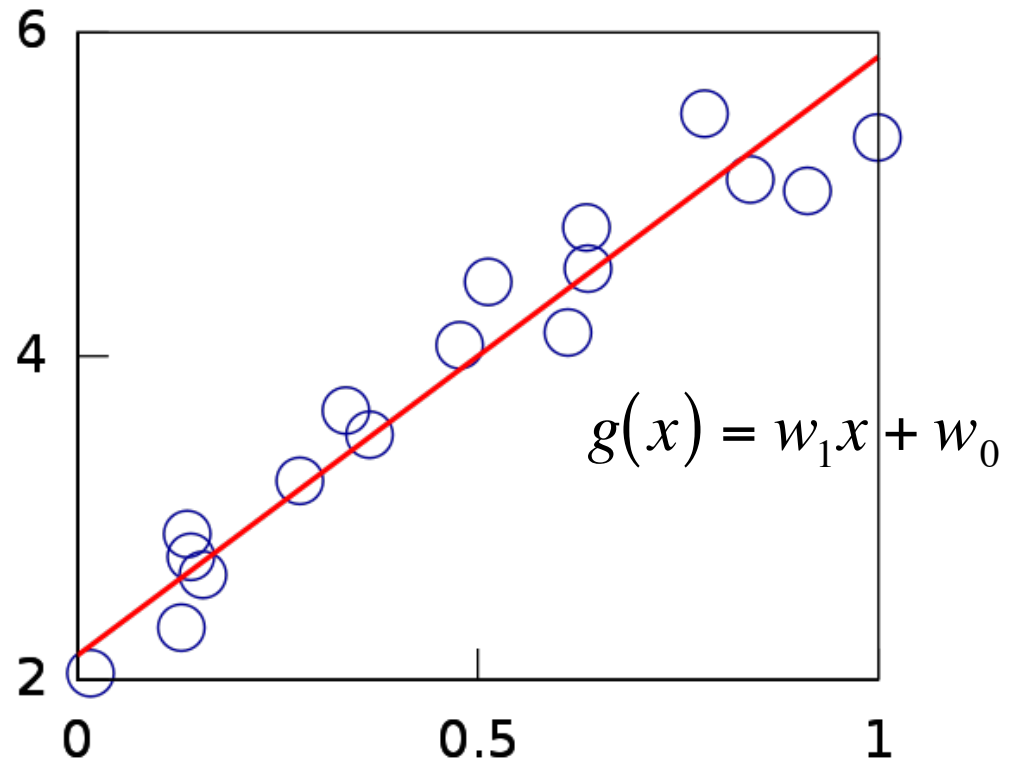
$$\mathcal{X} = \{x^t, r^t\}_{t=1}^N, \quad r^t \in \mathfrak{R}$$

$$r^t = g(x^t) + \varepsilon, \quad (\varepsilon: \text{random noise})$$

Training Error:

$$E(g | \mathcal{X}) = \frac{1}{N} \sum_{t=1}^N [r^t - g(x^t)]^2$$

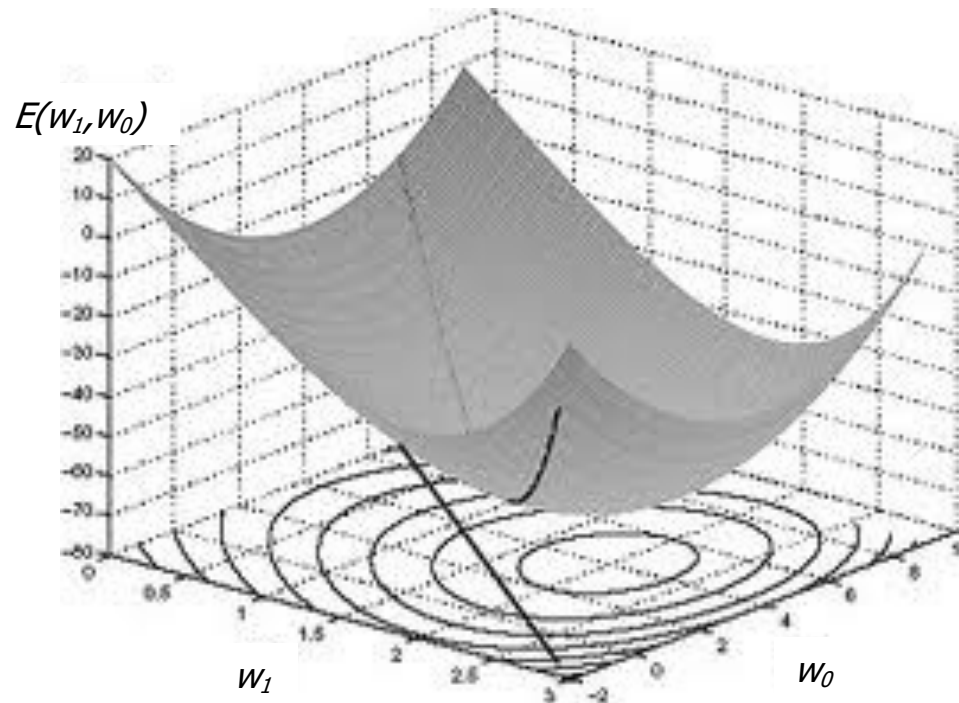
$$E(w_1, w_0 | \mathcal{X}) = \frac{1}{N} \sum_{t=1}^N [r^t - (w_1 x^t + w_0)]^2$$



