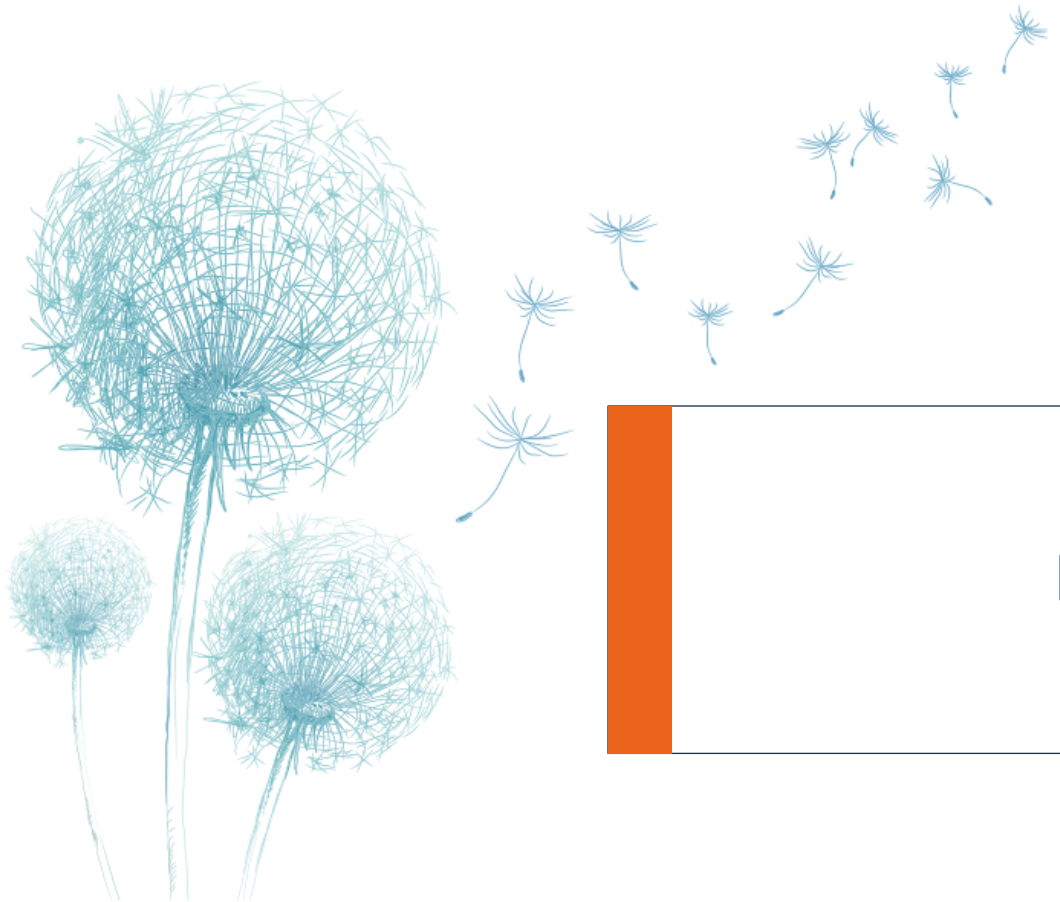
Neural networks and deep learning

Machine learning overview

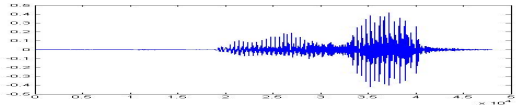
Kun Suo
Computer Science, Kennesaw State University
https://kevinsuo.github.io/



Machine learning

Machine learning ≈ build a mapping function

- Speech Recognition

$f($  $)=$ “Hello”

- Image Identification

$f($  $)=$ “Cat”

- Go

$f($  $)=$ “5-5” (position)

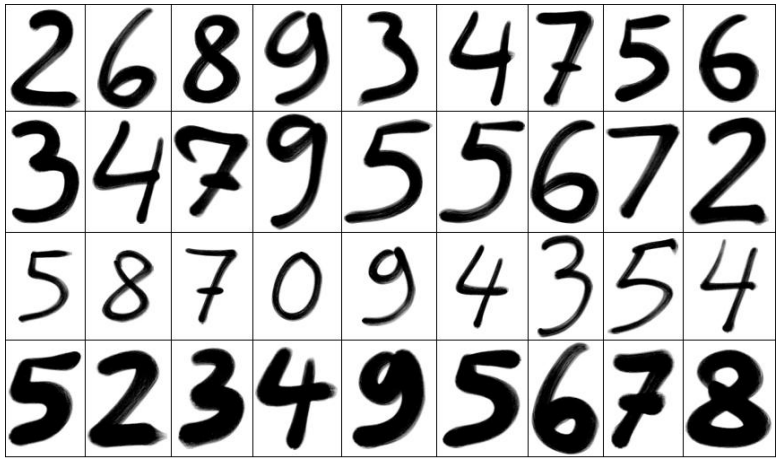
- Dialogue system

$f($ “Hi” $)=$ “The weather is nice today”

