



Machine learning

# Machine learning ≈ build a mapping function

- ▶ Speech Recognition       $f(\text{Speech waveform}) = \text{"Hello"}$
  - ▶ Image Identification       $f(\text{Image of a cat}) = \text{"Cat"}$
  - ▶ Go       $f(\text{Go board position}) = \text{"5-5"} \quad (\text{position})$
  - ▶ Dialogue system       $f(\text{"Hi"}) = \text{"The weather is nice today"}$
- User input                          Machine

# Why we need "machine learning"?

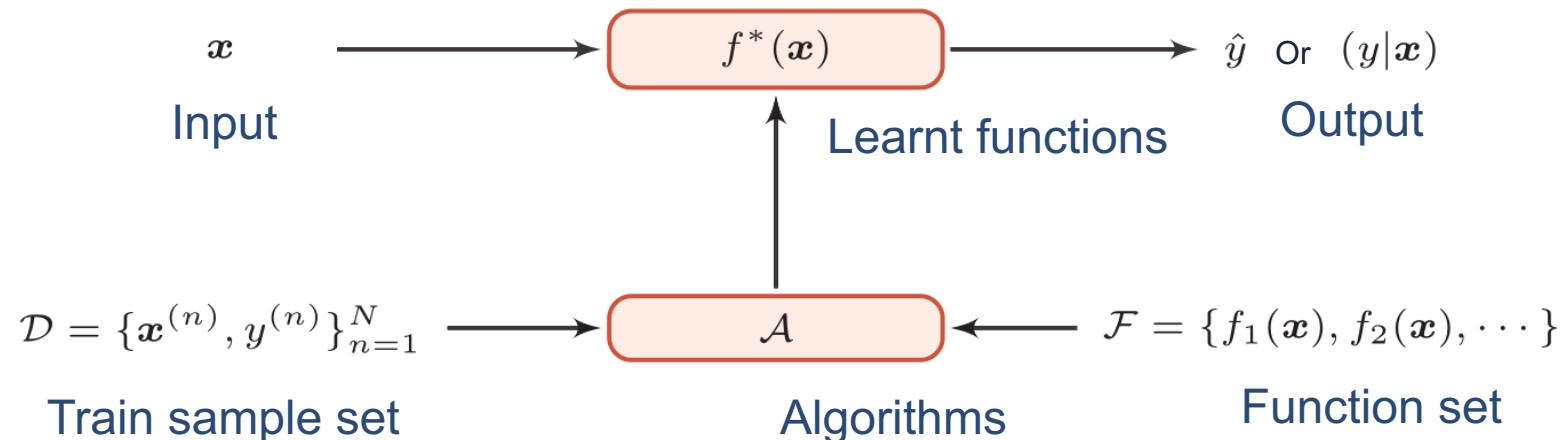
- ▶ Real world problems are too complicated
  - It is difficult to implement manually through rules



2	6	8	9	3	4	7	5	6
3	4	7	9	5	5	6	7	2
5	8	7	0	9	4	3	5	4
5	2	3	4	9	5	6	7	8

# What is machine learning?

- ▶ Machine learning: Through algorithms, machines can learn rules from large amounts of data to make decisions on new samples.
  - Law: Decision (prediction) function



# The three elements of machine learning

---

## ► Model

- Linear method:  $f(\mathbf{x}, \theta) = \mathbf{w}^T \mathbf{x} + b$
- Generalized linear method :  $f(\mathbf{x}, \theta) = \mathbf{w}^T \phi(\mathbf{x}) + b$ 
  - If  $\phi(\mathbf{x})$  is a learnable nonlinear basis function,  $f(\mathbf{x}, \theta)$  is equivalent to a neural network.

## ► Study criteria

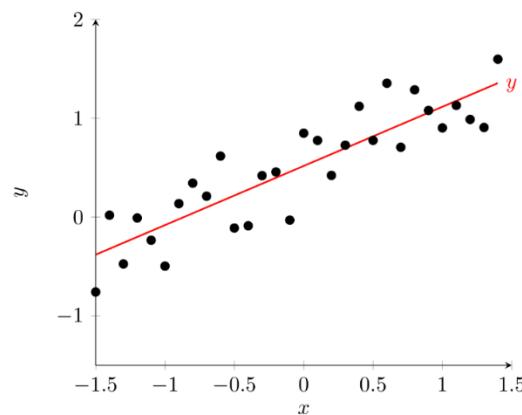
- Expected risk

$$\mathcal{R}(f) = \mathbb{E}_{(\mathbf{x}, y) \sim p(\mathbf{x}, y)} [\mathcal{L}(f(\mathbf{x}), y)],$$

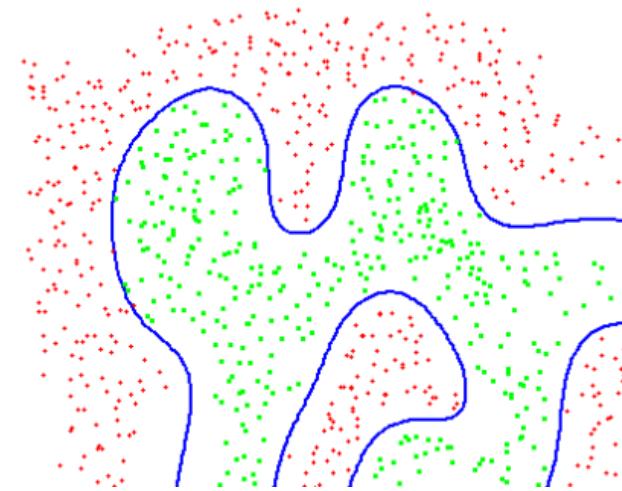
## ► Optimization

- Gradient descent

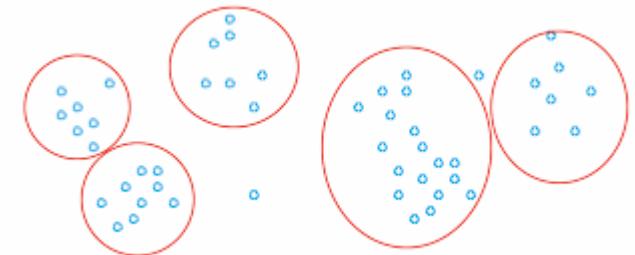
# Common machine learning problems and models



regression



classification



clustering

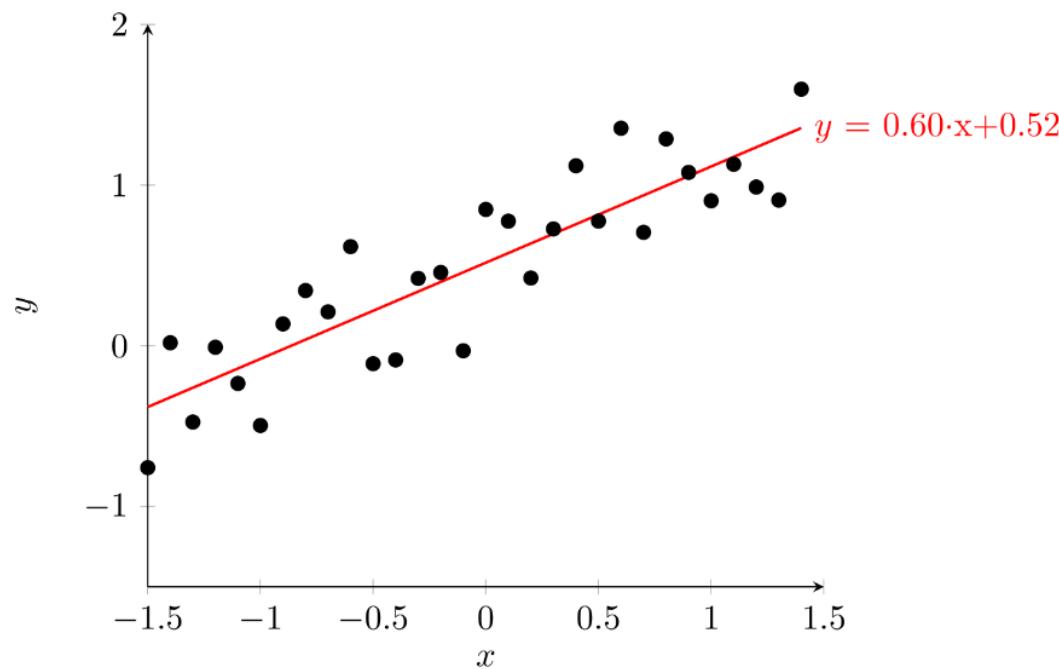
# Model

---

## ► Take Linear Regression as an example

- Model:

$$f(\mathbf{x}; \mathbf{w}, b) = \mathbf{w}^T \mathbf{x} + b$$



# Study criteria

---

## ► Loss function

- 0-1 Loss function

$$\mathcal{L}(y, f(x, \theta)) = \begin{cases} 0 & \text{if } y = f(x, \theta) \\ 1 & \text{if } y \neq f(x, \theta) \end{cases}$$

- Square loss function

$$\mathcal{L}(y, \hat{y}) = (y - f(x, \theta))^2$$

# Study criteria

---

- ▶ If expected risk is unknown, we can approximate it by empirical risk
  - Train data:  $\mathcal{D} = \{x^{(n)}, y^{(n)}\}, i \in [1, N]$
- $$\mathcal{R}_{\mathcal{D}}^{emp}(\theta) = \frac{1}{N} \sum_{n=1}^N \mathcal{L}(y^{(n)}, f(x^{(n)}, \theta))$$
- ▶ Minimize experience risk
  - After selecting the appropriate risk function, we look for a parameter  $\theta^*$  to minimize the empirical risk function.

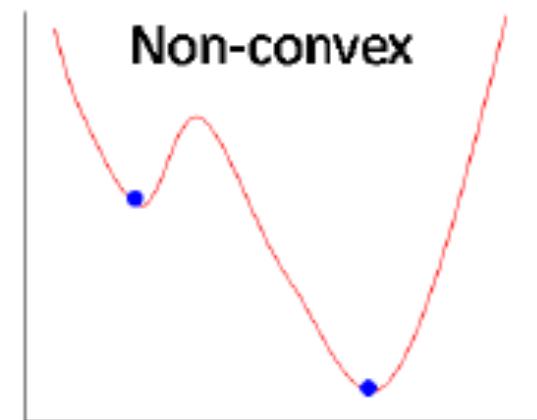
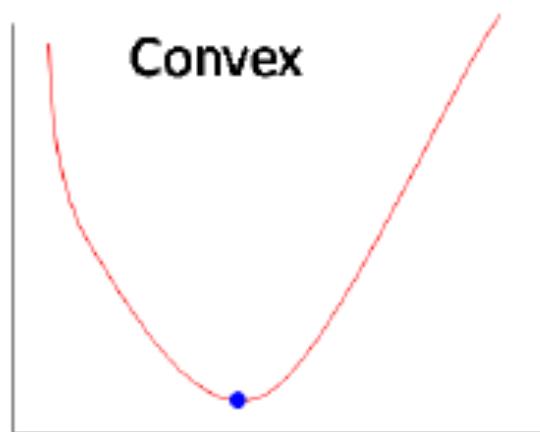
$$\theta^* = \arg \min_{\theta} \mathcal{R}_{\mathcal{D}}^{emp}(\theta)$$

- ▶ Machine learning problem turned into an optimization problem

# Optimization problem

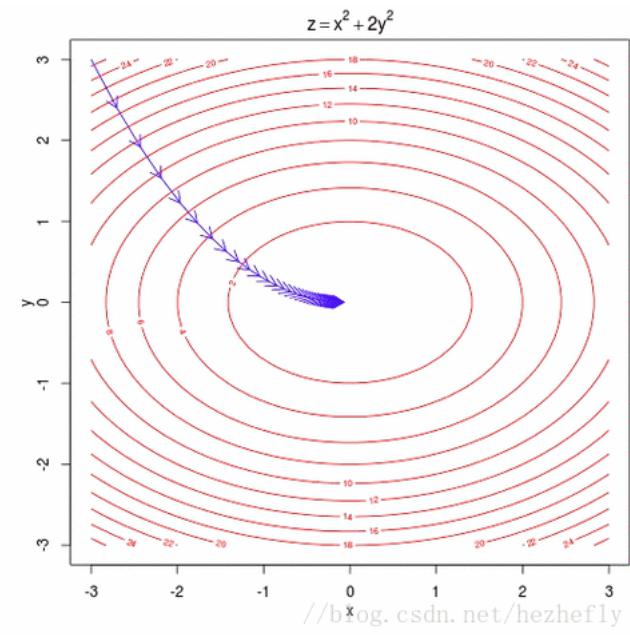
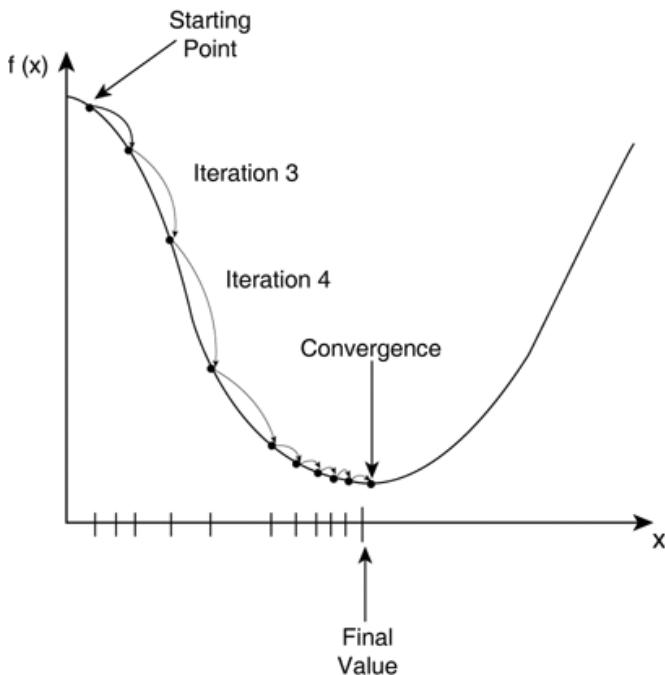
---

- ▶ Machine learning problem turned into an optimization problem



$$\min_{\mathbf{x}} f(\mathbf{x})$$

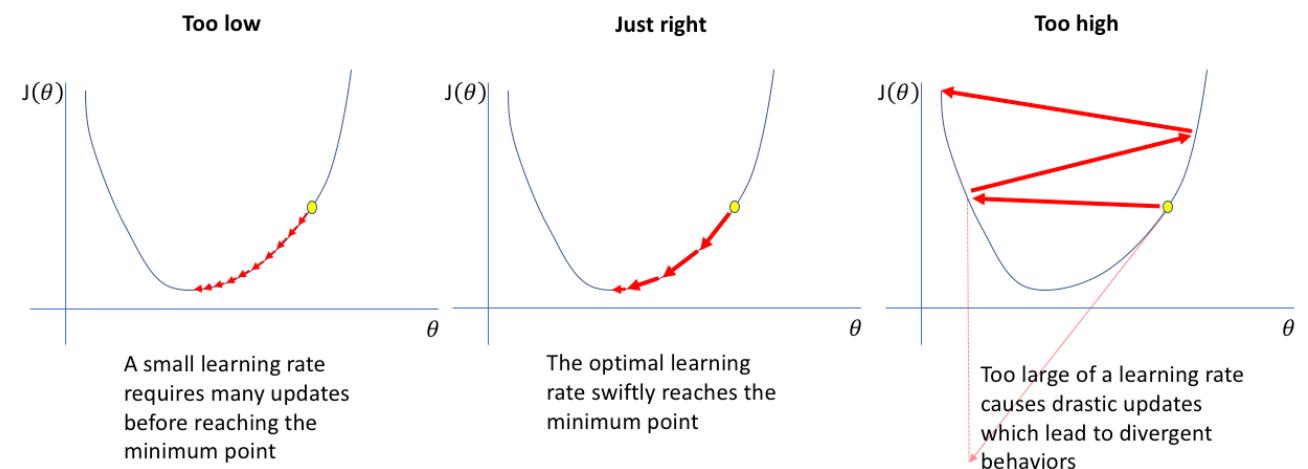
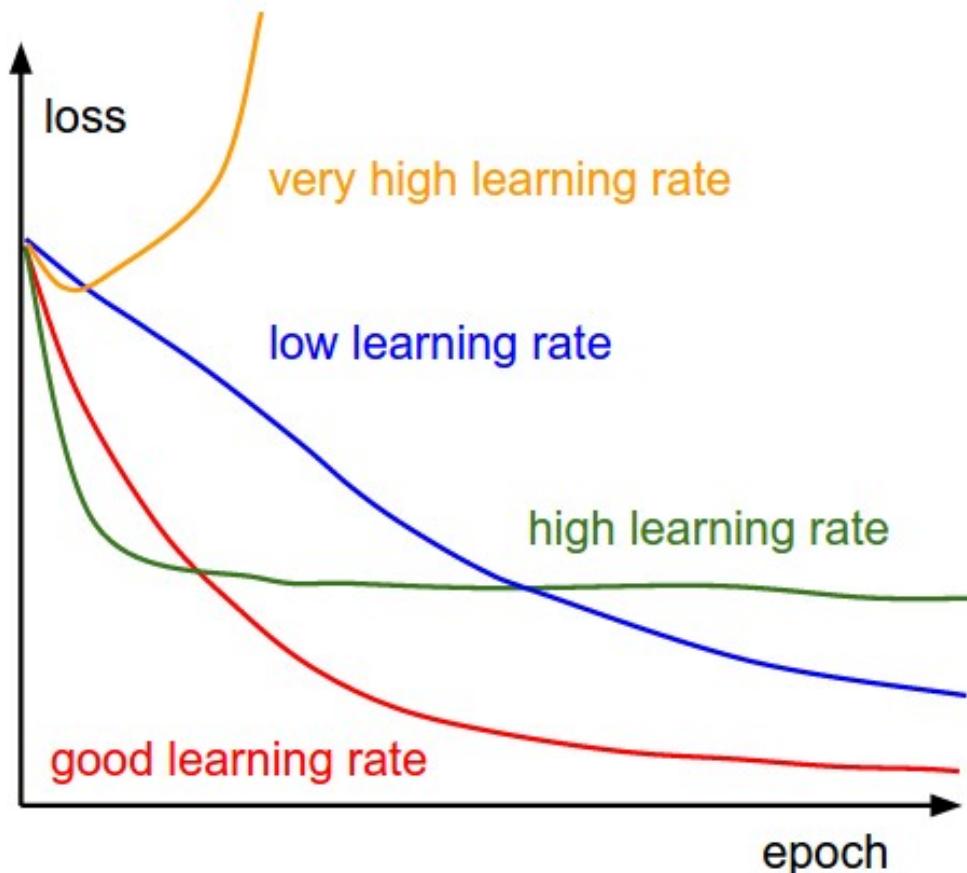
# Gradient Descent



$$\begin{aligned}\theta_{t+1} &= \theta_t - \alpha \frac{\partial \mathcal{R}(\theta)}{\partial \theta_t} \\ &= \theta_t - \alpha \frac{1}{N} \sum_{i=1}^N \frac{\partial \mathcal{L}(\theta_t; x^{(i)}, y^{(i)})}{\partial \theta}.\end{aligned}$$