Regression

- How does the error function look like?

$$E(w_1, w_0 | \mathcal{X}) = \frac{1}{N} \sum_{t=1}^N \left[r^t - (w_1 x^t + w_0) \right]^2$$



Regression

- Find the g to minimize training error

$$E(w_1, w_0 | \mathcal{X}) = \frac{1}{N} \sum_{t=1}^N \left[r^t - (w_1 x^t + w_0) \right]^2$$

$$\frac{\partial E(w_1, w_0 | \mathcal{X})}{\partial w_0} = \frac{1}{N} \sum_{t=1}^N \left[(r^t - w_1 x^t - w_0)(-1) \right] = 0$$

$$\frac{\partial E(w_1, w_0 | \mathcal{X})}{\partial w_1} = \frac{1}{N} \sum_{t=1}^N \left[(r^t - w_1 x^t - w_0)(-x^t) \right] = 0$$

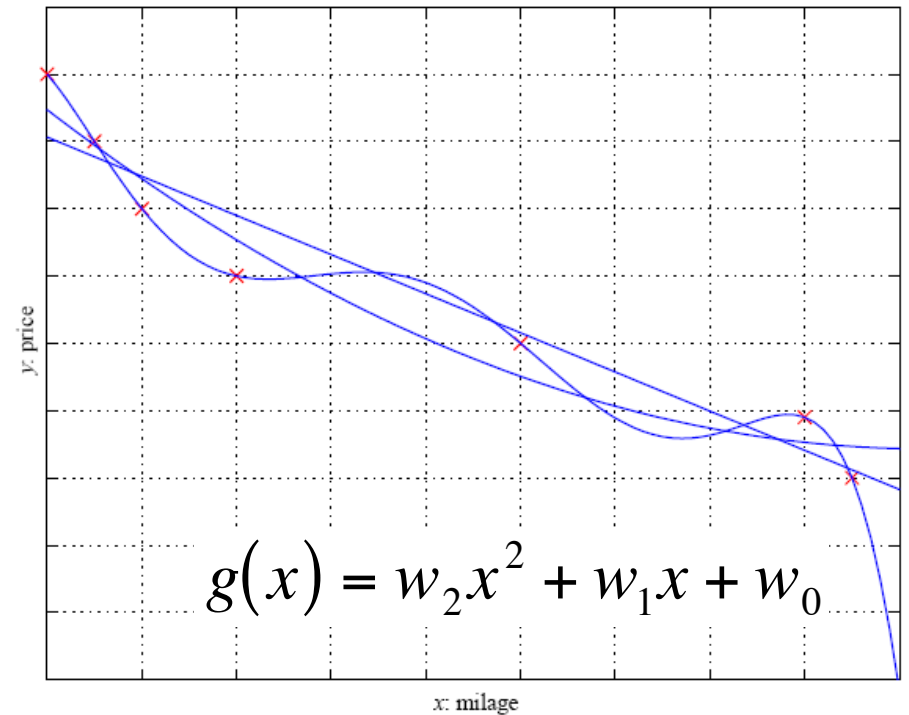
$$w_1 = \frac{\sum_t x^t r^t - N \bar{x} \bar{r}}{\sum_t (x^t)^2 - N \bar{x}^2}, w_0 = \bar{r} - w_1 \bar{x}$$