User input Machine Output

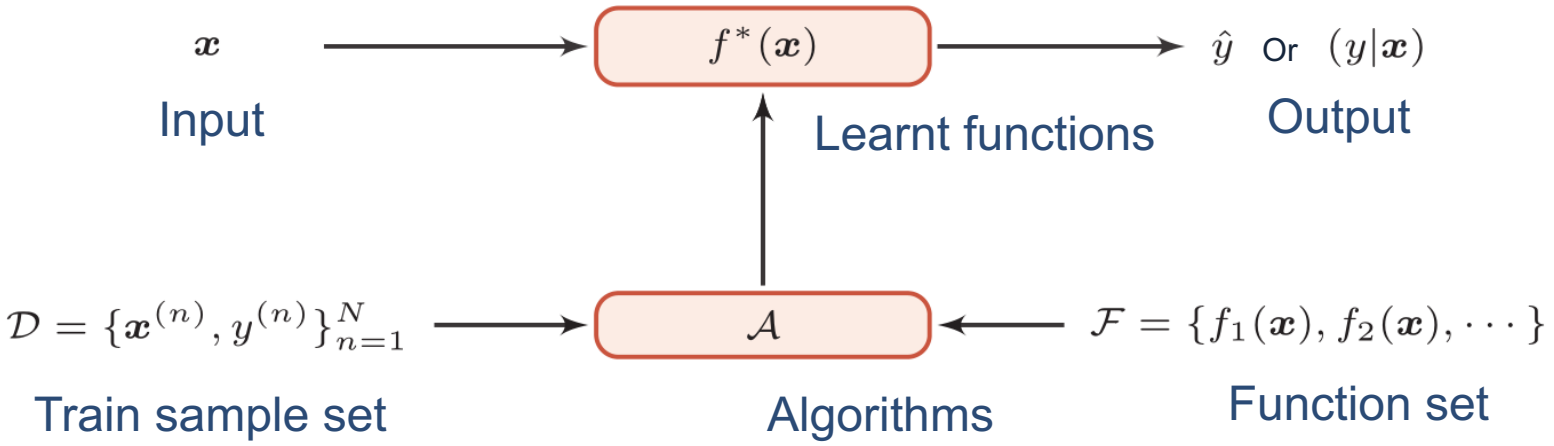
Why we need "machine learning"?

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



What is machine learning?

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



The three elements of machine learning

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

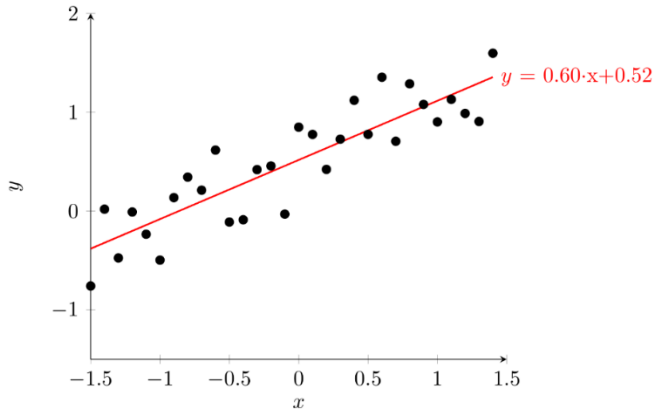
- Study criteria
  - Expected risk

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

- Optimization
  - Gradient descent

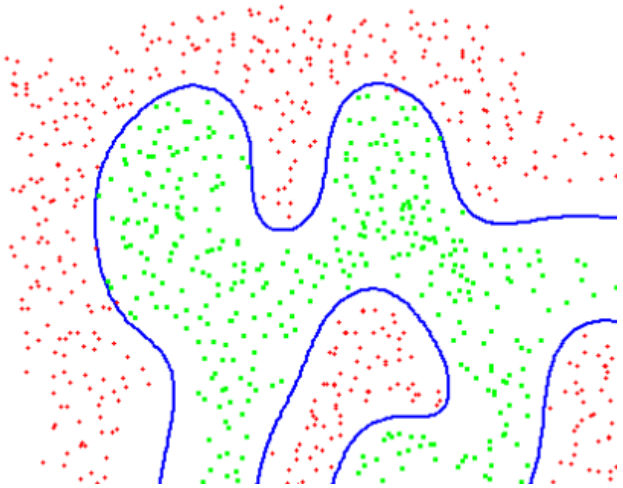
Common machine learning problems and models

Predict y under x=20



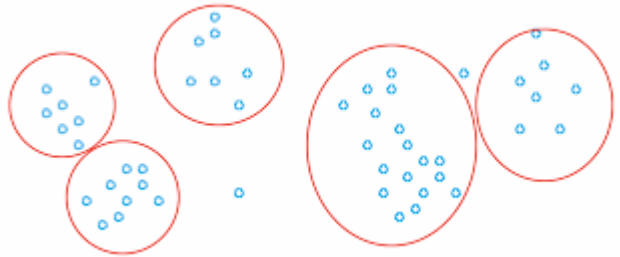
regression

Which group point (3,4) is?



classification

How many types are in my data set?

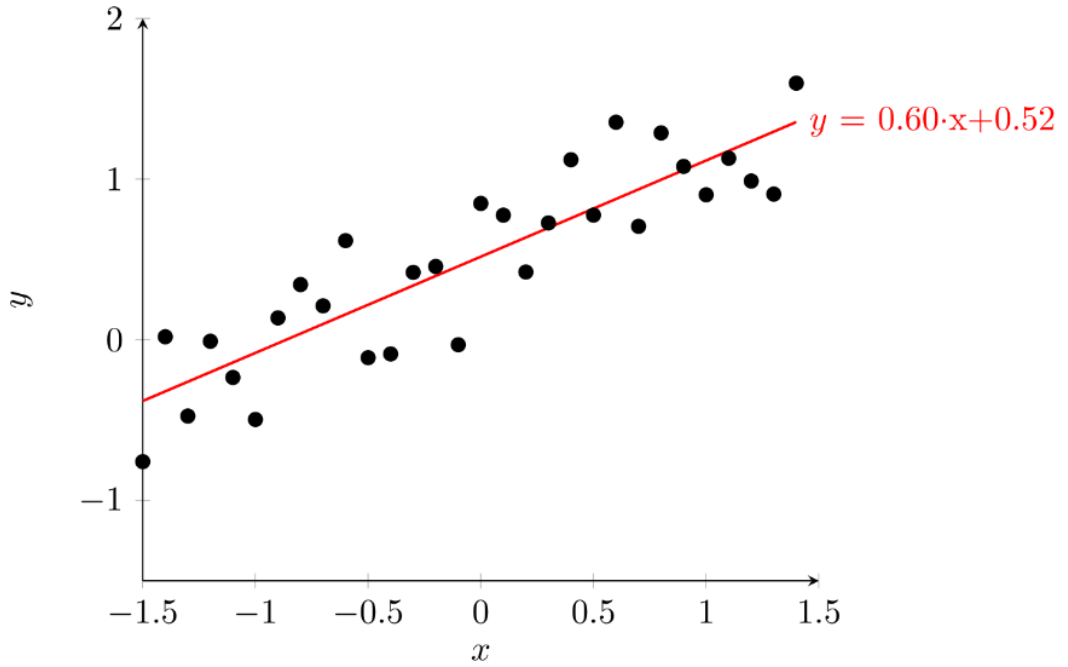


clustering

Model

- Take Linear Regression as an example
  - Model:

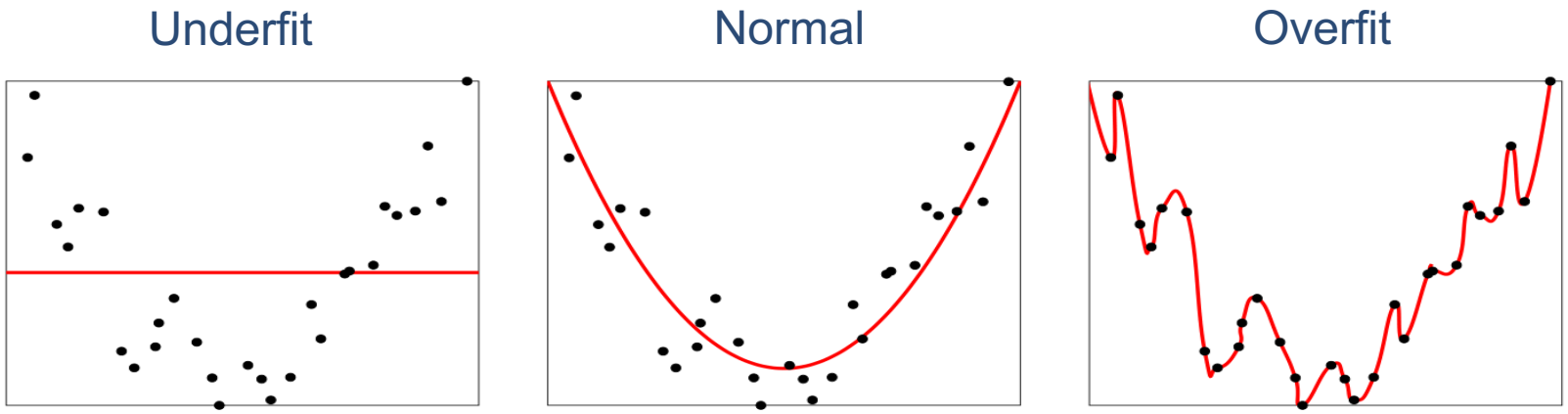
$$f(\boldsymbol{x};\theta) = \boldsymbol{w}^{\top}\boldsymbol{x} + b$$



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***.



Underfitting problems occurs when a model is too simple — informed by too few features or regularized too much.

Overfitting problems are often caused by reasons such as less training data and noise.

Model selection: bias and variance

Bias and variance decomposition

- The expected error can be broken down into

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

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

$$\mathbb{E}_{\mathbf{x}}\left[\mathbb{E}_{\mathcal{D}}\left[\left(f_{\mathcal{D}}(\mathbf{x}) - \mathbb{E}_{\mathcal{D}}\left[f_{\mathcal{D}}(\mathbf{x})\right]\right)^2\right]\right]$$