Search step  $\alpha$  is also called **Learning Rate**

# Learning rate is a very important hyperparameter!



# Stochastic Gradient Descent

---

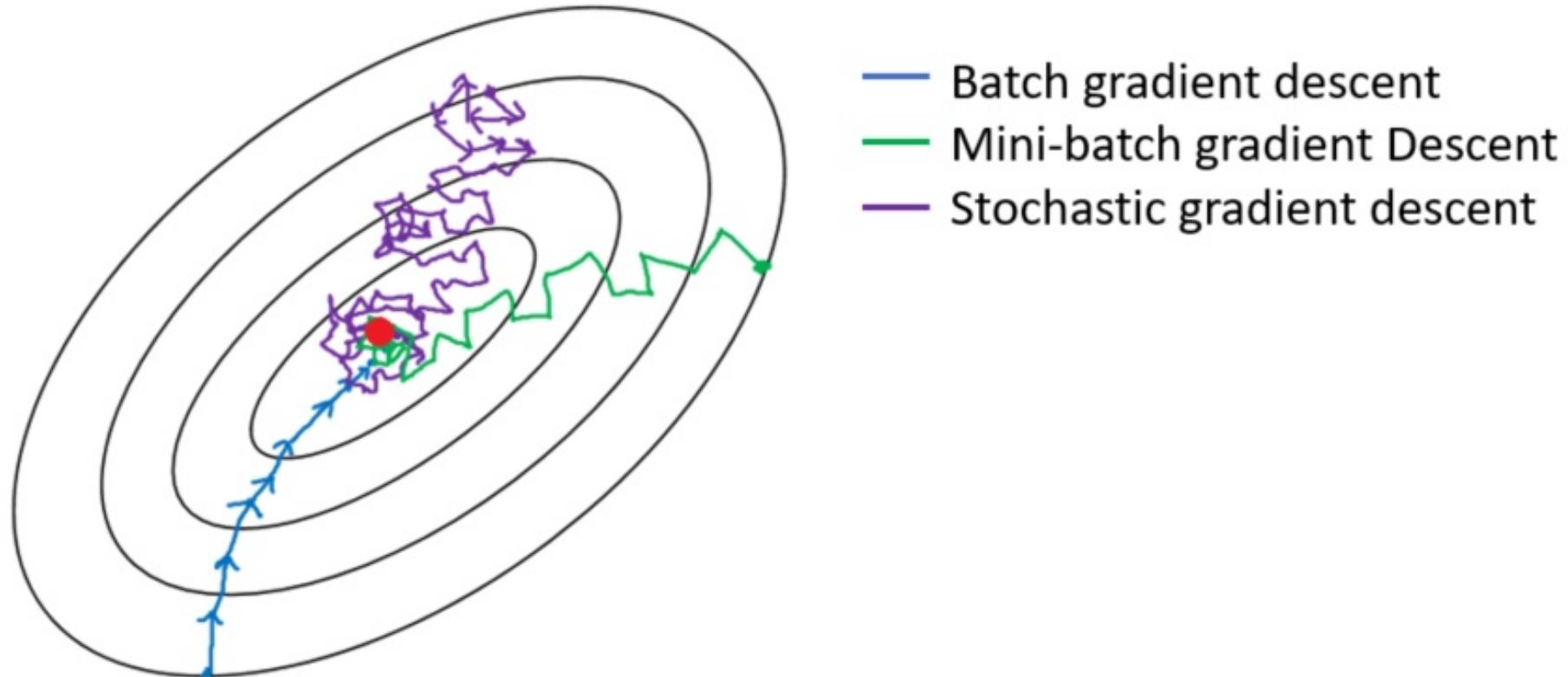
- ▶ Stochastic Gradient Descent (SGD) is also called incremental gradient descent, and each sample is updated

$$\theta_{t+1} = \theta_t - \alpha \frac{\partial \mathcal{L}(\theta_t; x^{(t)}, y^{(t)})}{\partial \theta},$$

- ▶ Mini-Batch Stochastic Gradient Descent
  - Use  $b$  samples for each update. In fact, batch gradient descent is a compromise method.

# Stochastic Gradient Descent

---

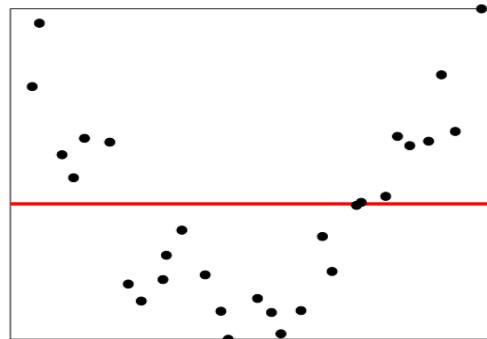


# Machine learning = optimization?

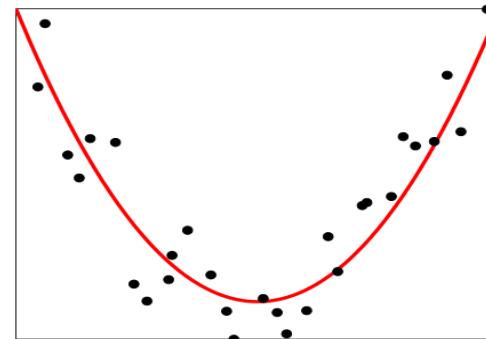
Machine learning = optimization?

NO!

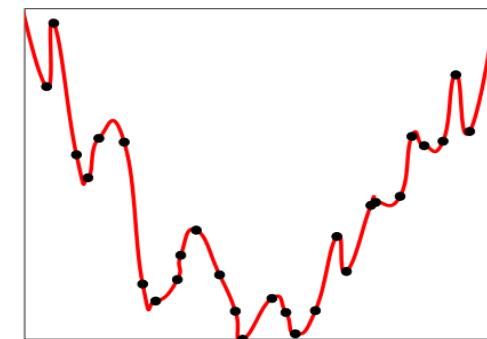
Underfit



Normal



Overfit



Overfitting: The principle of empirical risk minimization can easily lead to a low error rate of the model on the training set, but a high error rate on the unknown data.

Over-fitting problems are often caused by reasons such as less training data and noise.

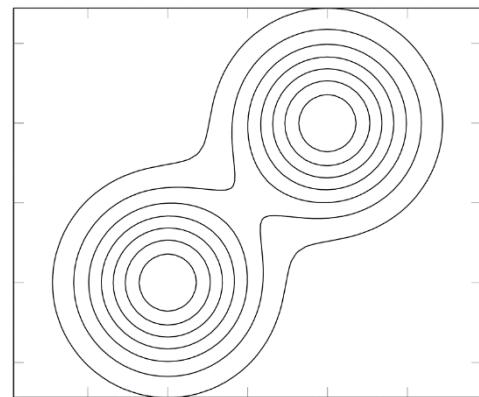
# Generalization error

---

## Expected risk

$$\mathcal{R}(f) = \mathbb{E}_{(\mathbf{x}, y) \sim p(\mathbf{x}, y)} [\mathcal{L}(f(\mathbf{x}), y)],$$

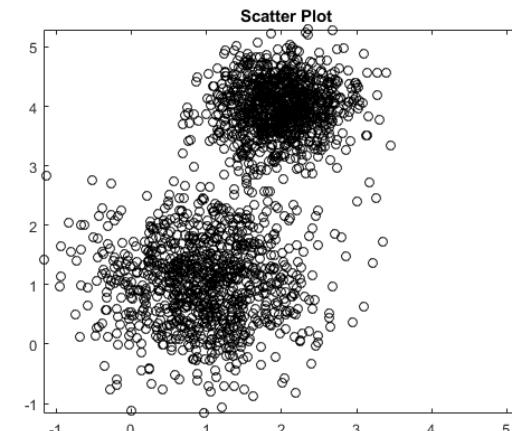
Real distribution



$\neq$

## Empirical risk

$$\mathcal{R}_{\mathcal{D}}^{emp}(\theta) = \frac{1}{N} \sum_{n=1}^N \mathcal{L}(y^{(n)}, f(x^{(n)}, \theta))$$



$$\mathcal{G}_{\mathcal{D}}(f) = \mathcal{R}(f) - \mathcal{R}_{\mathcal{D}}^{emp}(f)$$

Generalization error

# How to reduce generalization errors?

---

Optimization

Minimal empirical risk

Regularization

Reduce model complexity



# Regularization

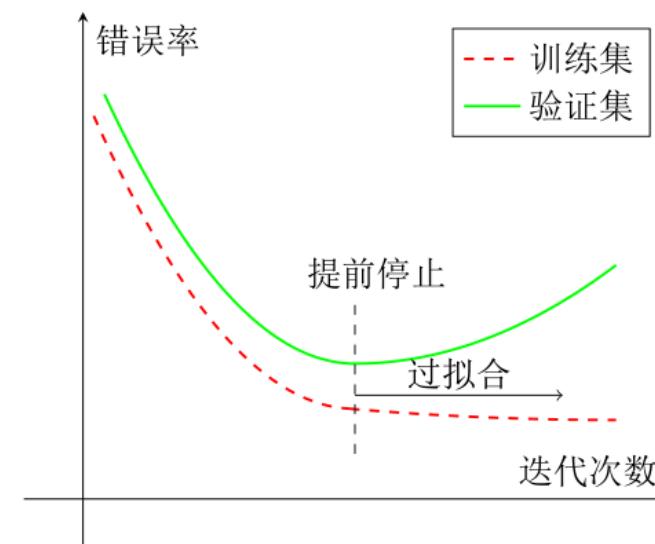
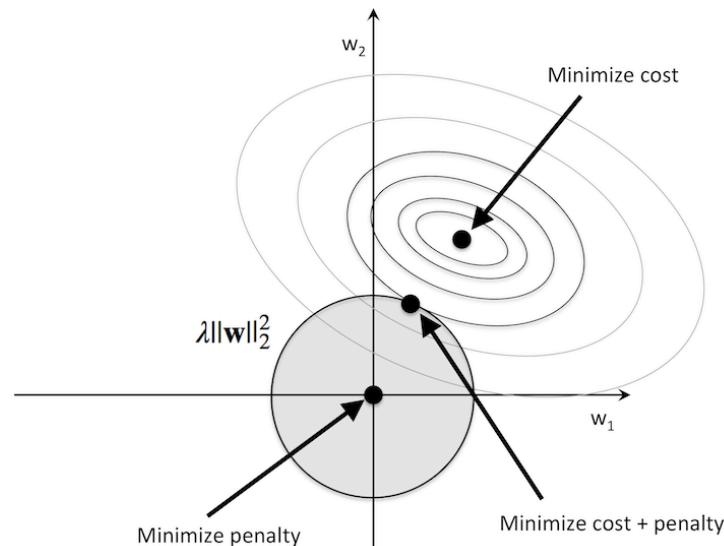
All regularization will harm the optimization

Increase optimization constraints

Interfere optimization process

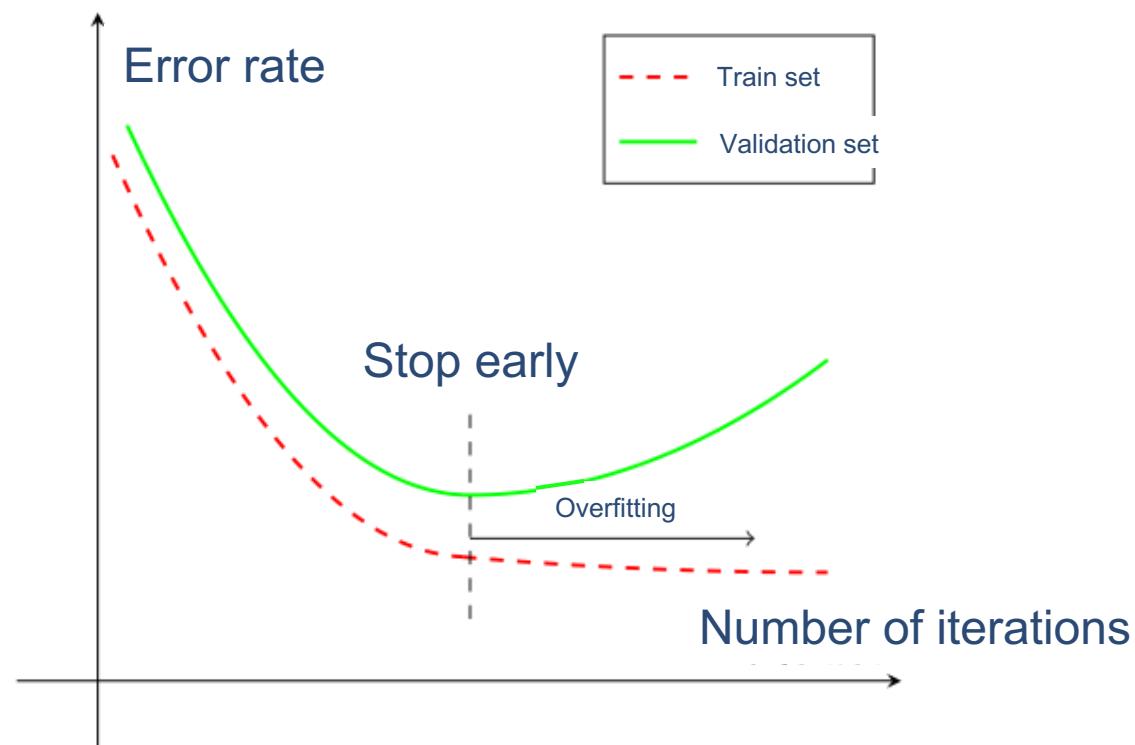
L1/L2 constraints, data enhancement

Weight decay, stochastic gradient descent, early stop



## Stop early

- We use a Validation Dataset to test whether the parameters of each iteration are optimal on the validation set. If the error rate on the validation set no longer drops, stop the iteration.



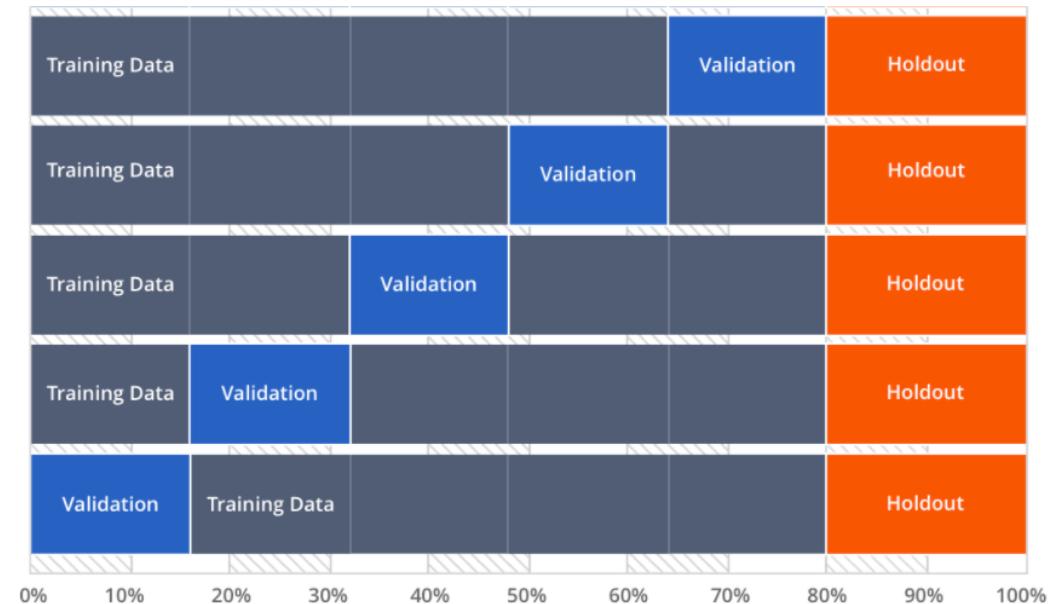
# Cross Validation

- ▶ Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample.
- ▶ Repeated use of data, split the sample data obtained, and combine them into different training sets and test sets



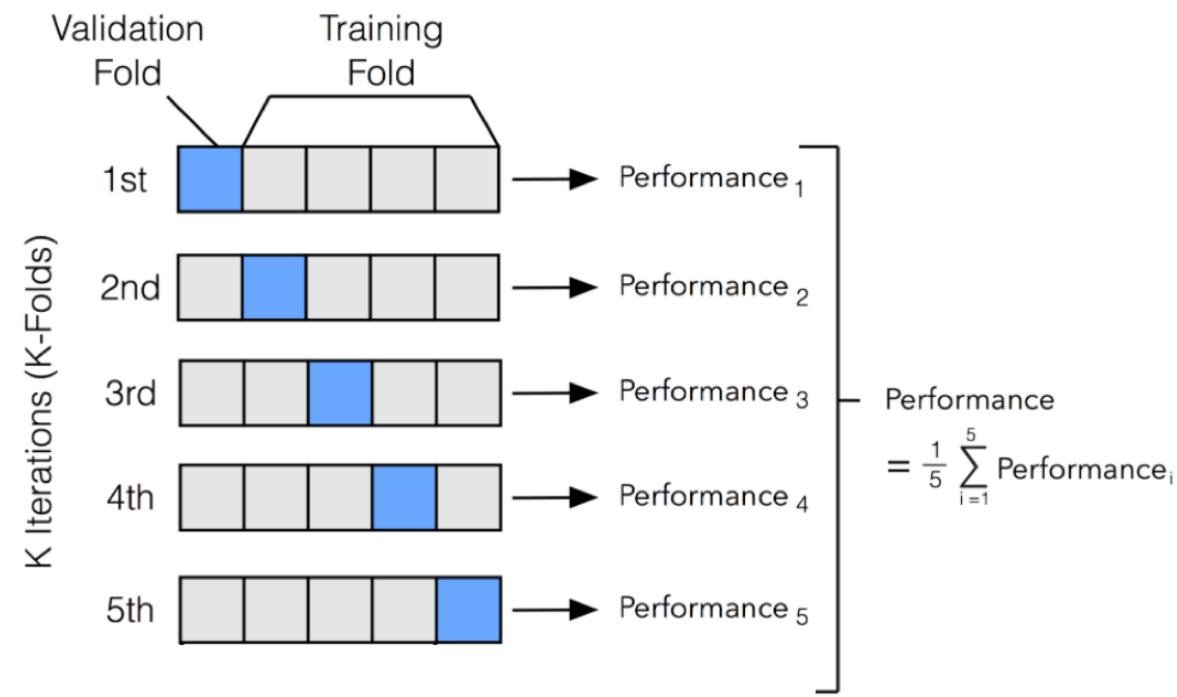
# Why we need cross validation?