Polynomial Regression

- Is polynomial fitting very different?

$$g(x) = \sum_{i=1}^P w_p (x)^p + w_0$$

- It is the same as linear regression with a polynomial mapping.



$$g(x) = w^T x$$

$$w = [w_P, \dots, w_1, w_0]$$

$$x = [x^P, \dots, x^1, x^0]$$

Summary of Supervised Learning

1. **Model:** $g(\mathbf{x} | \theta)$ $g(x) = w_1 x + w_0$

2. **Loss function:** $E(\theta | \mathcal{X}) = \sum_t L(r^t, g(\mathbf{x}^t | \theta))$

$$E(h | \mathcal{X}) = \sum_{t=1}^N 1(h(\mathbf{x}^t) \neq r^t) \quad E(g | \mathcal{X}) = \frac{1}{N} \sum_{t=1}^N [r^t - g(x^t)]^2$$

3. **Optimization procedure:** $\theta^* = \arg \min_{\theta} E(\theta | \mathcal{X})$

Algorithms: KNN, perceptron, linear regression



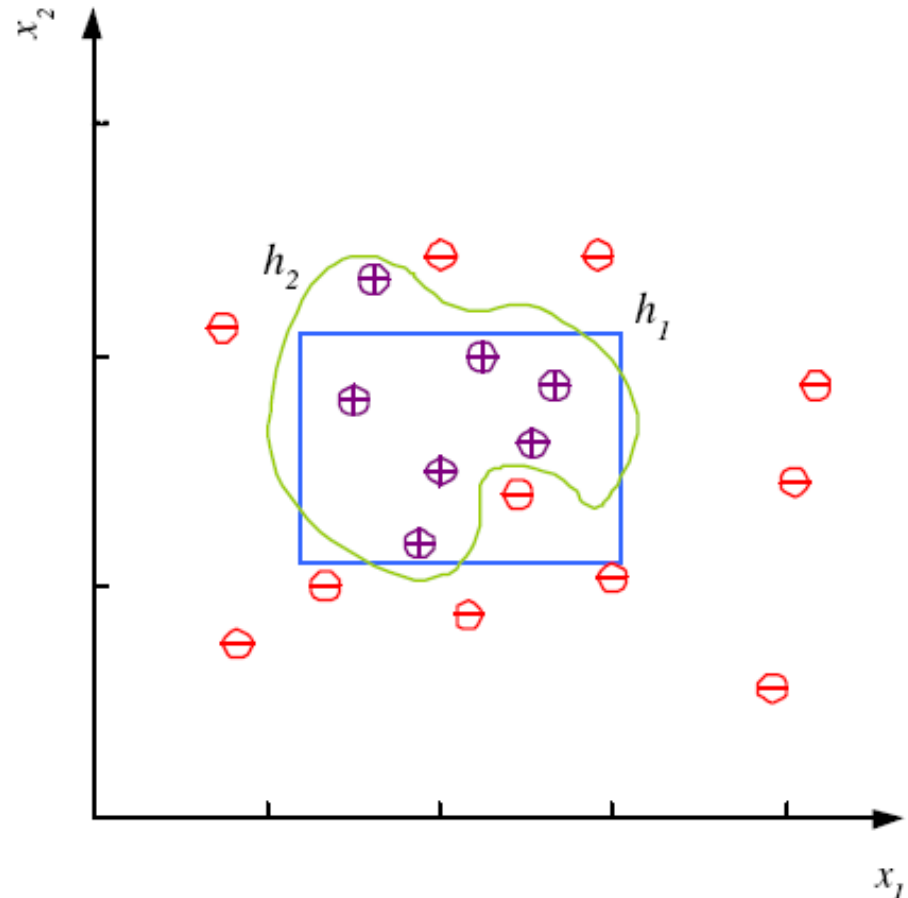
Key Concepts

- Formulation of supervised learning: **classification** and **regression**
- First example of classification algorithms: **perceptron learning**
- Fundamental learning theory: **model complexity** and **VC dimension**
- Multiclass classifier: **K-nearest neighbors (KNN)**
- Line regression by **least squares**
- **Trade-off by model complexity and cross-validation**