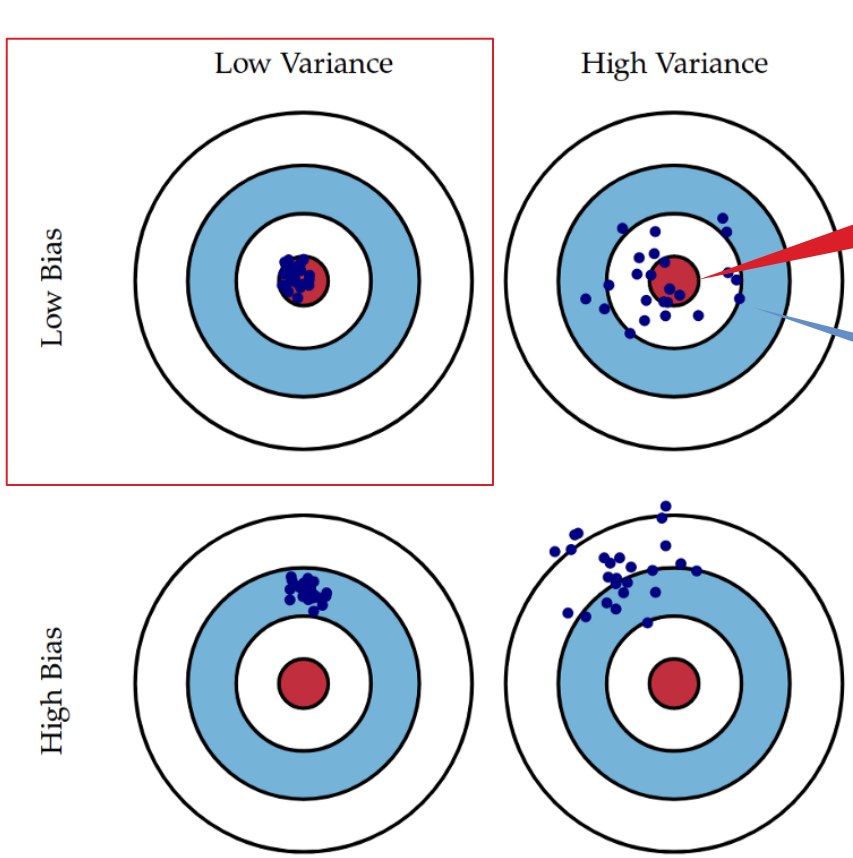
Model selection: bias and variance

- 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

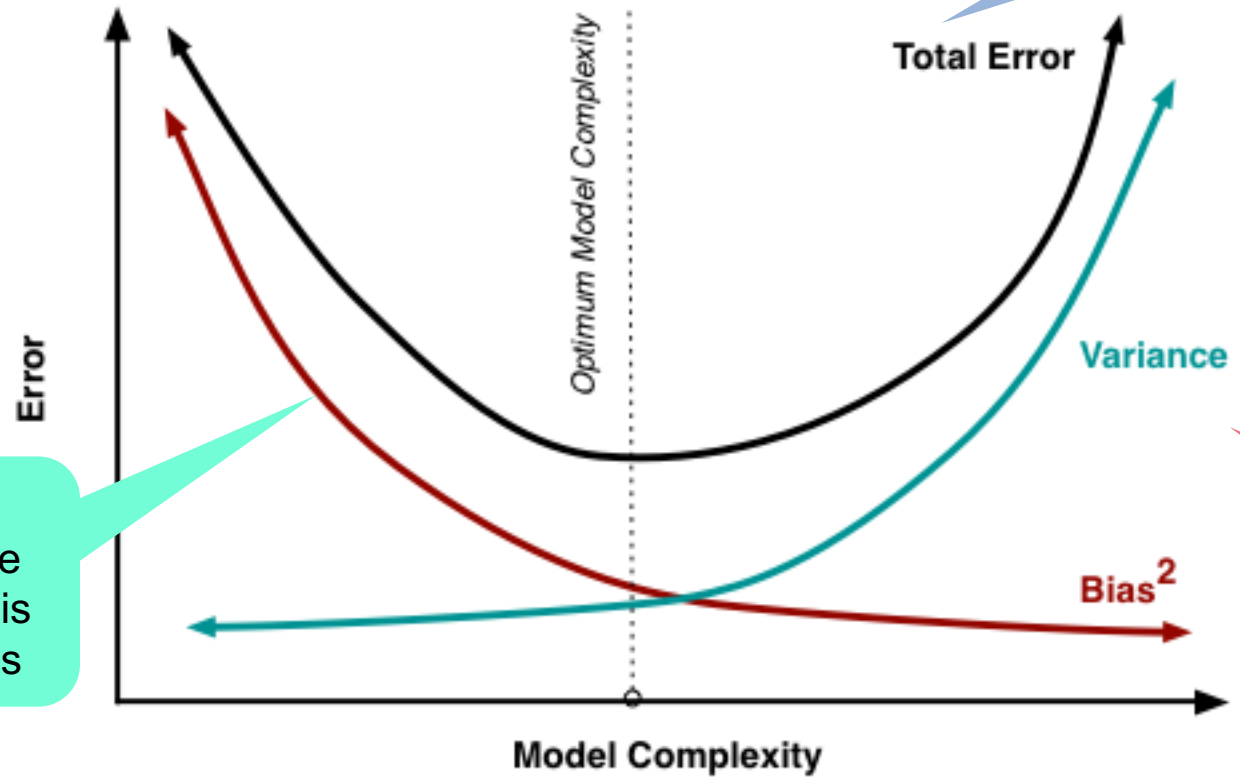


The perfect correct prediction value of the learning algorithm

The predicted value of the model trained by the training data set

Model selection: bias and variance

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



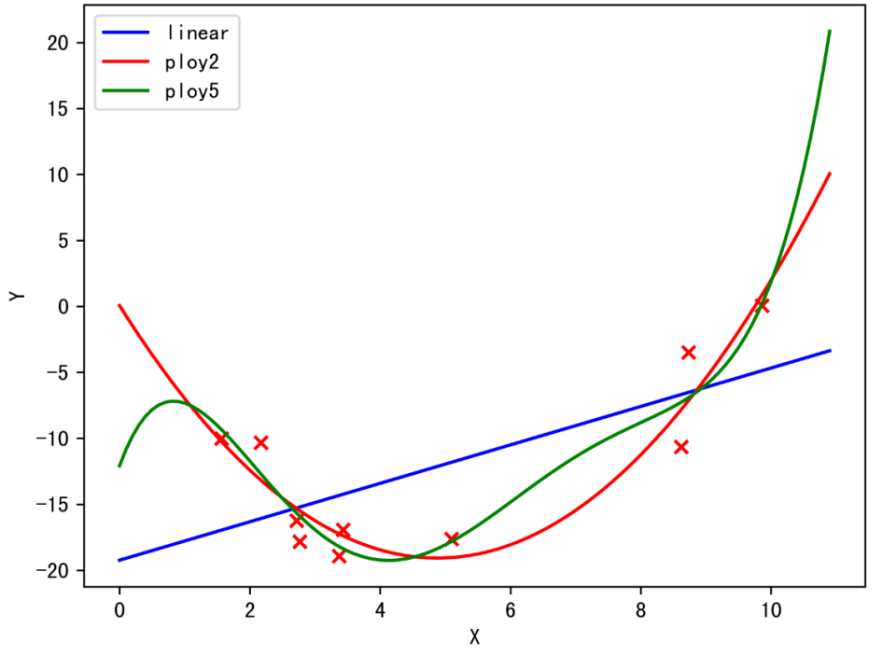
A model with too low complexity cannot fit the training data well, but it is easy to cause large bias

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.