- ▶ Cross-validation is used when the amount of data is not very sufficient (for example, the amount of data is less than 10,000), and can obtain as much effective information as possible from the limited data.
- ▶ Cross-validation is used to evaluate the prediction performance of the model, especially the performance of the trained model on new data, which can reduce overfitting to a certain extent.



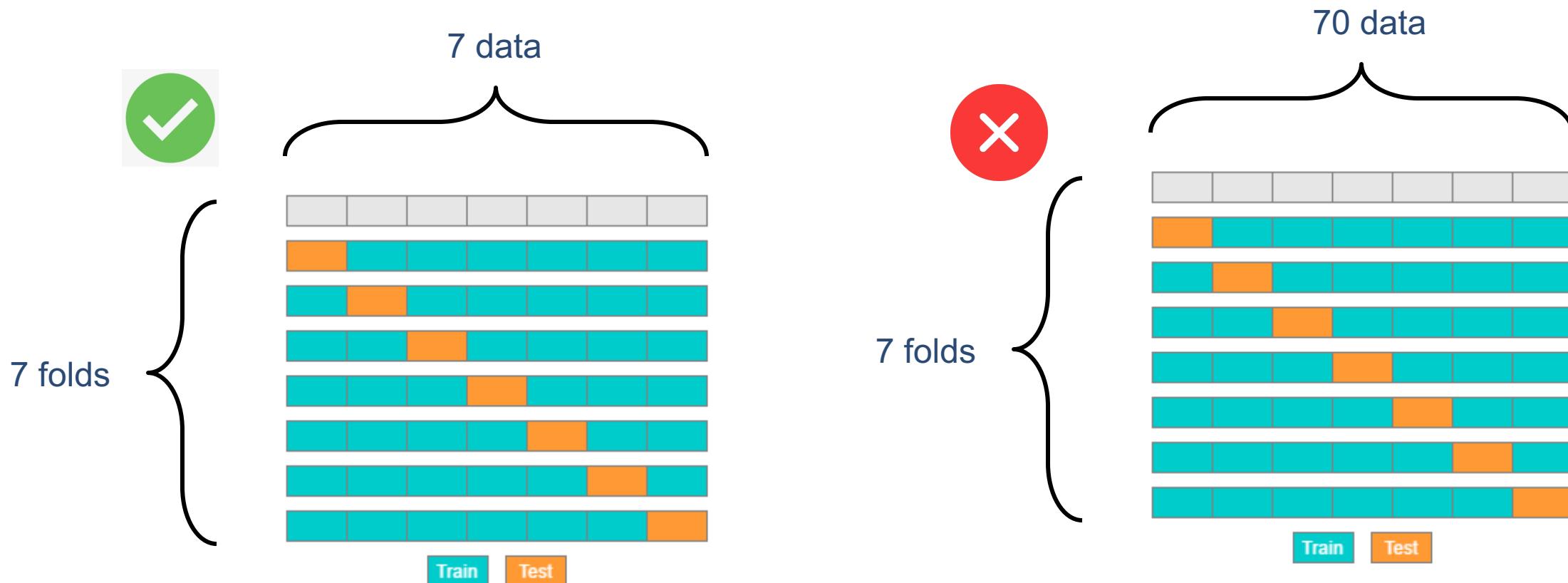
# k-fold Cross Validation

- ▶ The initial sample is divided into K sub-samples, one is used to verify the model, and k-1 are used for training
- ▶ Cross-validation is repeated K times, each sub-sample is validated once, and the results are averaged K times
- ▶ 10-fold cross-validation is the most commonly used



# Leave-one-out cross-validation

- ▶ A special case of cross-validation where the number of folds equals the number of instances in the data set

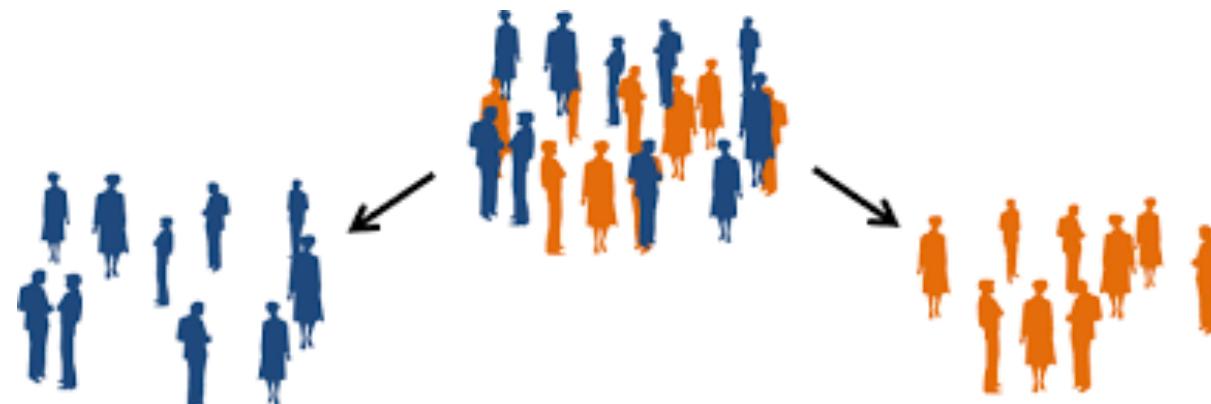


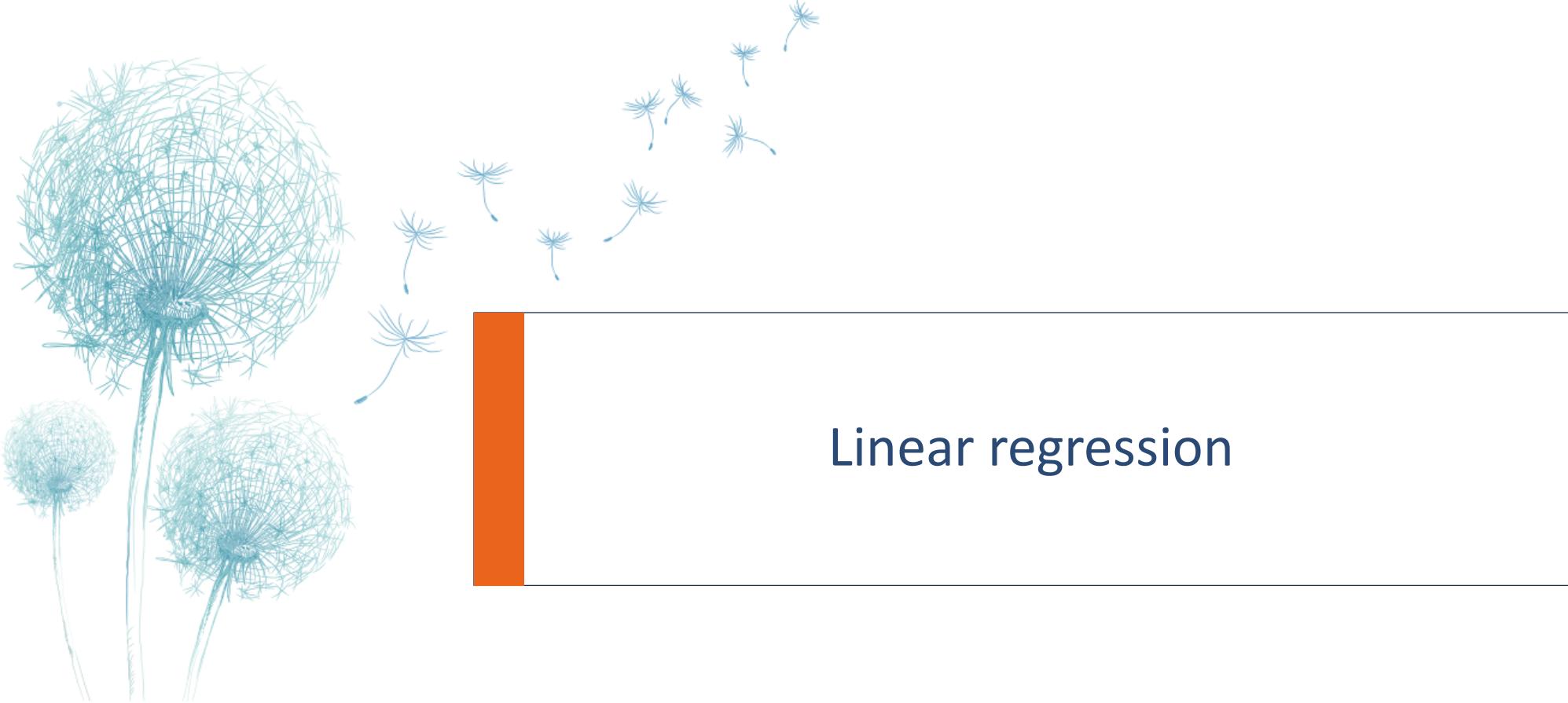
# Leave-one-out cross-validation

---

## ► Disadvantages:

- Long calculation time
- Stratification problem
  - In the leave-one-out cross-validation, all test sets contain only one data, which does not represent the full picture of the data set. Leave-one-out cross-validation is suitable for small data sets.





# Linear Regression

## ► Model:

$$f(\mathbf{x}; \mathbf{w}, b) = \mathbf{w}^T \mathbf{x} + b$$

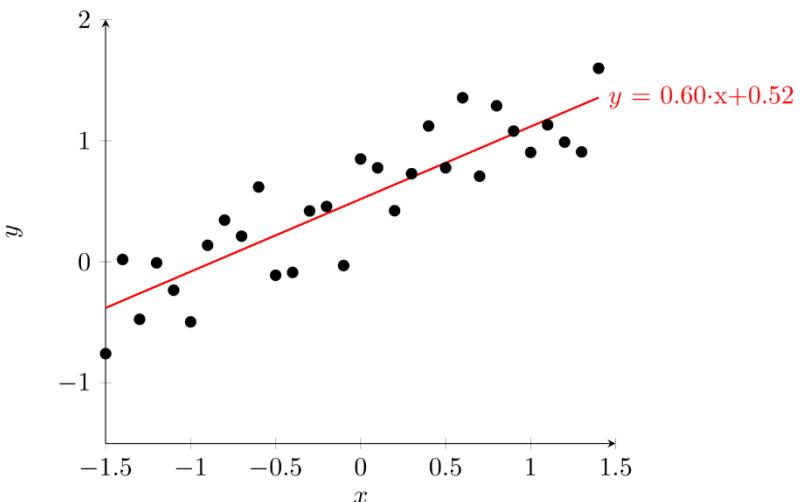
- Augmented weight vector and augmented feature vector

$$\hat{\mathbf{x}} = \mathbf{x} \oplus 1 \triangleq \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} = \begin{bmatrix} x_1 \\ \vdots \\ x_k \\ 1 \end{bmatrix},$$

$$\hat{\mathbf{w}} = \mathbf{w} \oplus b \triangleq \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix} = \begin{bmatrix} w_1 \\ \vdots \\ w_k \\ b \end{bmatrix},$$



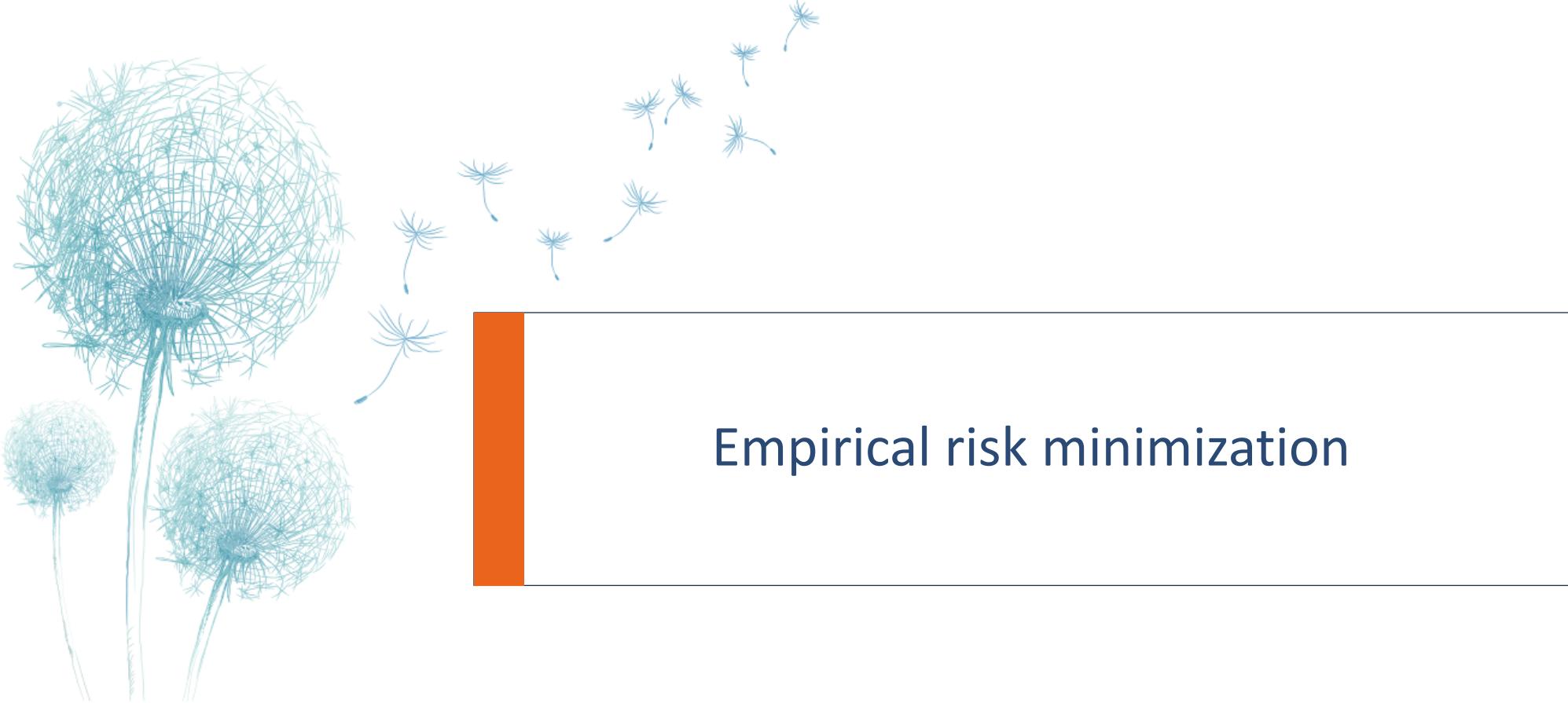
$$f(\mathbf{x}; \hat{\mathbf{w}}) = \hat{\mathbf{w}}^T \hat{\mathbf{x}},$$



# Optimization

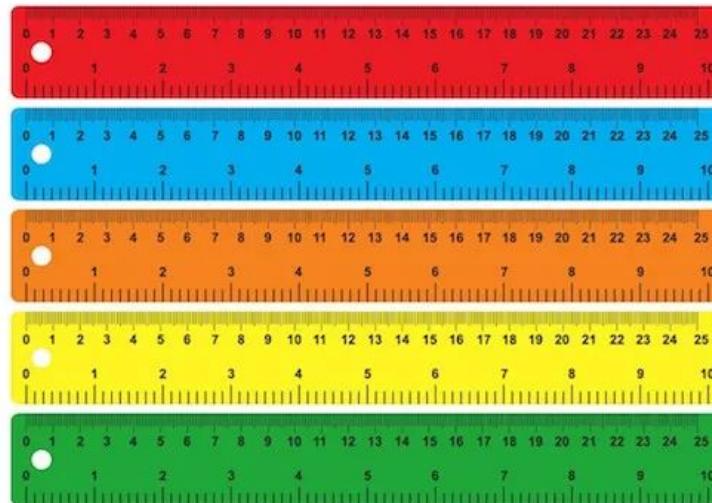
---

- ▶ Empirical risk minimization (least squares method)
- ▶ Structural risk minimization (ridge regression)
- ▶ Maximum likelihood estimation
- ▶ Maximum posterior estimate



# Empirical risk minimization (least squares method)

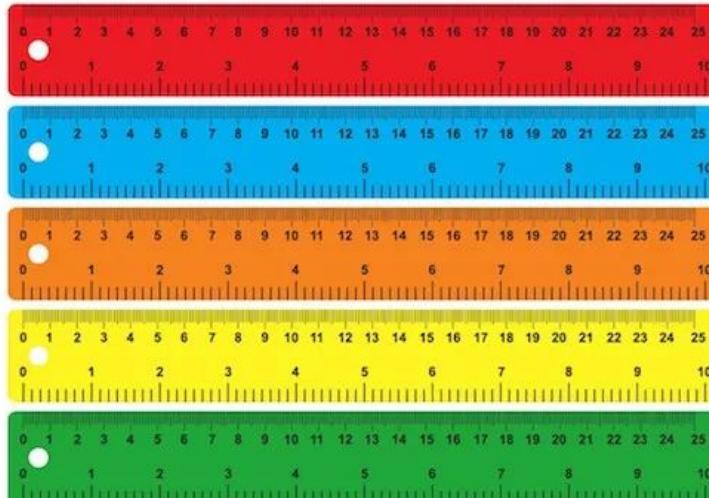
## ► Five rulers from five brands



	Length
Red	10.2
Blue	10.3
Orange	9.8
Yellow	9.9
Green	9.8

# Empirical risk minimization (least squares method)

## ► Five rulers from five brands



The different values may be due to:

Different manufacturers' rulers have different production accuracy

The material of the ruler is different, and the heat expansion and contraction are different

...