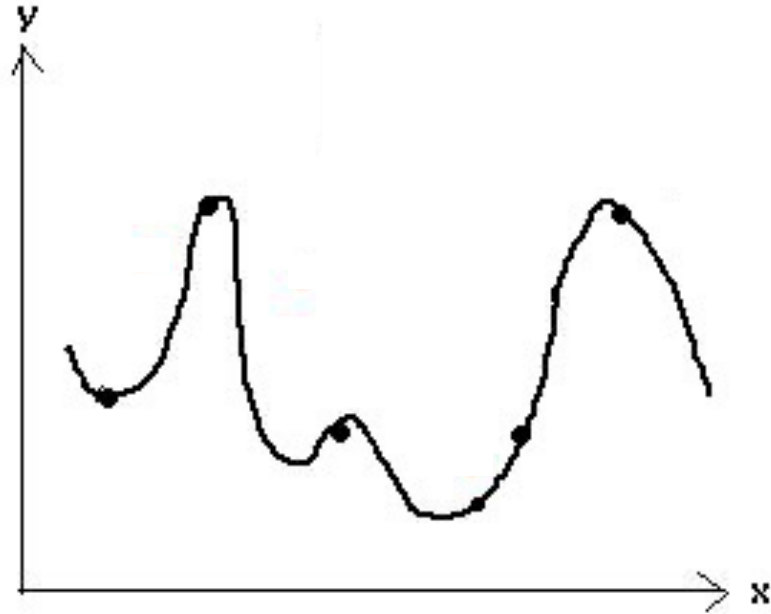
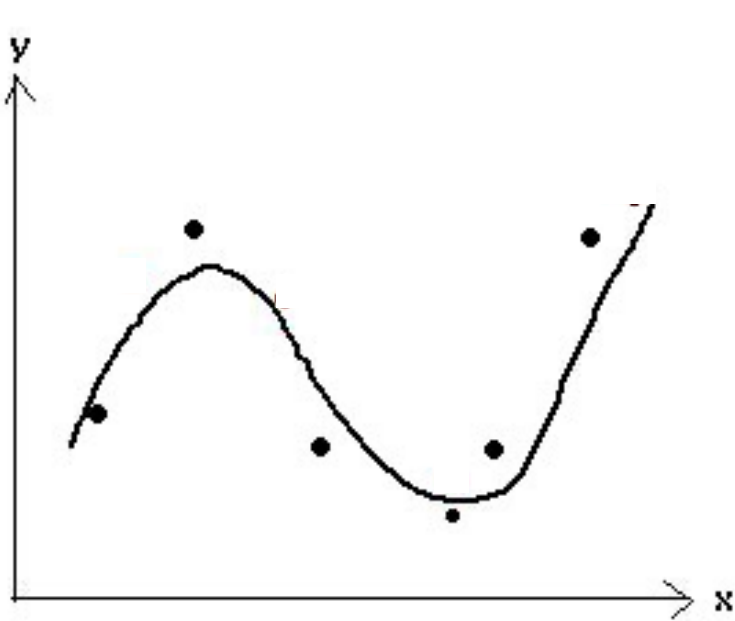
Noise and Model Complexity

Data is not perfect

- Data recording might not be perfect (shifted data points)
- Wrong labeling of the data
- There might be additional unobservable hidden variables.