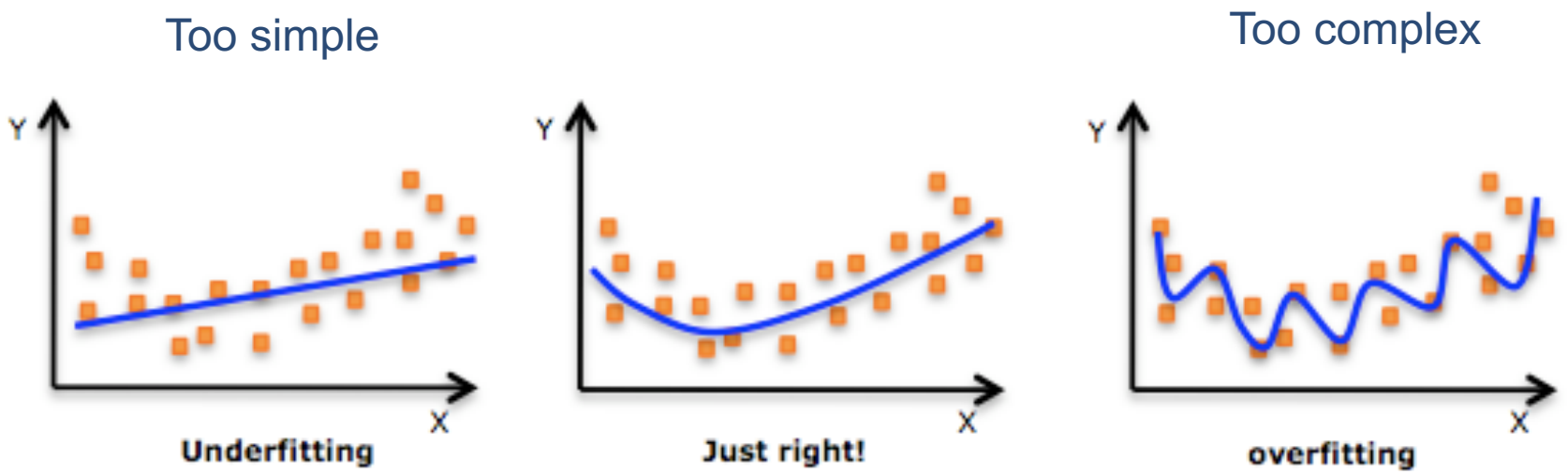
  - Too small data is always bad

  - 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



Study criteria

- A good model $f(\boldsymbol{x}, \theta^*)$ should have the value of each (x, y) in space consistent with the real mapping function $y = g(\boldsymbol{x})$

- In order to describe the quality of the model, we introduce the expected risk to measure:

$$\mathcal{R}(\theta) = \mathbb{E}_{(\boldsymbol{x},y)\sim p_r(\boldsymbol{x},y)}[\mathcal{L}(y, f(\boldsymbol{x};\theta))]$$

$p_r(x, y)$ is real data distribution

$L(y, f(x; \theta))$ is the loss function

Study criteria

- Loss function
  - 0-1 Loss function

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

  - 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: $\mathcal{D} = \{\mathrm{x}^{(\mathrm{n})}, y^{(\mathrm{n})}\}, i \in [1, N]$, θ is the model parameter

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

- Minimize empirical risk
  - After selecting the appropriate risk function, we look for a parameter θ* to minimize the empirical risk function.

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

- Machine learning problem turned into an optimization problem

Study criteria

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

Empirical risk minimization

Linear model with empirical risk minimization (least squares method)

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

- Study criteria

$$\mathcal{R}(\mathbf{w}) = \sum_{n=1}^{N} \mathcal{L}(y^{(n)}, f(\mathbf{x}^{(n)}; \mathbf{w}))$$

$$= \frac{1}{2} \sum_{n=1}^{N} \left(y^{(n)} - \mathbf{w}^{\mathrm{T}}\mathbf{x}^{(n)} \right)^2$$

$$= \frac{1}{2} \|\mathbf{y} - X^{\mathrm{T}}\mathbf{w}\|^2,$$

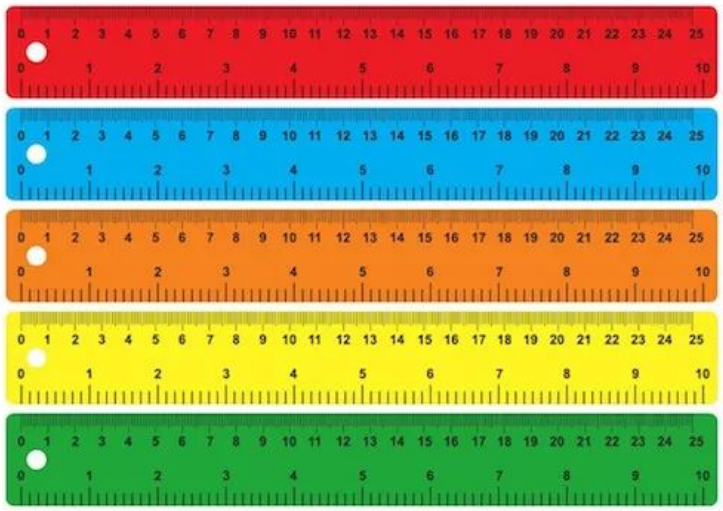
$Suppose\ \mathrm{L}(a, b) = \frac{1}{2} * (a - b)^2$

- Optimization

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

Empirical risk minimization (least squares method)