# Empirical risk minimization (least squares method)

---

## ► Five rulers from five brands

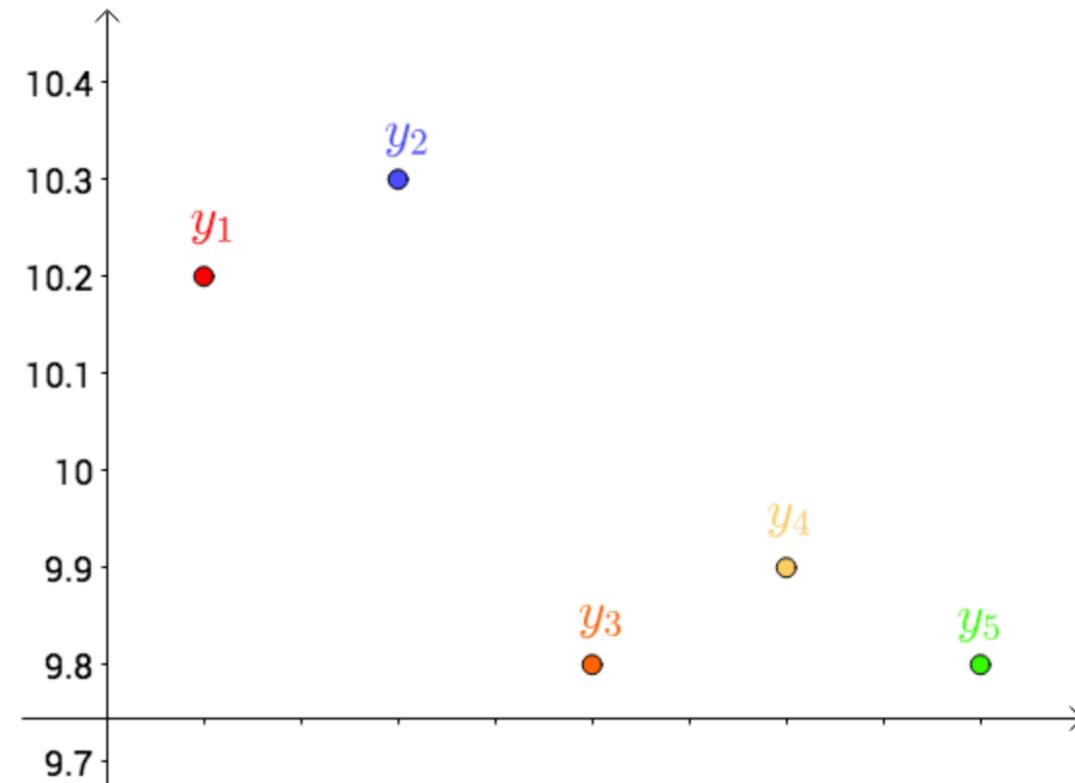
	Length
Red	10.2
Blue	10.3
Orange	9.8
Yellow	9.9
Green	9.8

$$\bar{x} = \frac{10.2 + 10.3 + 9.8 + 9.9 + 9.8}{5} = 10$$

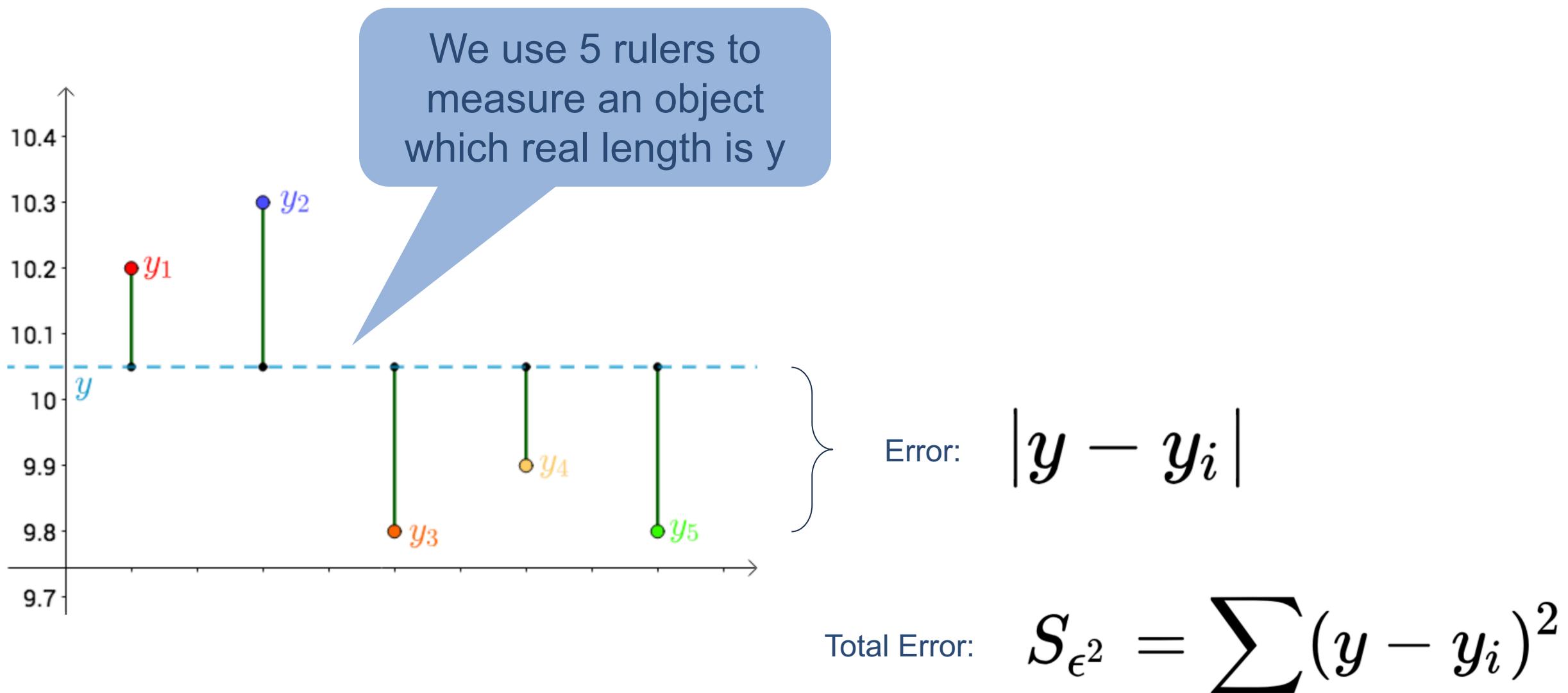
# Empirical risk minimization (least squares method)

## ► Five rulers from five brands

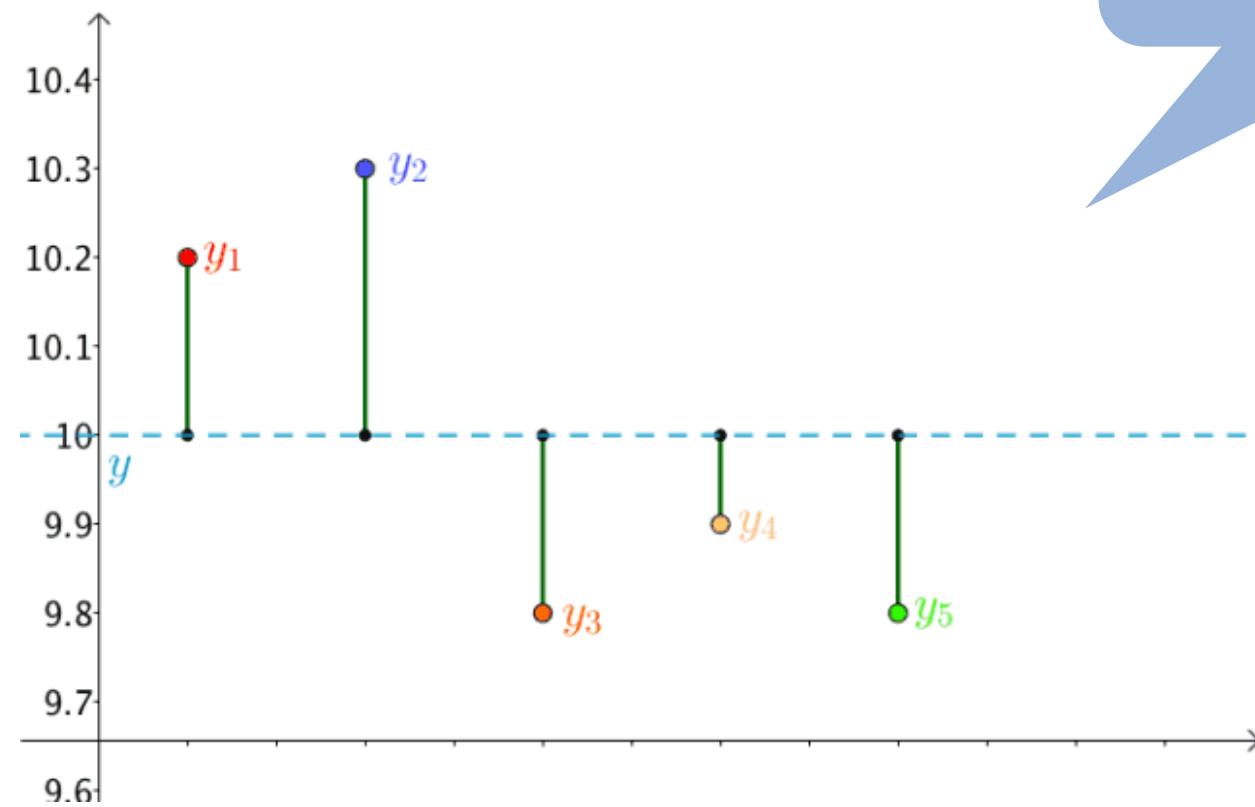
	Length
Red	10.2
Blue	10.3
Orange	9.8
Yellow	9.9
Green	9.8



# Empirical risk minimization (least squares method)



# Empirical risk minimization (least squares method)



Different objects have various errors for the five rulers.

Error:

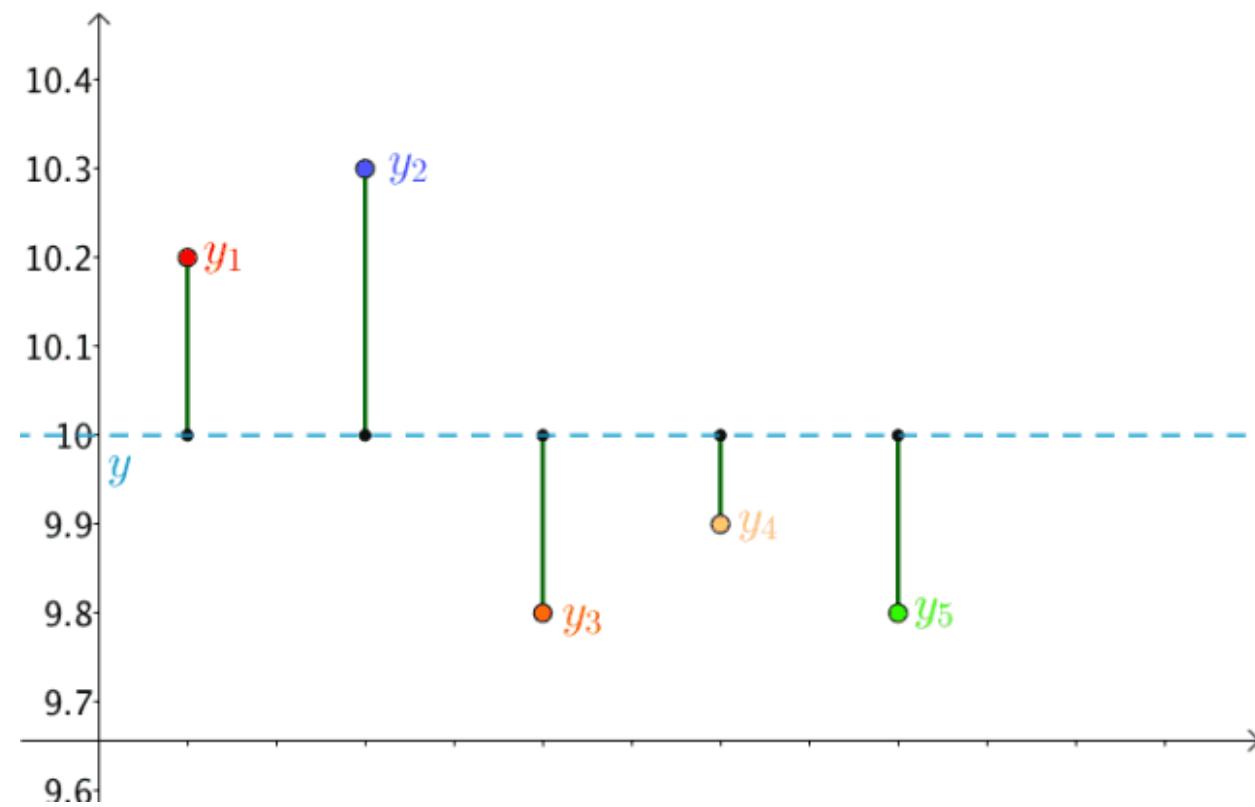
$$|y - y_i|$$

Total Error:

$$S_{\epsilon^2} = \sum (y - y_i)^2$$

$S$  is different for various  $y$

# Empirical risk minimization (least squares method)



Empirical risk minimization (least squares method) finds

$$y = \frac{y_1 + y_2 + y_3 + y_4 + y_5}{5}$$

can make minimize  $S_{\epsilon^2}$

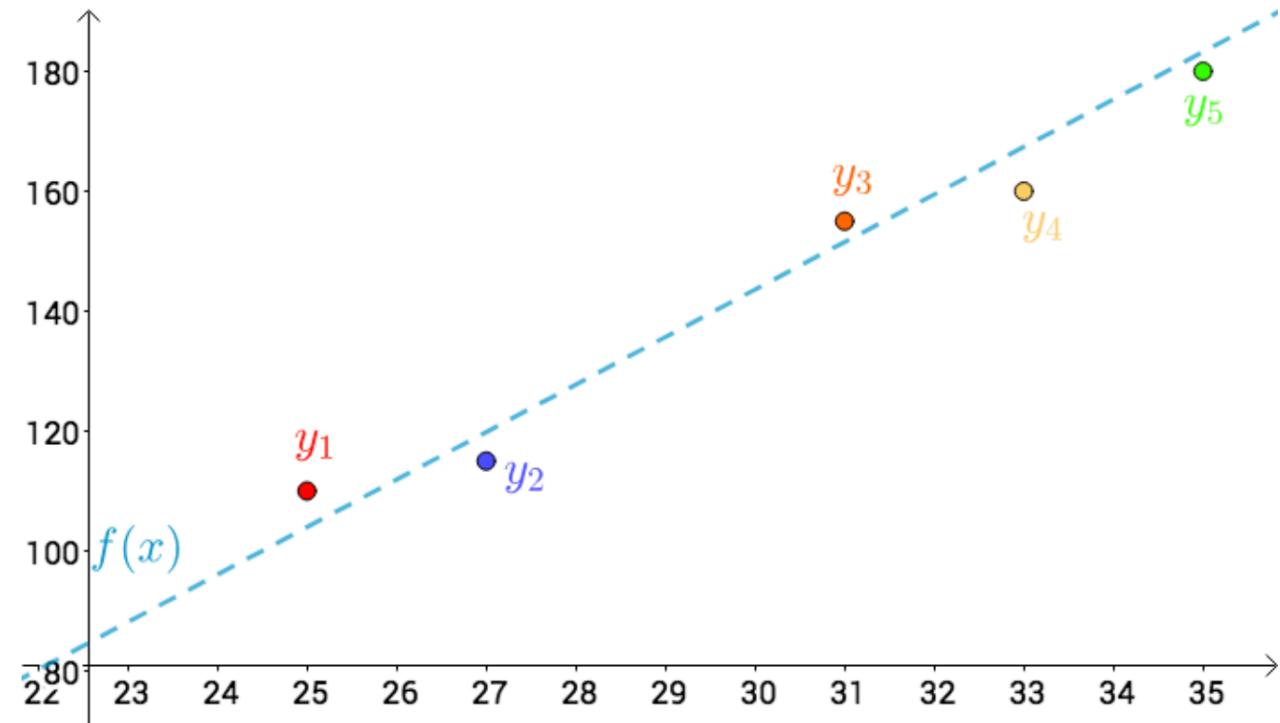
Total Error:

$$S_{\epsilon^2} = \sum (y - y_i)^2$$

# Empirical risk minimization (least squares method)

Temperature and ice cream sales:

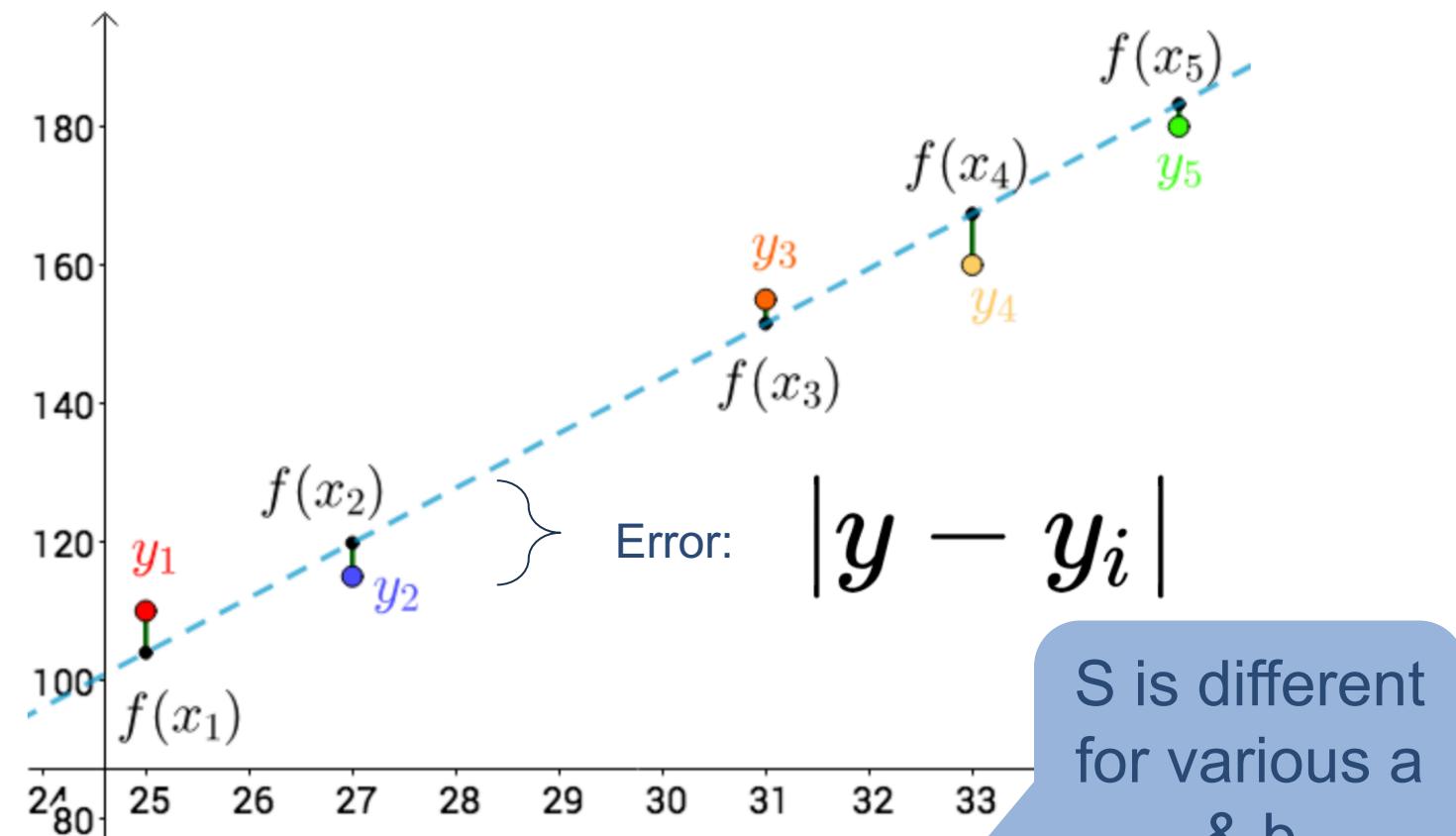
Temp (°C)	sales
25	110
27	115
31	155
33	160
35	180



# Empirical risk minimization (least squares method)

Temperature and ice cream sales:

Temp (°C)	sales
25	110
27	115
31	155
33	160
35	180



$$S_{\epsilon^2} = \sum (f(x_i) - y_i)^2 = \sum (ax_i + b - y_i)^2$$

# Empirical risk minimization (least squares)

S is different  
for various a  
& b

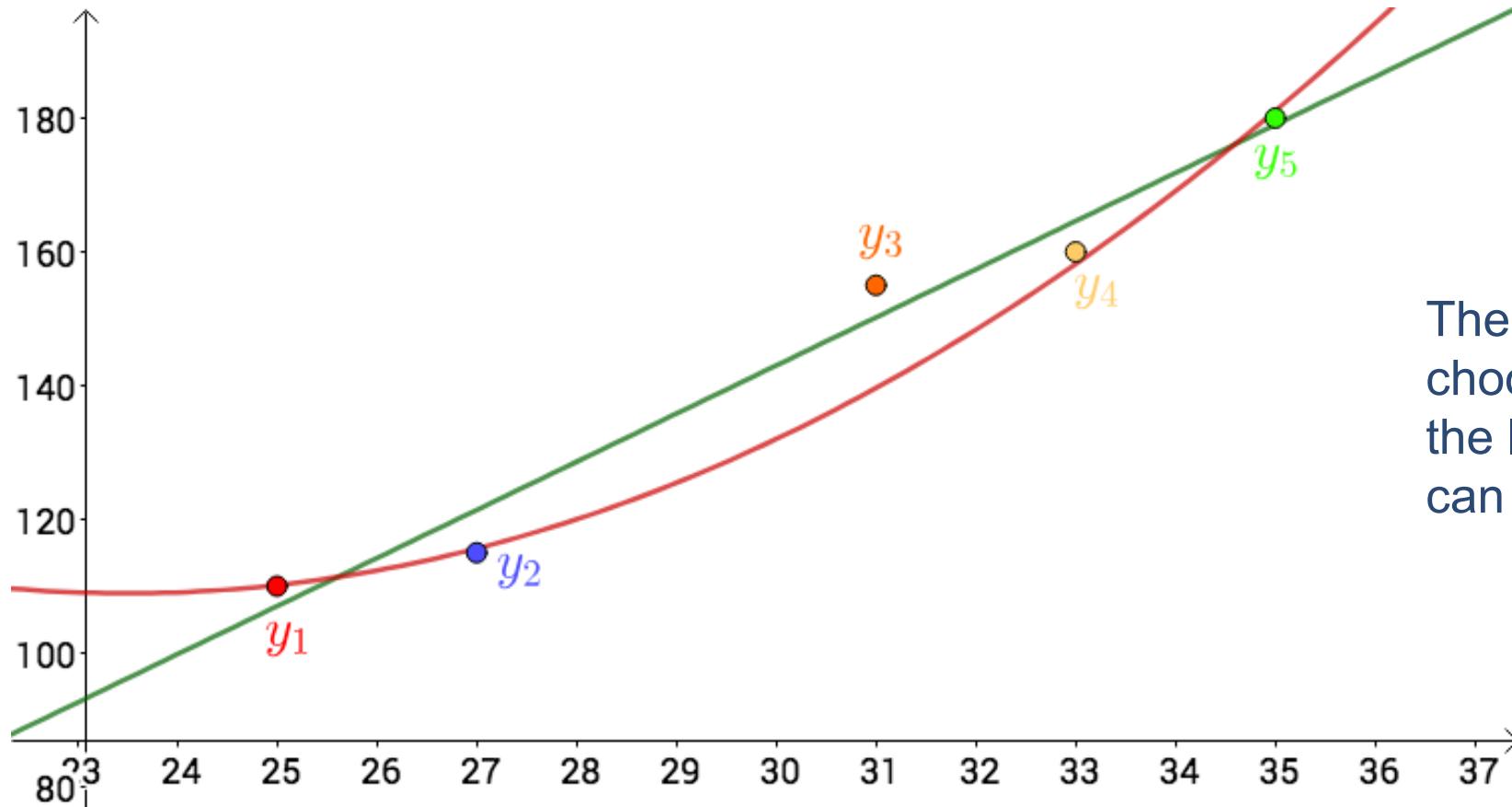
$$S_{\epsilon^2} = \sum(f(x_i) - y_i)^2 = \sum(ax_i + b - y_i)^2$$

When

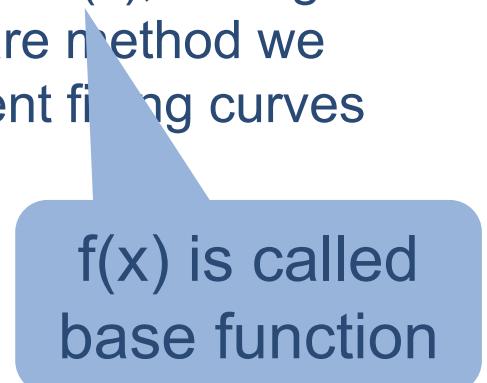
$$\begin{cases} \frac{\partial}{\partial a} S_{\epsilon^2} = 2 \sum(ax_i + b - y_i)x_i = 0 \\ \frac{\partial}{\partial b} S_{\epsilon^2} = 2 \sum(ax_i + b - y_i) = 0 \end{cases}$$

$S_{\epsilon^2}$  is minimum

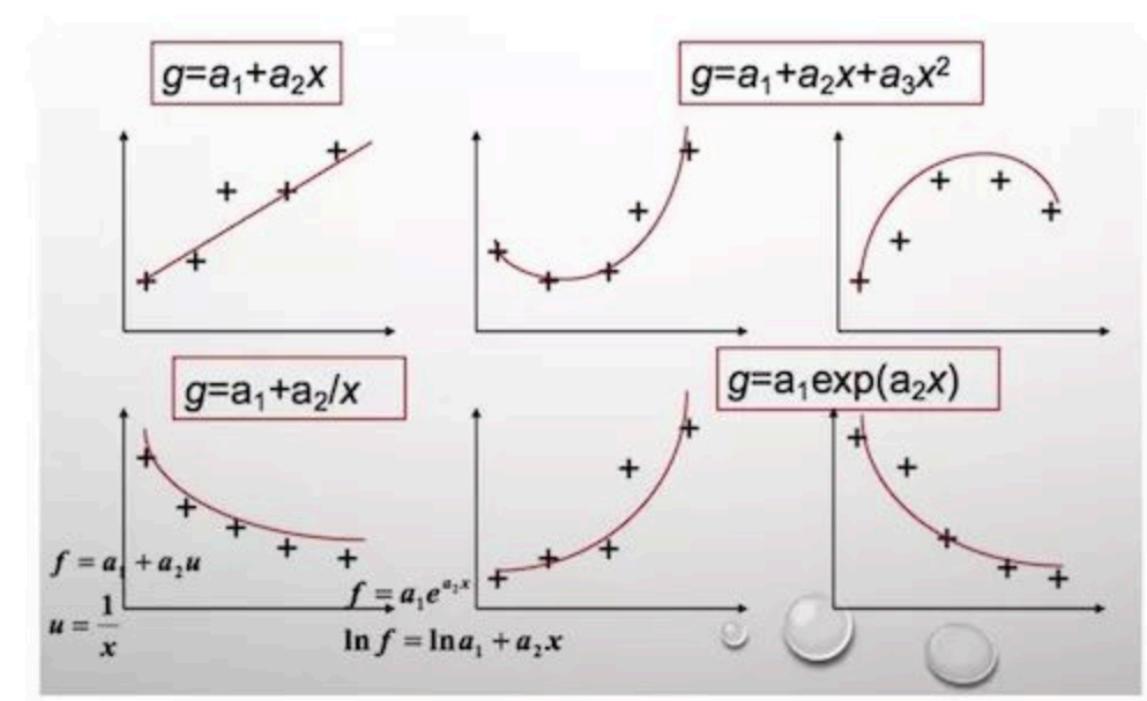
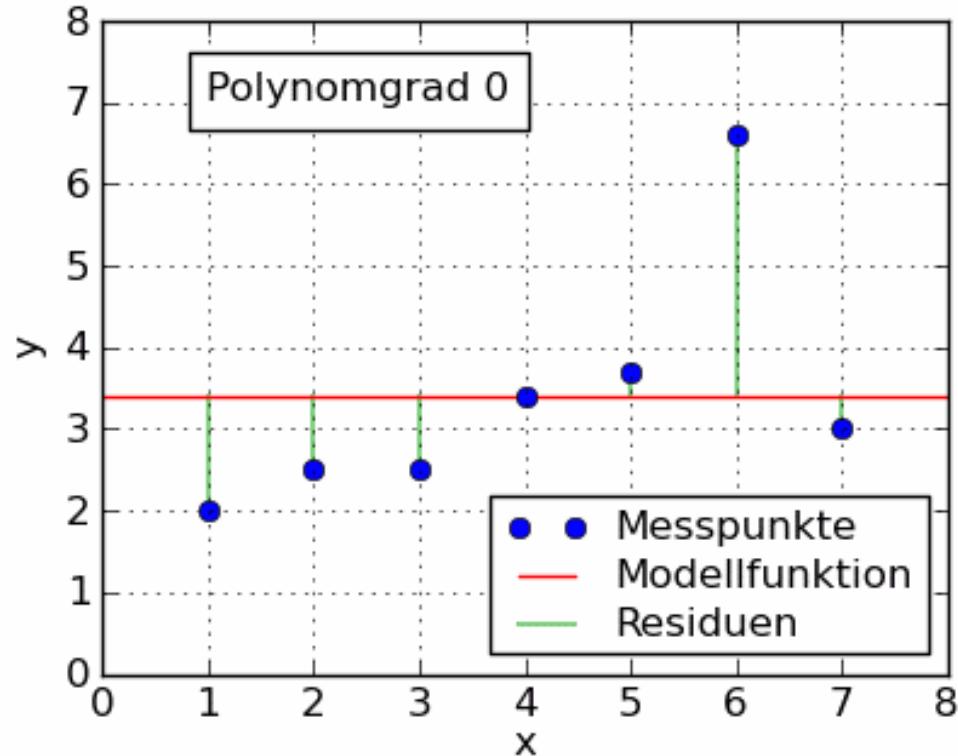
# Empirical risk minimization (least squares method)



The same set of data,  
choose different  $f(x)$ , through  
the least square method we  
can get different fitting curves

  $f(x)$  is called  
base function

# Empirical risk minimization (least squares method)



# Matrix calculus

---

## ► Partial derivative of a scalar with respect to a vector

$$\frac{\partial y}{\partial \mathbf{x}} = \left[ \frac{\partial y}{\partial x_1}, \dots, \frac{\partial y}{\partial x_M} \right]^\top$$

## ► Partial derivative of vector with respect to vector

$$\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \dots & \frac{\partial y_N}{\partial x_1} \\ \vdots & \vdots & \vdots \\ \frac{\partial y_1}{\partial x_M} & \dots & \frac{\partial y_N}{\partial x_M} \end{bmatrix}$$

## ► Vector functions and their derivatives

$$\frac{\partial \mathbf{x}}{\partial \mathbf{x}} = \mathbf{I},$$

$$\frac{\partial A\mathbf{x}}{\partial \mathbf{x}} = A^\top,$$

$$\frac{\partial \mathbf{x}^\top A}{\partial \mathbf{x}} = A$$

# Empirical risk minimization (least squares method)

---

- Model  $f(\mathbf{x}; \mathbf{w}) = \mathbf{w}^T \mathbf{x}$

- Study criteria

$$\begin{aligned}\mathcal{R}(\mathbf{w}) &= \sum_{n=1}^N \mathcal{L}(y^{(n)}, f(\mathbf{x}^{(n)}; \mathbf{w})) \\ &= \frac{1}{2} \sum_{n=1}^N \left( y^{(n)} - \mathbf{w}^T \mathbf{x}^{(n)} \right)^2 \\ &= \frac{1}{2} \|\mathbf{y} - \mathbf{X}^T \mathbf{w}\|^2,\end{aligned}$$

- Model

$$\frac{\partial}{\partial \mathbf{w}} \mathcal{R}(\mathbf{w}) = 0$$



Maximum likelihood estimation

# Probability vs. Likelihood

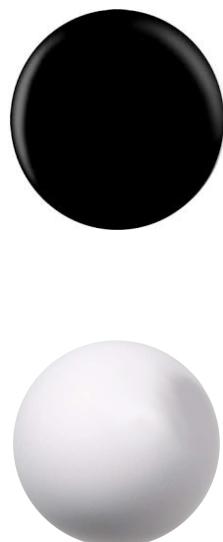
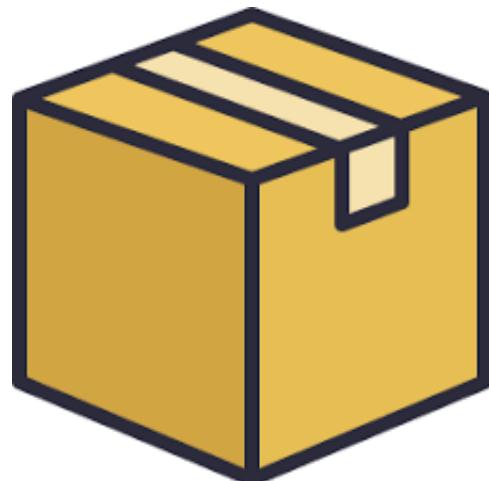
---

- ▶ **Likelihood** is the **probability** that an event that has already occurred would yield a specific outcome.
- ▶ **Probability** refers to the occurrence of future events, while a **likelihood** refers to past events with known outcomes.

# Maximum likelihood estimation

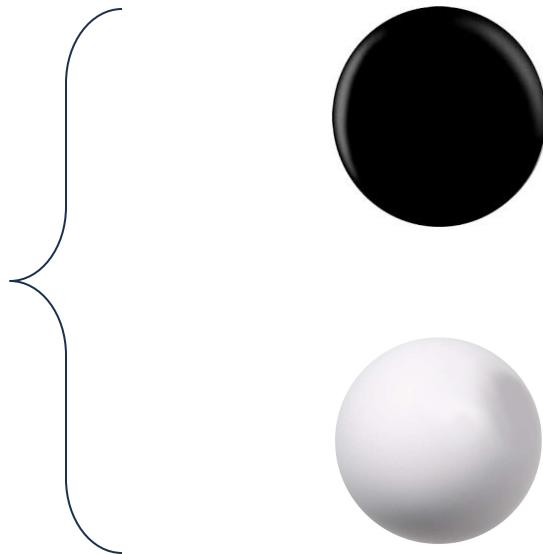
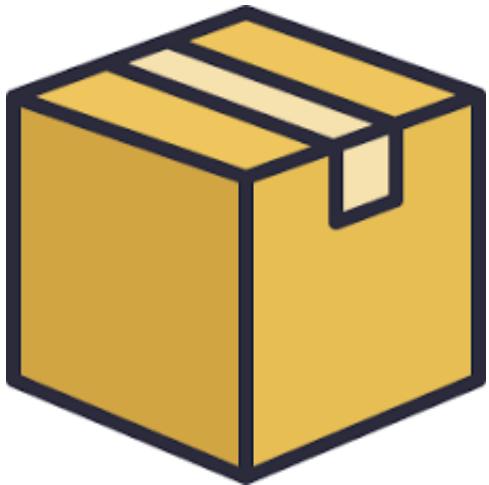
## ► Maximum likelihood estimation:

- Given a set of data and a model with undetermined parameters, how to determine the parameters of the model so that the model with determined parameters has the greatest probability of producing known data in all models



We do not know  
how many black ball  
and white ball.