Noise and Model Complexity



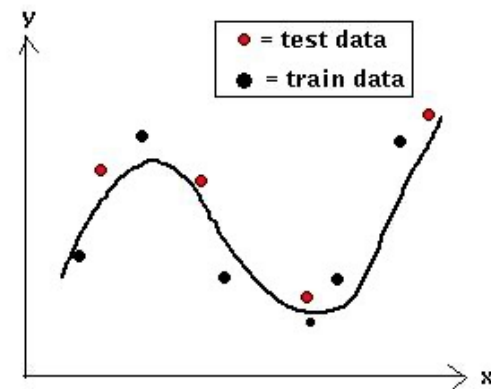
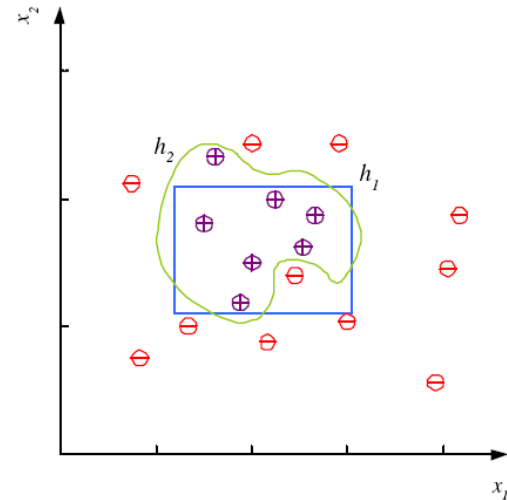
Options:

- Simple model with training errors
- Complex model with no training error

Noise and Model Complexity

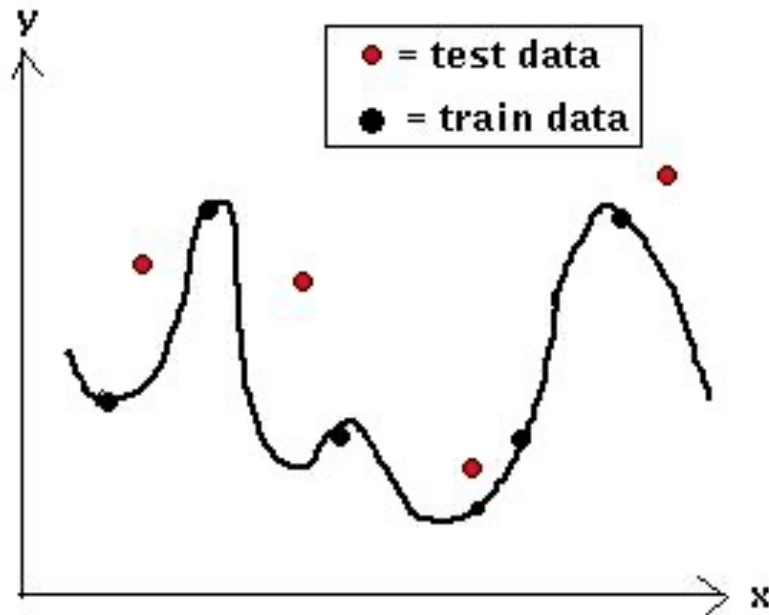
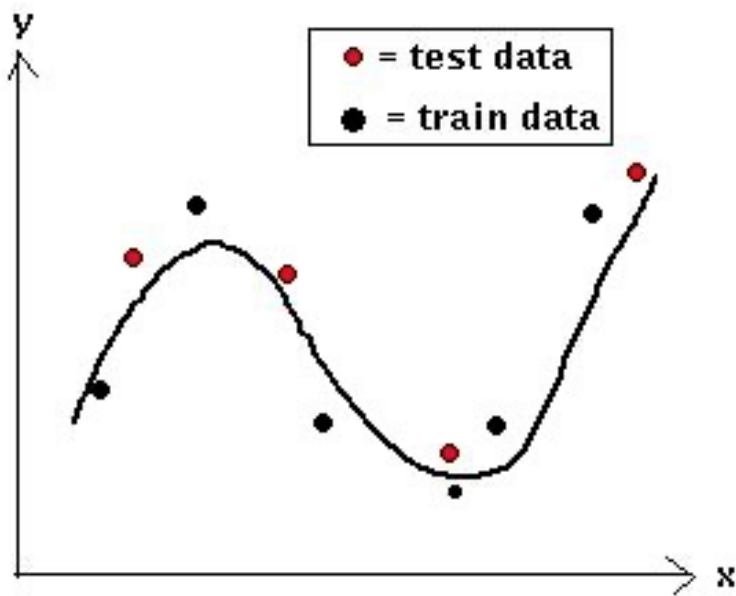
Given similar training error
use the simpler one

- Simpler to use (lower computational complexity)
- Easier to train (lower space complexity)
- Easier to explain (more interpretable)
- Generalizes better (lower variance - Occam's razor)