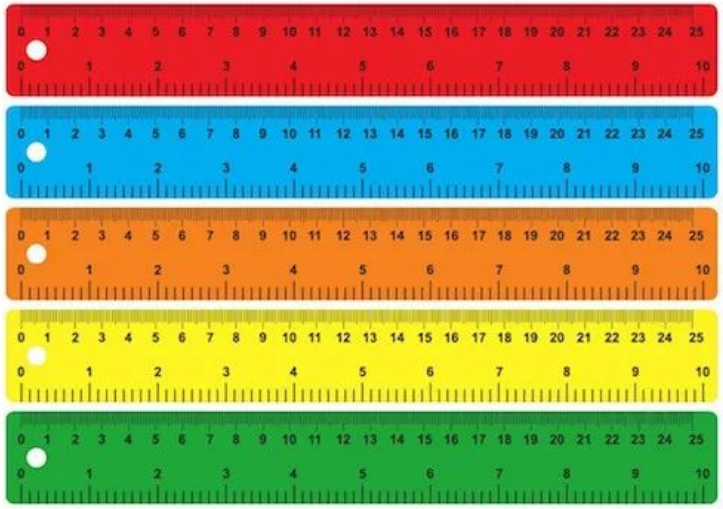
- Five rulers from five brands



	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

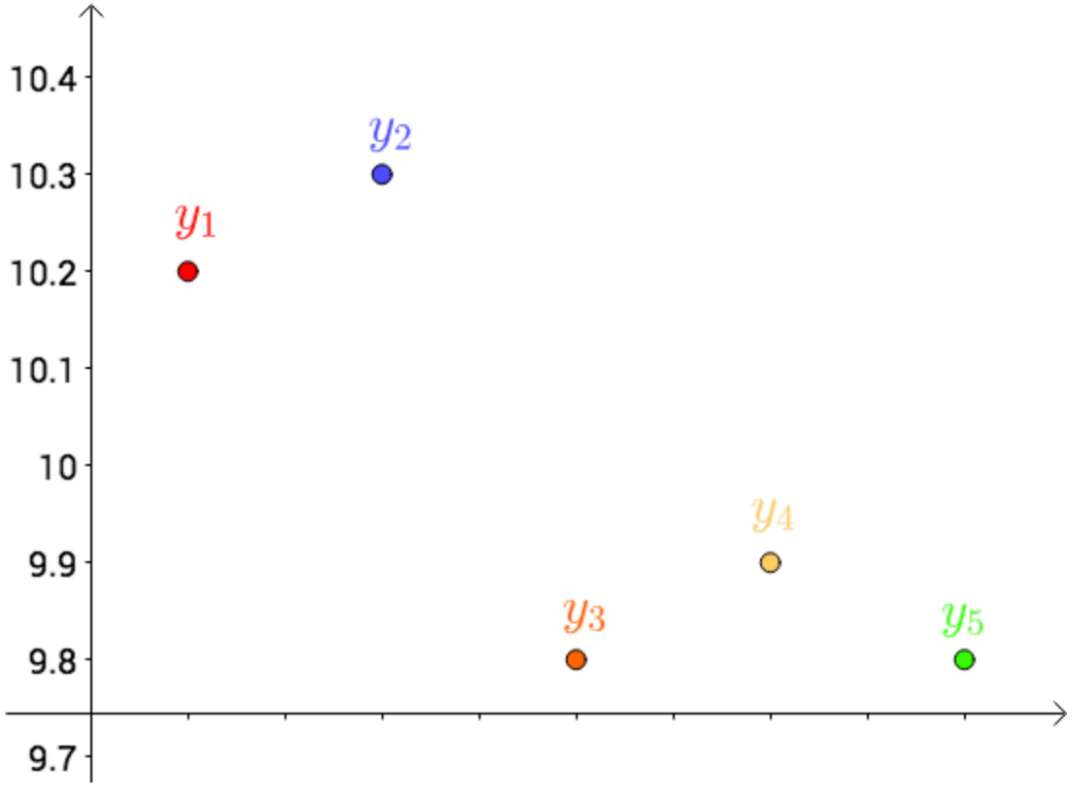
The material of the ruler is different, and the heat expansion and contraction are different

…

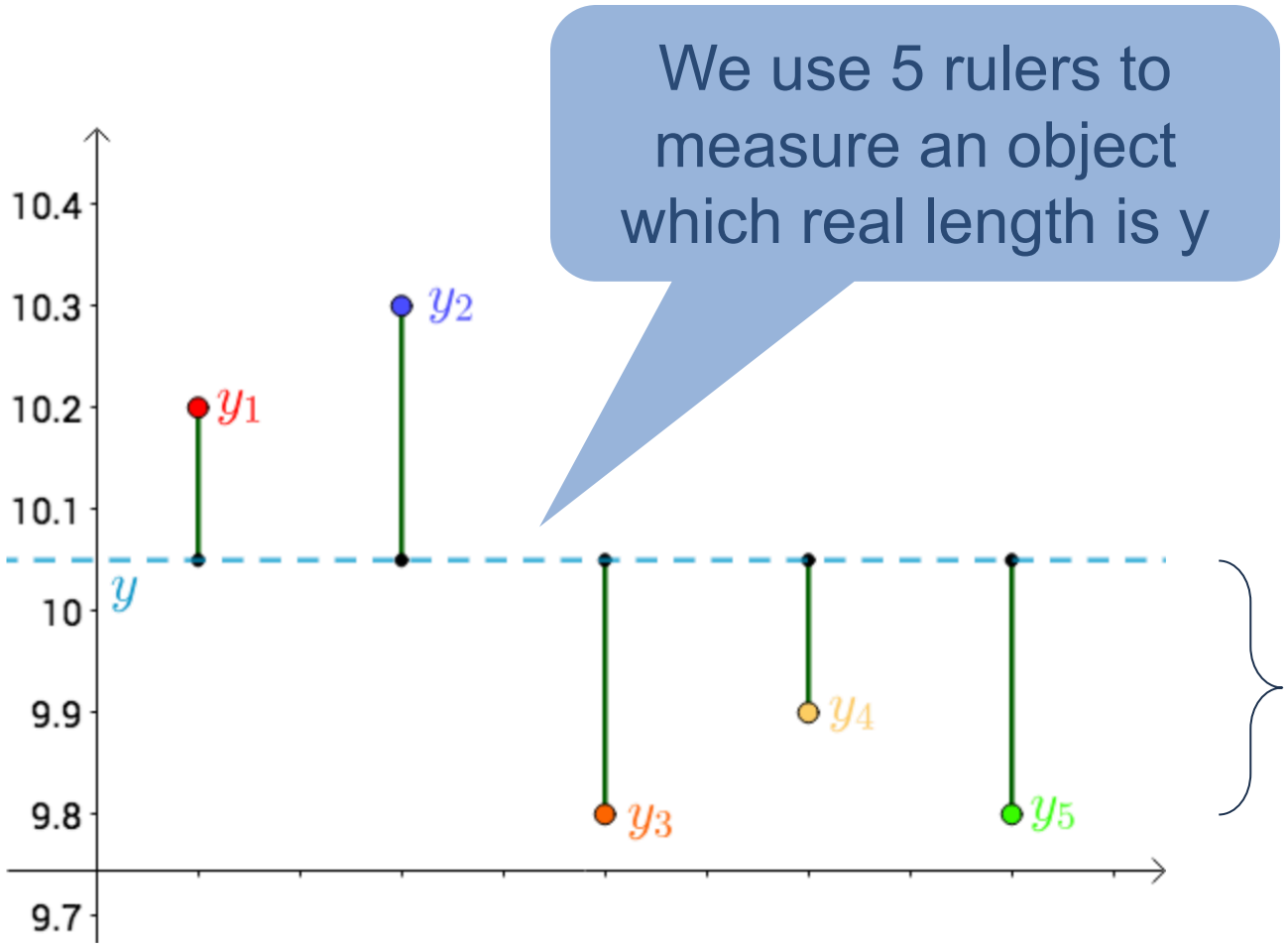
Empirical risk minimization (least squares method)