# Maximum likelihood estimation



We do not know  
how many black ball  
and white ball.

If in the previous 100 repetition records, 70 times were white balls, what is the most likely proportion of white balls in the box?

70%

# Maximum likelihood estimation

---

- ▶ We assume that the proportion of white balls in the box is  $p$ , then the proportion of black balls is  $1-p$
- ▶ In 100 samplings, the probability of being a white ball 70 times is  $P(\text{Data} | M)$ , where Data is all the data, and  $M$  is the given model

$$P(\text{Data} | M)$$

$$= P(x_1, x_2, \dots, x_{100} | M)$$

$$= P(x_1 | M)P(x_2 | M)\dots P(x_{100} | M)$$

$$= p^{70} (1 - p)^{30}$$

What value of  $p$  takes, the value of  $P(\text{Data} | M)$  is the largest

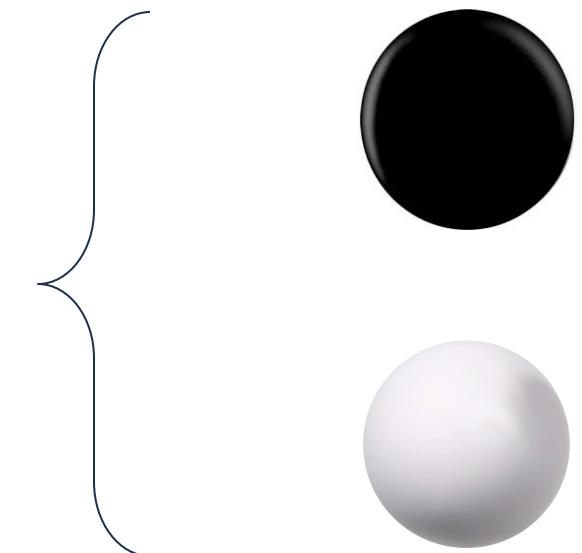
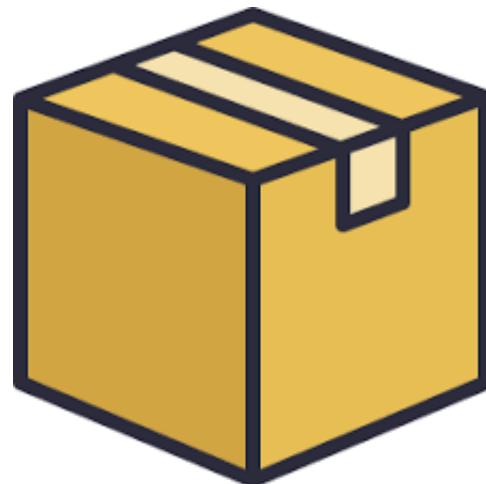
# Maximum likelihood estimation

$$f(p) = p^{70} (1 - p)^{30}$$

What value of  $p$  takes,  
the value of  $P(\text{Data} | M)$  is the largest

$$f(p)' = 70 * p^{69} (1 - p)^{30} - p^{70} * 30 * (1 - p)^{29} = 0$$

→  $P = 0.7$



# Some basic concepts about probability

---

## ► Probability

- The probability of a random event occurring, which is a real number between 0 and 1.

## ► Random Variable

- For example, if you randomly roll a dice, the number of points you get can be regarded as a random variable  $X$  whose value is  $\{1,2,3,4,5,6\}$ .

## ► Probability Distribution

$$P(X = x_i) = p(x_i), \quad \forall i \in \{1, \dots, n\}.$$

$$\sum_{i=1}^n p(x_i) = 1, \quad p(x_i) \geq 0, \quad \forall i \in \{1, \dots, n\}.$$

# Some basic concepts about probability

---

## ► Bernoulli Distribution

- In an experiment, the probability of occurrence of event A is  $\mu$ , and the probability of not occurring is  $1 - \mu$ . If the variable X is used to represent the number of occurrences of event A, the values of X are 0 and 1, and the corresponding distribution is

$$p(x) = \mu^x (1 - \mu)^{(1-x)}$$

## ► Binomial Distribution

- In the n-th Bernoulli distribution, if the variable X represents the number of occurrences of event A, the value of X is  $\{0, \dots, n\}$ , and the corresponding distribution

$$P(X = k) = \binom{n}{k} \mu^k (1 - \mu)^{n-k}, \quad k = 1 \dots, n$$

# Some basic concepts about probability

---

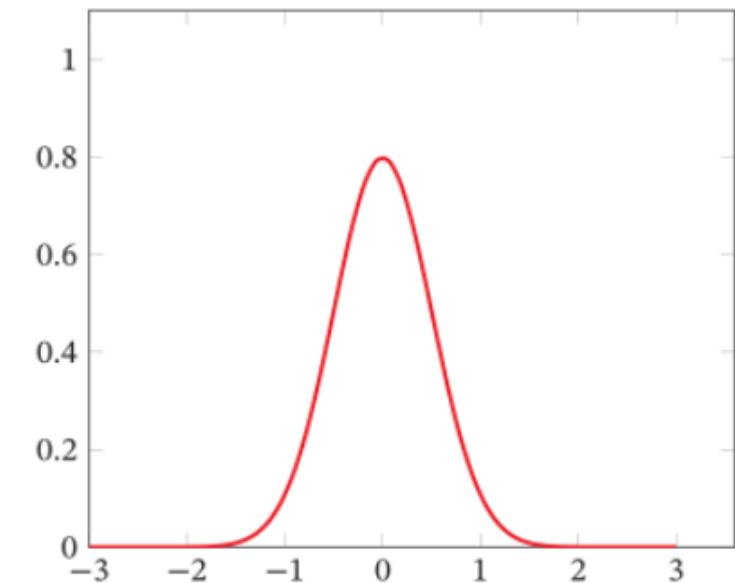
## ► Gaussian Distribution (normal distribution)

- a type of continuous probability distribution for a real-valued random variable

$$X \sim \mathcal{N}(\mu, \sigma^2)$$

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Expectation is  $\mu$   
Variance is  $\sigma^2$

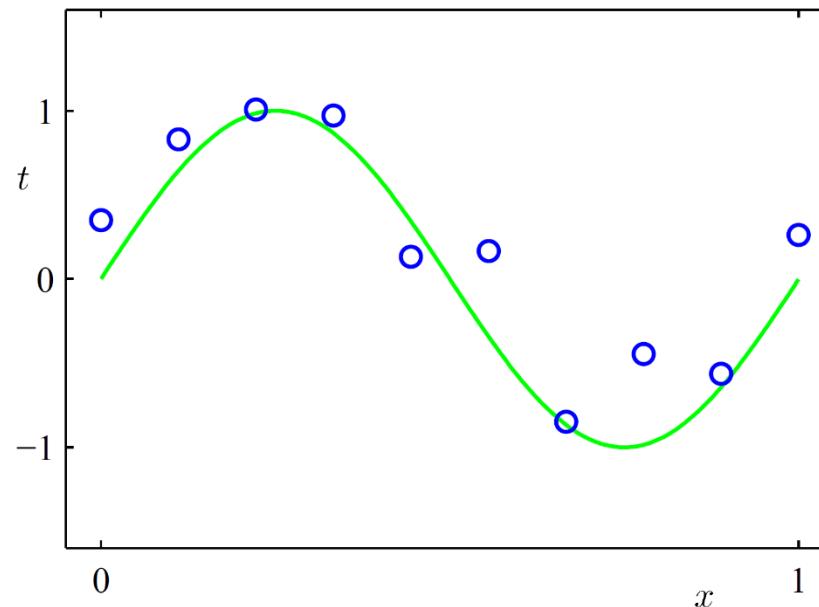




## Polynomial regression

# Polynomial Curve Fitting

---



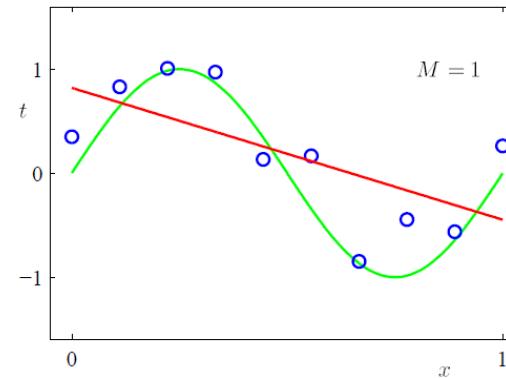
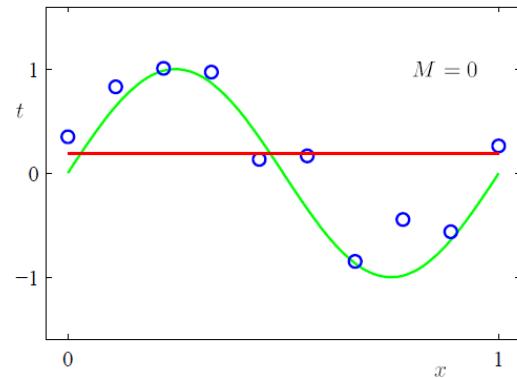
Model

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \dots + w_M x^M$$

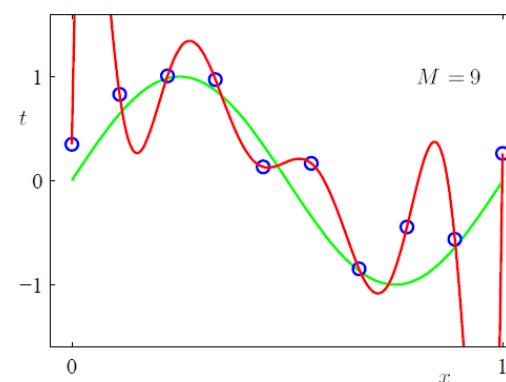
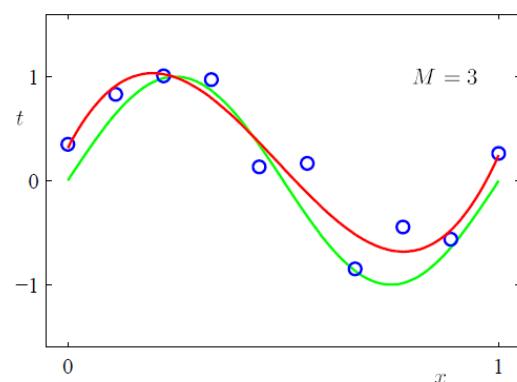
Loss function

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N \{y(x_n, \mathbf{w}) - t_n\}^2$$

# Which Degree of Polynomial?

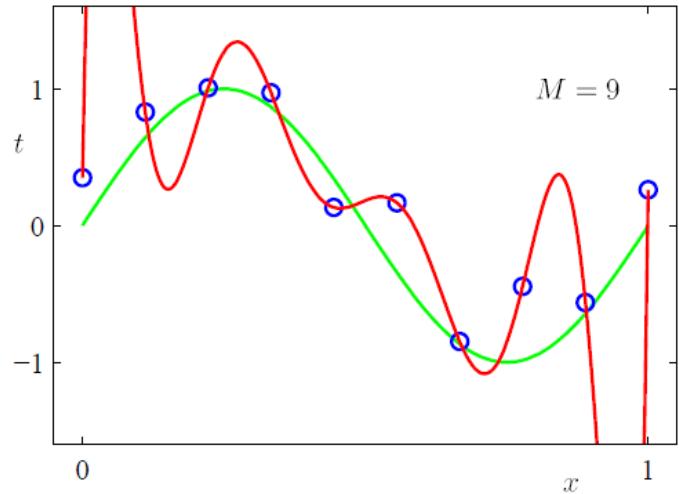


A model selection problem



$M = 9 \rightarrow E(w) = 0$ : This is overfitting

# Controlling Overfitting: Regularization



	$M = 0$	$M = 1$	$M = 3$	$M = 9$
$w_0^*$	0.19	0.82	0.31	0.35
$w_1^*$		-1.27	7.99	232.37
$w_2^*$			-25.43	-5321.83
$w_3^*$			17.37	48568.31
$w_4^*$				-231639.30
$w_5^*$				640042.26
$w_6^*$				-1061800.52
$w_7^*$				1042400.18
$w_8^*$				-557682.99
$w_9^*$				125201.43

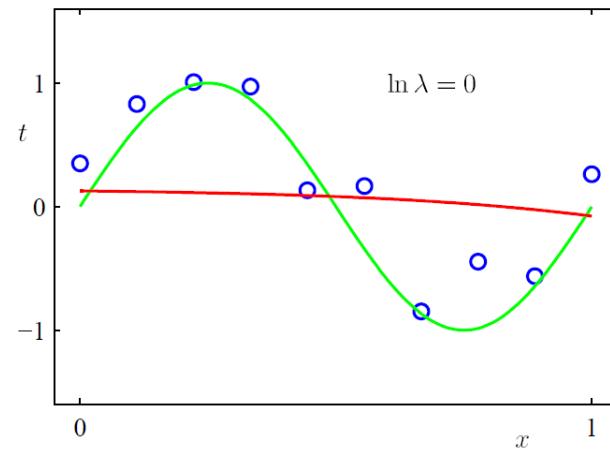
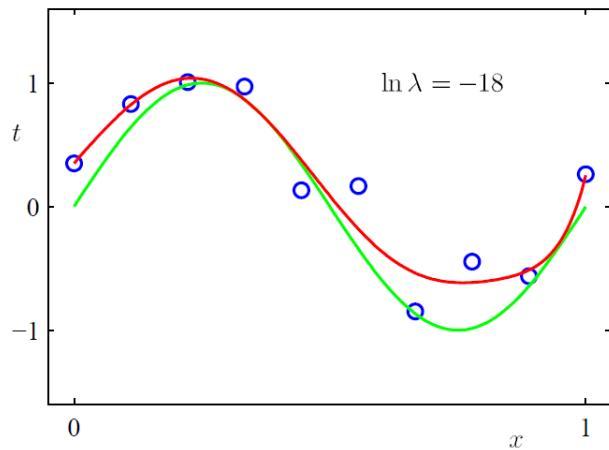
As order of polynomial M increases, so do coefficient magnitudes!

$$\tilde{E}(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N \{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2} \|\mathbf{w}\|^2$$

Penalize large coefficients

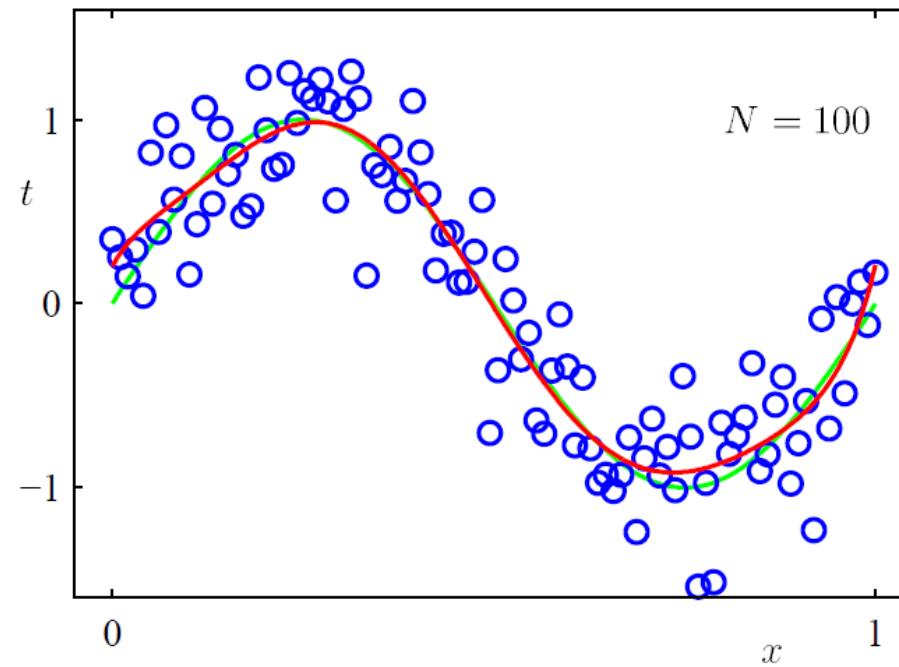
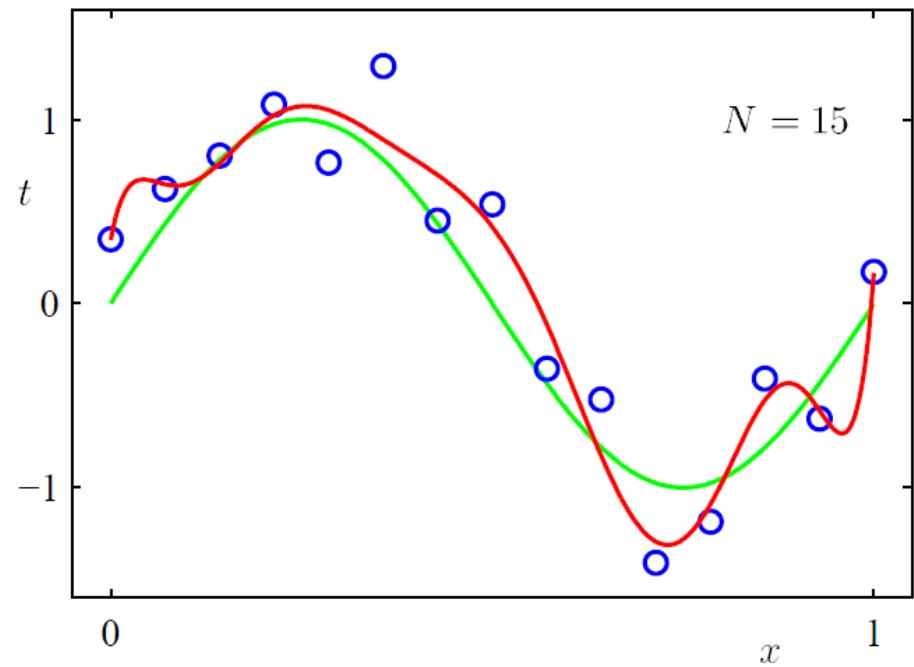
# Controlling Overfitting: Regularization

$$\tilde{E}(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N \{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2} \|\mathbf{w}\|^2$$



	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
$w_0^*$	0.35	0.35	0.13
$w_1^*$	232.37	4.74	-0.05
$w_2^*$	-5321.83	-0.77	-0.06
$w_3^*$	48568.31	-31.97	-0.05
$w_4^*$	-231639.30	-3.89	-0.03
$w_5^*$	640042.26	55.28	-0.02
$w_6^*$	-1061800.52	41.32	-0.01
$w_7^*$	1042400.18	-45.95	-0.00
$w_8^*$	-557682.99	-91.53	0.00
$w_9^*$	125201.43	72.68	0.01

# Controlling Overfitting: Dataset size





Several key points of machine learning

# How to choose a suitable model?

---

## ► Model selection

- A model with a strong fitting ability is generally more complex and easier to overfit.
- If the model complexity is limited and the fitting ability is reduced, it may be underfitting.