Generalization and Overfitting

- **Generalization:** How well a model performs on new data
- **Overfitting:** \mathcal{H} more complex than C or f
- **Underfitting:** \mathcal{H} less complex than C or f





Model Selection & Generalization

- Learning is an **ill-posed problem**; data is not sufficient to find a unique solution
- Given d binary inputs, there are at most 2^D samples, and 2^{2^D} binary functions
- Each sample eliminates half of the functions;
- Thus, N samples leaves $2^{2^D - N}$ viable functions

- Not possible to check all functions. Need for **inductive bias**, assumptions about \mathcal{H}



Cross-Validation

- To better estimate generalization error, we need data unseen during training. We split the data as
 - Training set (50%)
 - Validation set (25%)
 - Test set (25%)
- Resampling when there is few data

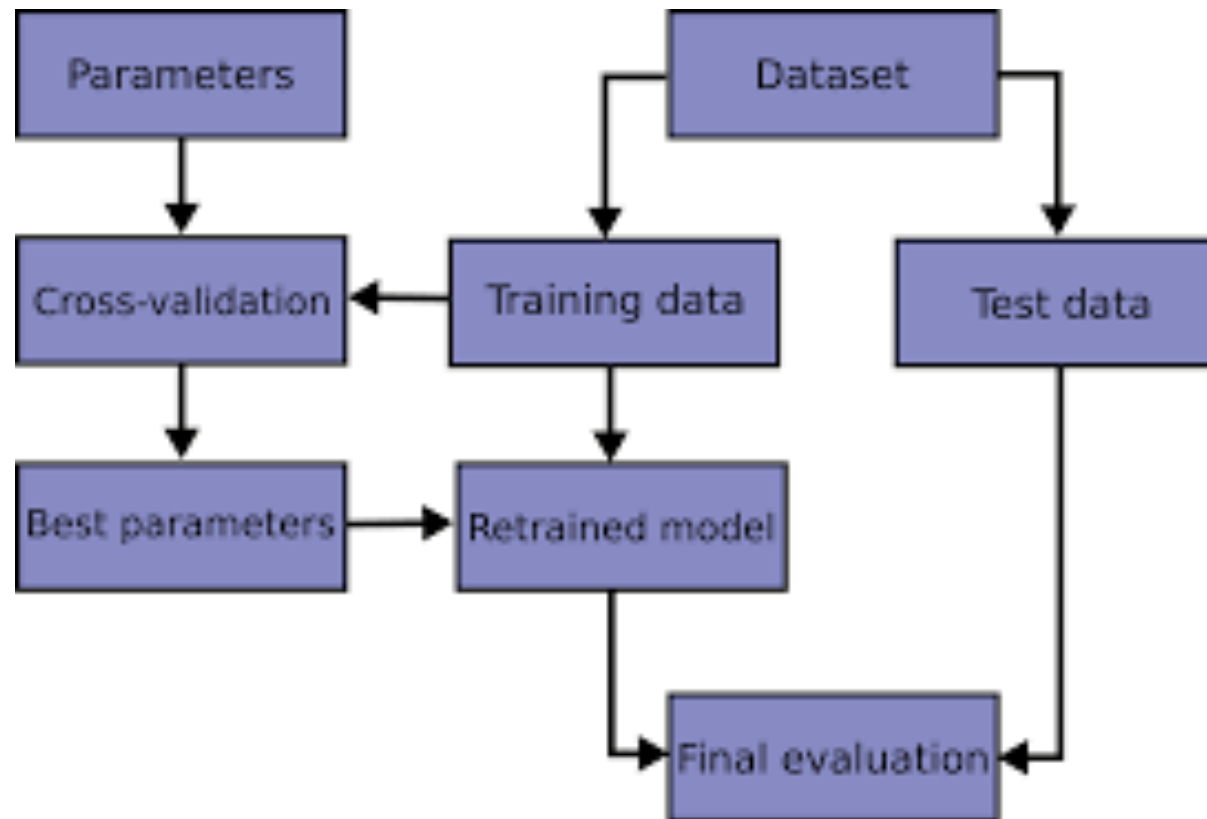
Cross-Validation

4-fold validation (k=4)



<https://www.mathworks.com/discovery/cross-validation.html>

Cross-Validation (good practice)



https://scikit-learn.org/stable/modules/cross_validation.html