- Five rulers from five brands

	Length
Red	10.2
Blue	10.3
Orange	9.8
Yellow	9.9
Green	9.8



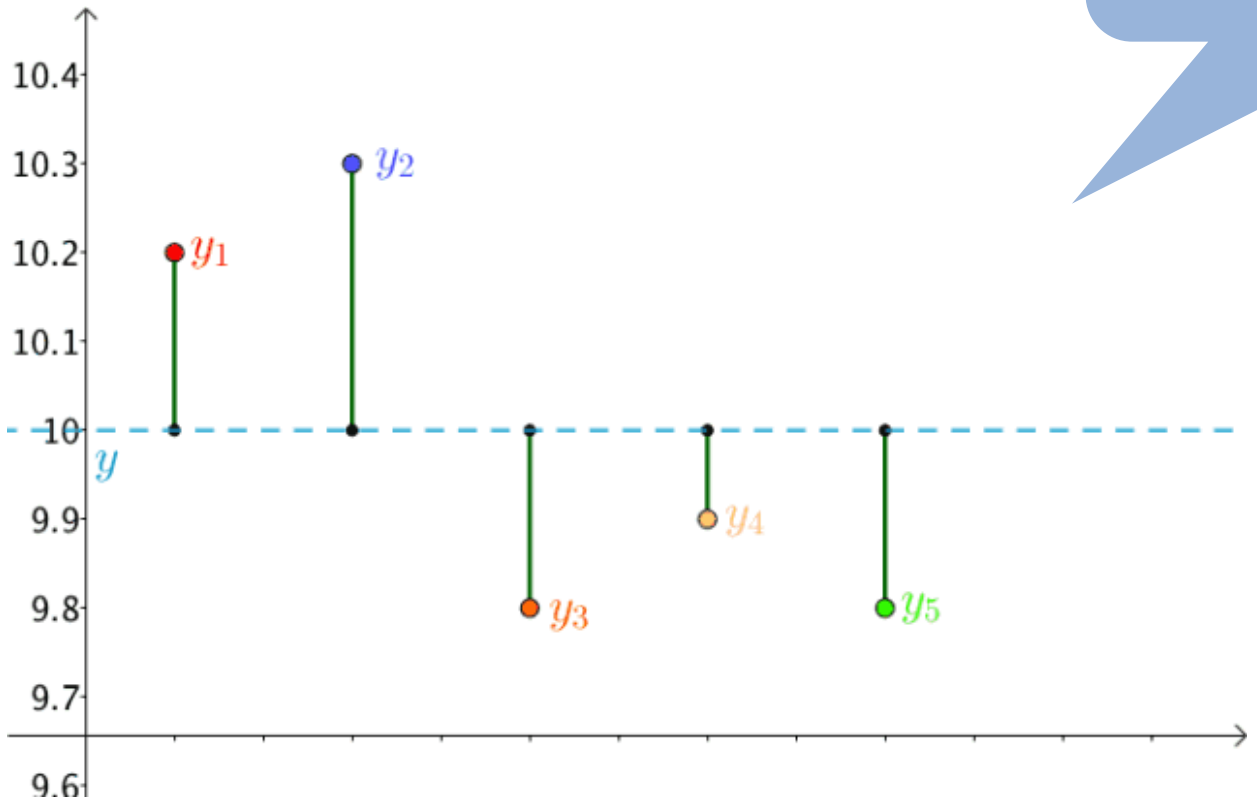
Empirical risk minimization (least squares method)



Error: $|y - y_i|$

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

Empirical risk minimization (least squares method)