## ► Bias and variance decomposition

- The expected error can be broken down into

$$\mathcal{R}(f) = (\text{bias})^2 + \text{variance} + \varepsilon.$$

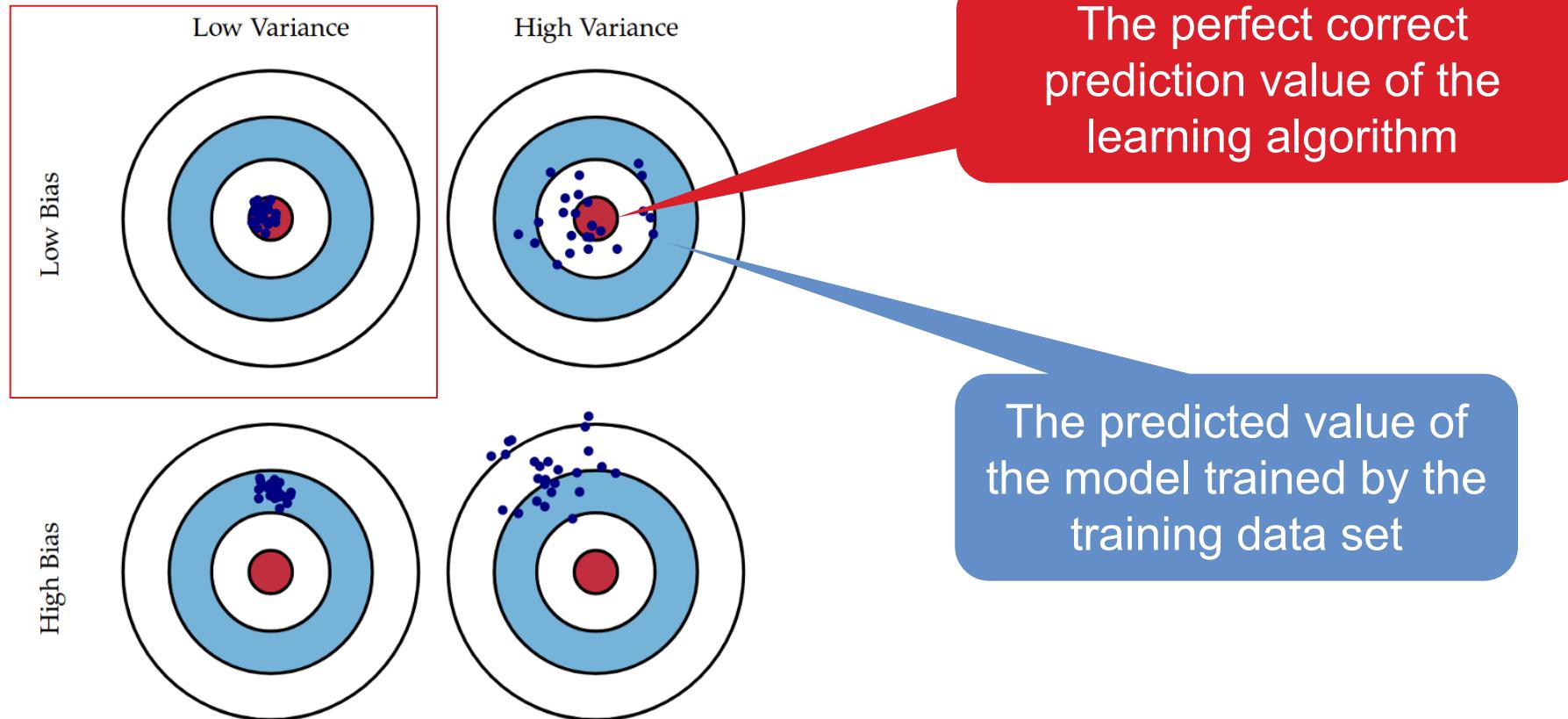
$$\mathbb{E}_{\mathbf{x}} \left[ \left( \mathbb{E}_{\mathcal{D}} [f_{\mathcal{D}}(\mathbf{x})] - f^*(\mathbf{x}) \right)^2 \right]$$
$$\mathbb{E}_{\mathbf{x}} \left[ \mathbb{E}_{\mathcal{D}} \left[ (f_{\mathcal{D}}(\mathbf{x}) - \mathbb{E}_{\mathcal{D}} [f_{\mathcal{D}}(\mathbf{x})])^2 \right] \right]$$
$$\mathbb{E}_{(\mathbf{x}, y) \sim p_r(\mathbf{x}, y)} \left[ (y - f^*(\mathbf{x}))^2 \right]$$

# Model selection: bias and variance

---

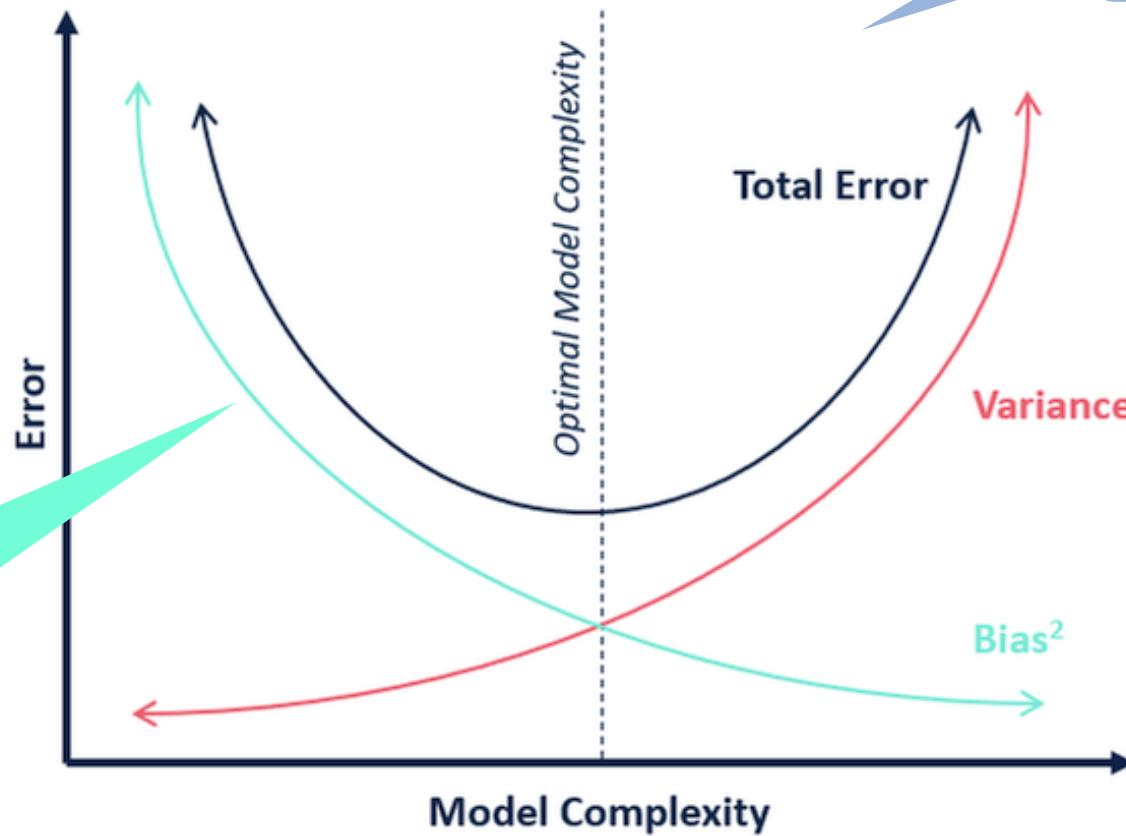
- ▶ Bias and Variance describe the gap between the model we learned and the real model from two aspects.
  - Bias is the difference between the average output of all models trained with all possible training data sets and the output value of the real model.
  - Variance is the difference between the output values of models trained on different training data sets.

# Model selection: bias and variance



# Model selection: bias and variance

Bias and variance are in conflict, which is called the bias-variance dilemma



A model with too low complexity cannot fit the training data well, but it is easy to cause large bias

A model with high complexity usually has a good fitting ability to the training data, but it is easy to cause large variance

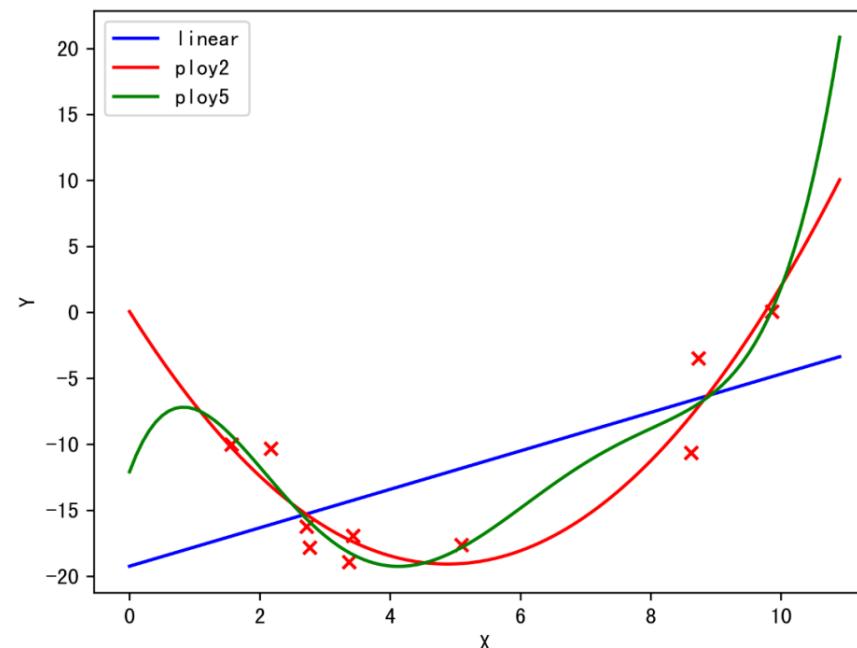
# Bias and variance

---

► Biases and variances cannot be completely avoided, only their impact can be minimized

1. Choose the correct model as much as possible

- For a nonlinear problem, if we have been using a linear model to solve it. In any case, high deviations cannot be avoided



# Bias and variance

---

► Biases and variances cannot be completely avoided, only their impact can be minimized

## 2. Choose the size of the data set carefully

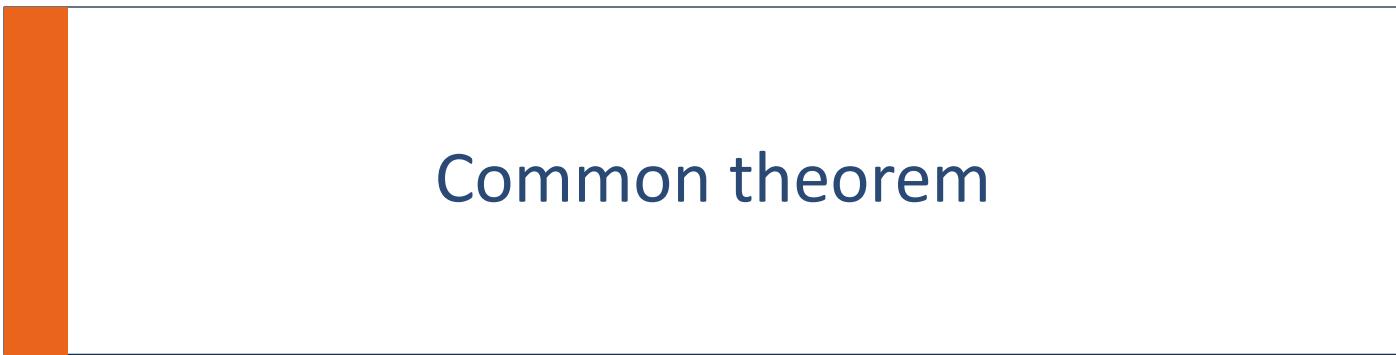
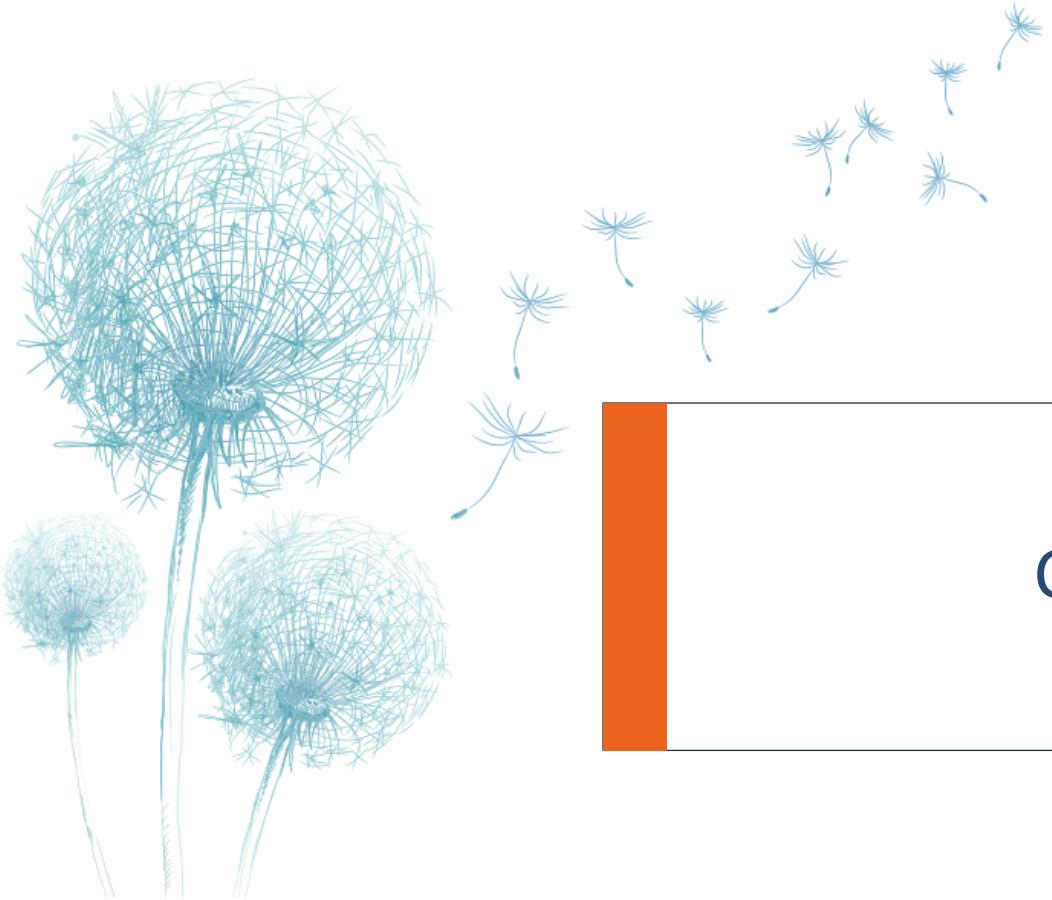
- Generally, the larger the data set, the better, but when the data set is so large that it has a certain representativeness for all the data, no amount of data can improve the model, but it will increase the amount of calculation.
- Too small data is always bad
- The model complexity is too high, the variance is large, and the models trained on different data sets vary greatly

# Bias and variance

► Biases and variances cannot be completely avoided, only their impact can be minimized

3. Choose the appropriate model complexity, the model with high complexity usually has a good fitting ability to the training data





Common theorem

# Common theorem

---

## ► No Free Lunch Theorem

- For an iterative-based optimization algorithm, there is no certain algorithm that is effective for all problems (within a limited search space). If an algorithm is effective for some problems, then it must be worse than pure random search algorithms on other problems.

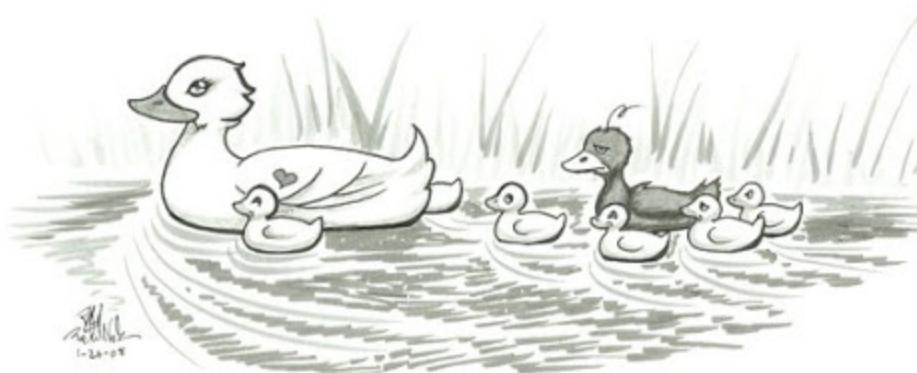


# Common theorem

---

## ► Ugly Duckling Theorem

- The difference between an ugly duckling and a white swan is as big as the difference between two white swans.



# Common theorem

---

## ► Occam's Razor

- If not necessary, do not add entities



# Inductive Bias

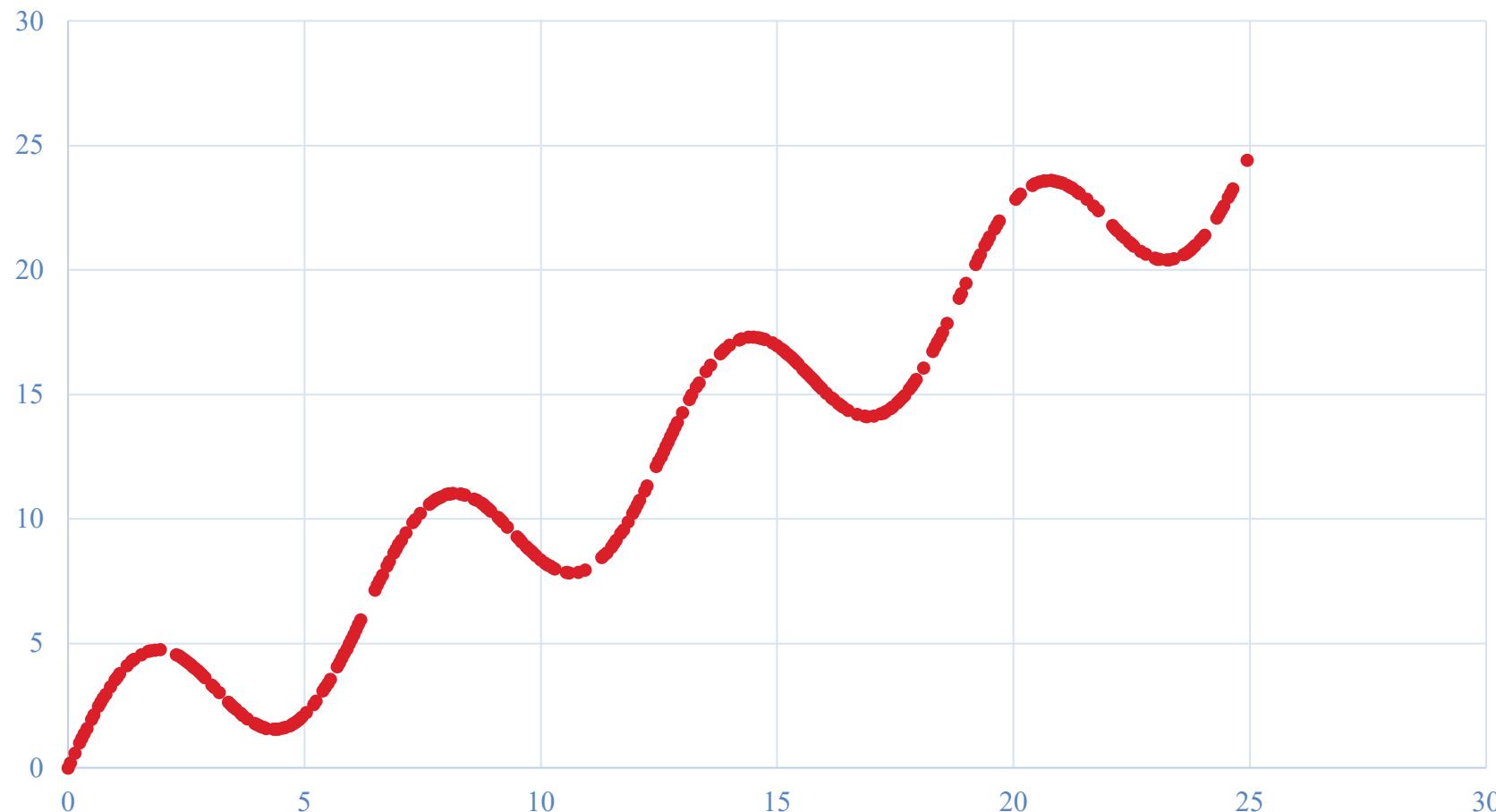
---

- ▶ Many learning algorithms often make some assumptions about the learning problem. These assumptions are called *inductive biases*.
  - In the nearest neighbor classifier, we will assume that in the feature space, most samples in a small local area belong to the same class.
  - In the Naive Bayes classifier, we will assume that the conditional probabilities of each feature are independent of each other.
- ▶ Inductive bias is often called a *prior* in Bayesian learning.

# Train dataset

$\{(x_1, y_1), \dots, (x_N, y_N)\}, N = 300$

Y-Values



# Data

```
"""-----data-----"""
def load_data(filename):
    xys = []
    with open(filename, 'r') as f:
        for line in f:
            xys.append(map(float, line.strip().split()))
    xs, ys = zip(*xys)
    return np.asarray(xs), np.asarray(ys)
"""-----data-----"""

if __name__ == '__main__':
    train_file = 'train.txt'
    test_file = 'test.txt'
    x_train, y_train = load_data(train_file)
    x_test, y_test = load_data(test_file)
    print(x_train.shape)
    print(x_test.shape)
```

Read a file,  
return an array

	train.txt		
1	2.65000	4.06609	
2	4.70000	1.70023	
3	14.25000	17.23092	
4	22.10000	21.77409	
5	10.55000	7.84291	
6	6.50000	7.14536	
7	4.10000	1.64517	
8	24.65000	23.25737	
9	24.40000	22.39327	
10	22.80000	20.62952	
11	0.90000	3.24998	
12	11.30000	8.43794	
13	21.80000	22.36996	
14	22.55000	20.95936	
15	19.20000	20.22994	
16	21.40000	23.07195	
17	11.70000	9.41405	
18	5.75000	4.22516	

# Basis function

```
def identity_basis(x):
    ret = np.expand_dims(x, axis=1)
    return ret

def multinomial_basis(x, feature_num=10):
    x = np.expand_dims(x, axis=1) # shape(N, 1)
    feat = [x]
    for i in range(2, feature_num+1):
        feat.append(x**i)
    ret = np.concatenate(feat, axis=1)
    return ret

def gaussian_basis(x, feature_num=10):
    centers = np.linspace(0, 25, feature_num)
    width = 1.0 * (centers[1] - centers[0])
    x = np.expand_dims(x, axis=1)
    x = np.concatenate([x]*feature_num, axis=1)

    out = (x-centers)/width
    ret = np.exp(-0.5 * out ** 2)
    return ret

"""-----Train model-----"""
def main(x_train, y_train):
    basis_func = gaussian_basis
    phi0 = np.expand_dims(np.ones_like(x_train), axis=1)
    phi1 = basis_func(x_train)
    phi = np.concatenate([phi0, phi1], axis=1)
    w = np.dot(np.linalg.pinv(phi), y_train)

    def f(x):
        phi0 = np.expand_dims(np.ones_like(x), axis=1)
        phi1 = basis_func(x)
        phi = np.concatenate([phi0, phi1], axis=1)
        y = np.dot(phi, w)
        return y
    pass

    return f

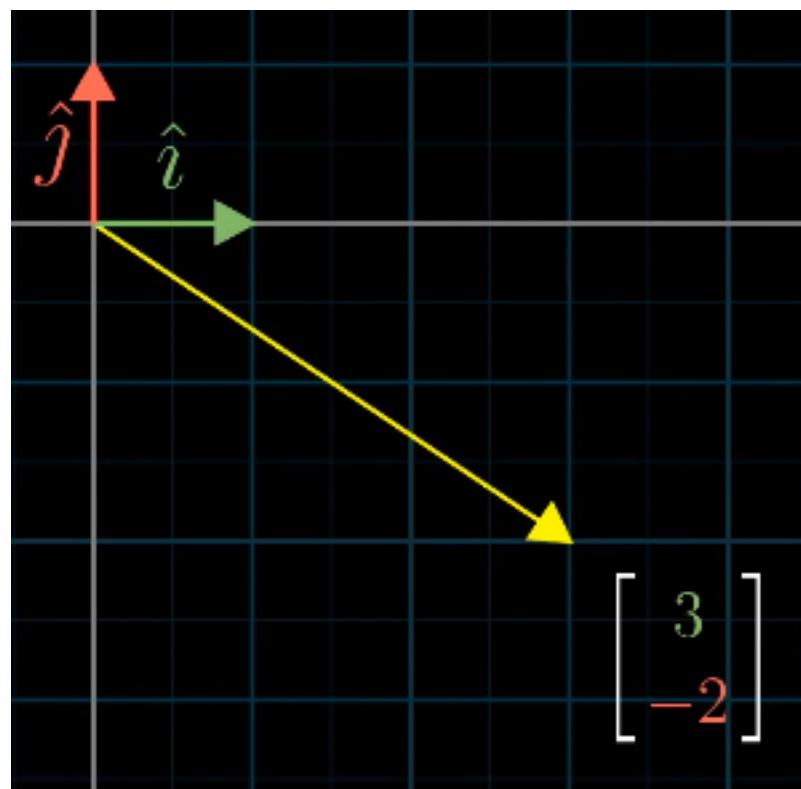
"""-----Train model-----"""


```

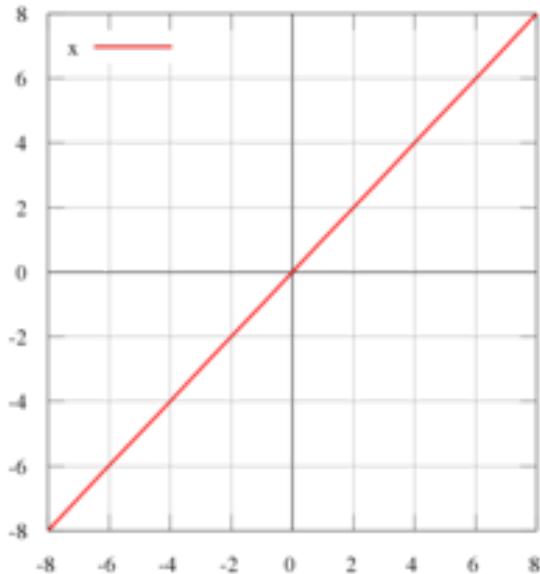
# Basis function

---

In mathematics, a basis function is an element of a particular basis for a function space. Every continuous function in the function space can be represented as a linear combination of basis functions



# Basis function



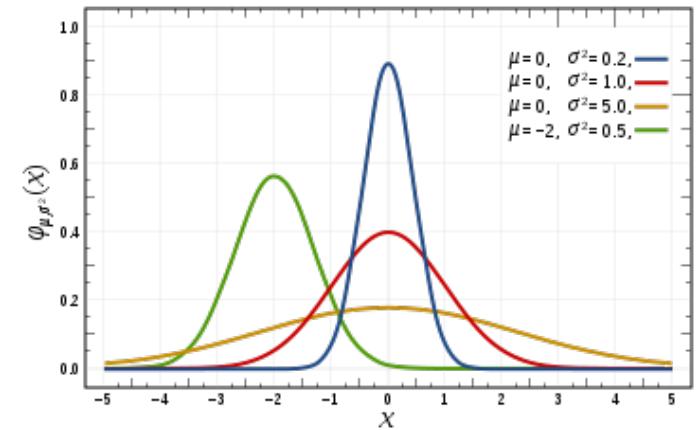
$f(x) = x$  for all elements  $x$  in  $M$

Identity function



$a_n x^n + a_{n-1} x^{n-1} + \cdots + a_2 x^2 + a_1 x + a_0$

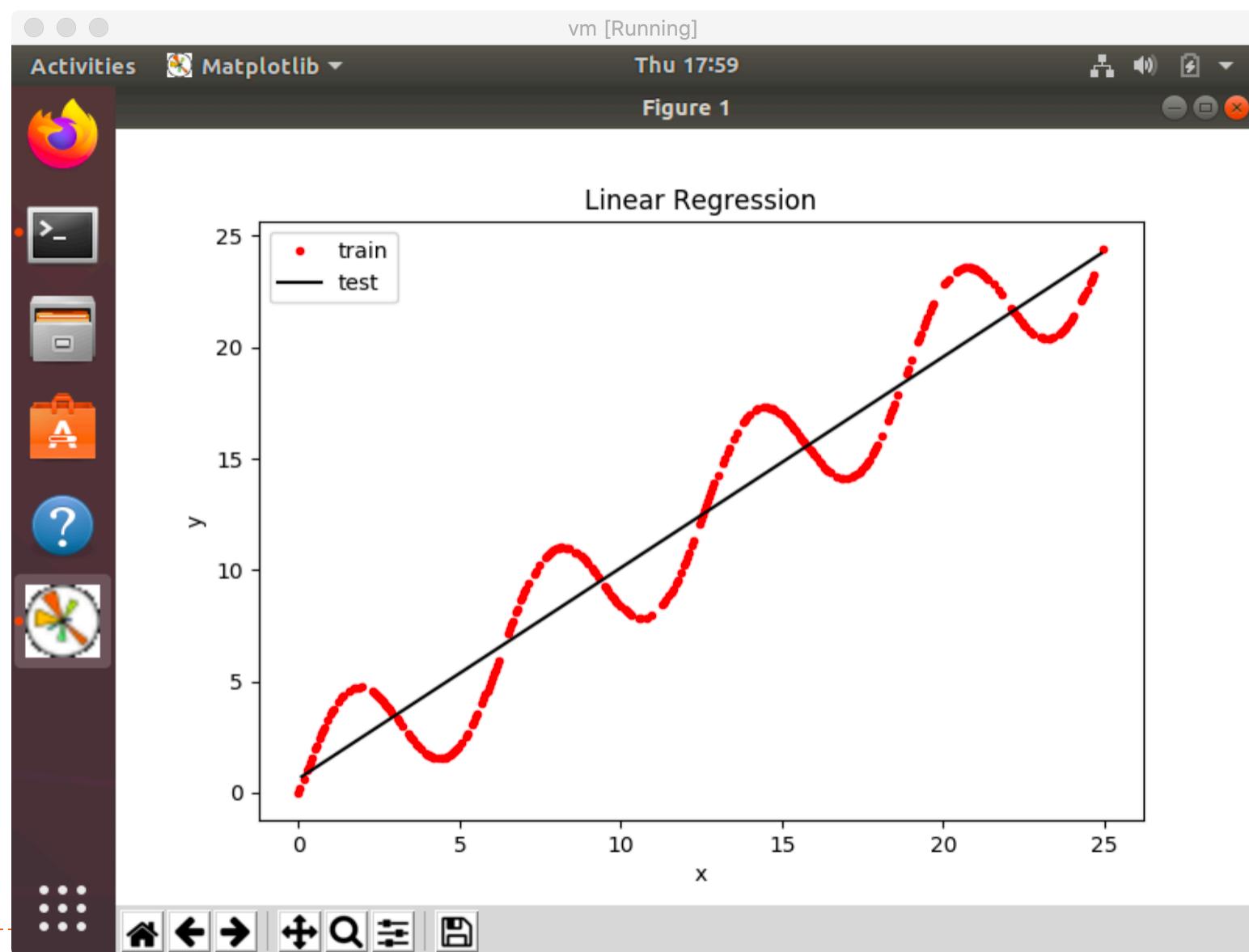
Polynomial function



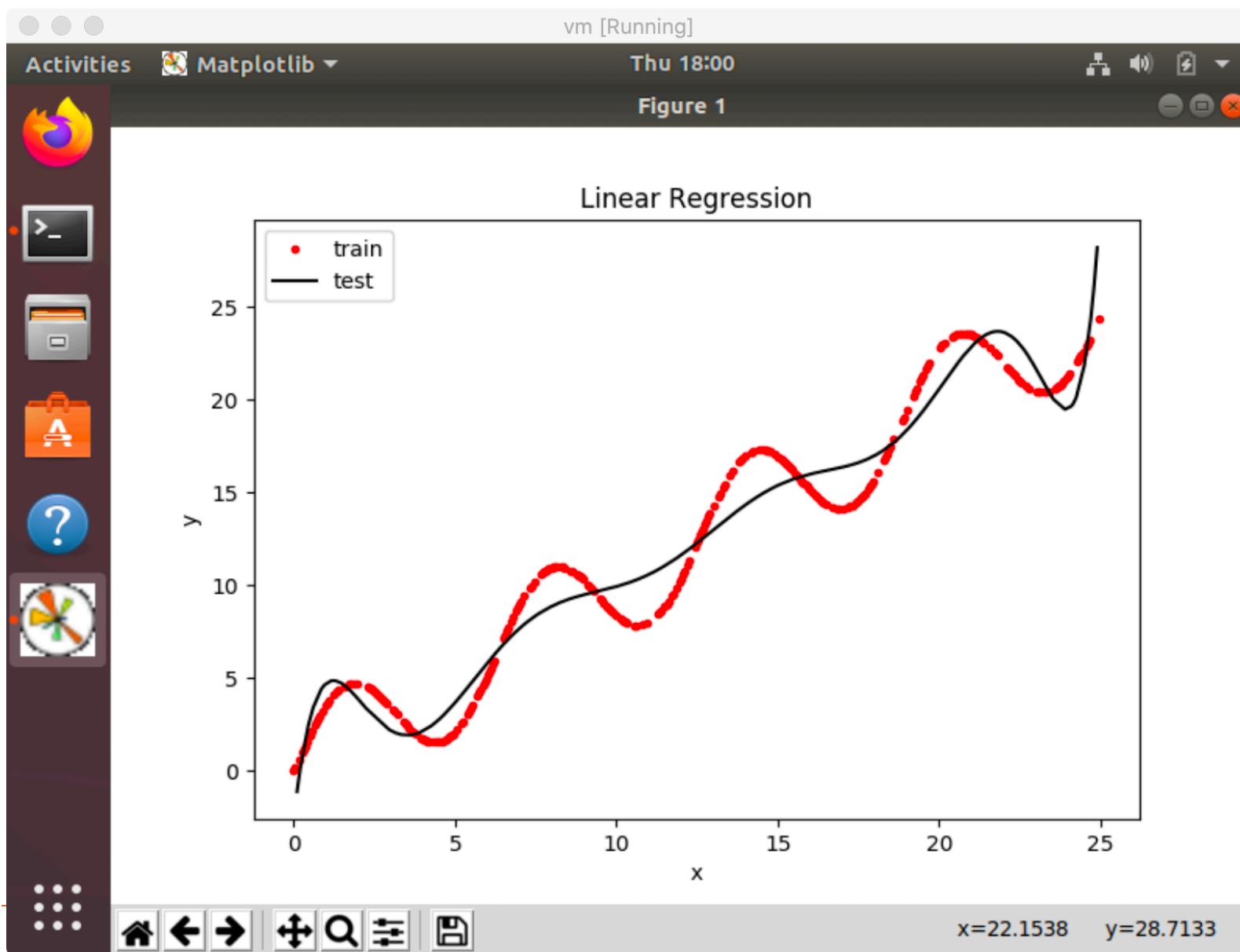
$$f(x) = a \cdot \exp\left(-\frac{(x-b)^2}{2c^2}\right)$$

Gaussian function

# Execution: identity\_basis



# Execution: multinomial\_basis



# Execution: gaussian\_basis