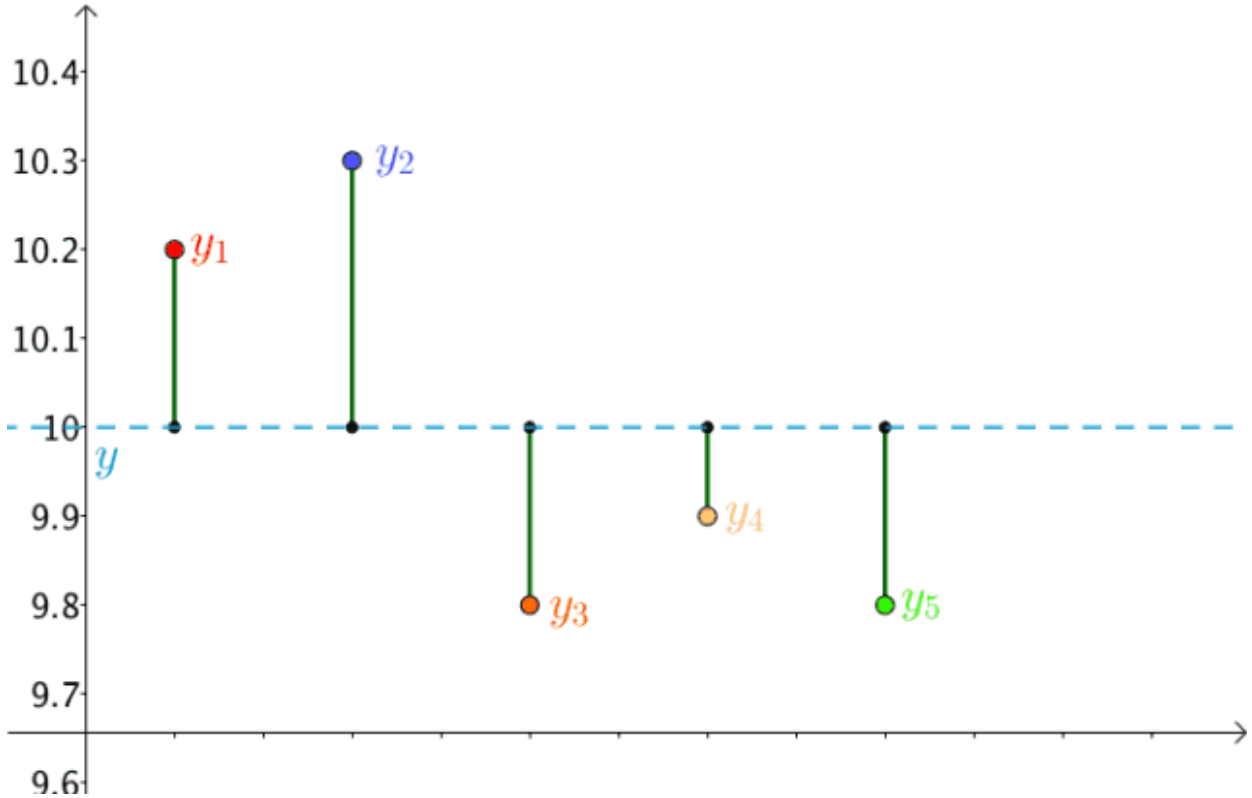
Different objects have various errors for the five rulers.

Error: $|y - y_i|$

S is different for various y

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

Empirical risk minimization (least squares method)



Empirical risk minimization (least squares method) founds

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

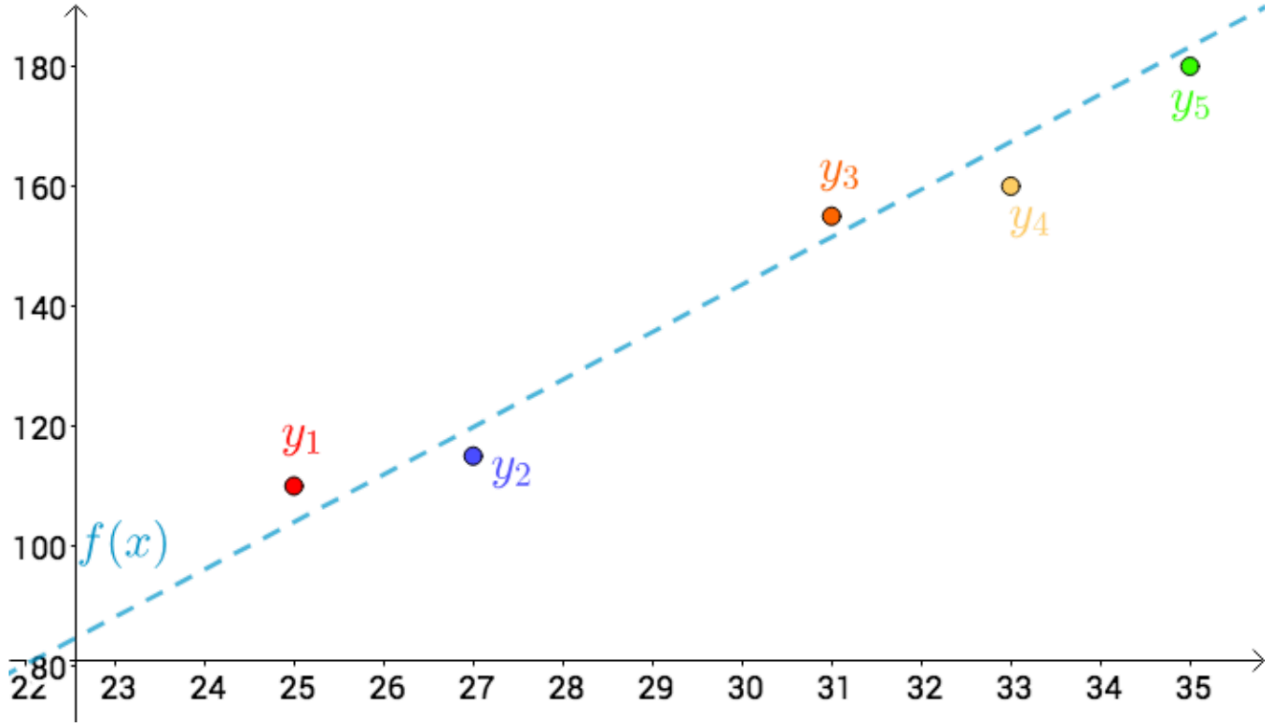
can make minimize S_{ϵ^2}

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

Empirical risk minimization (least squares method)

Temperature and ice cream sales:

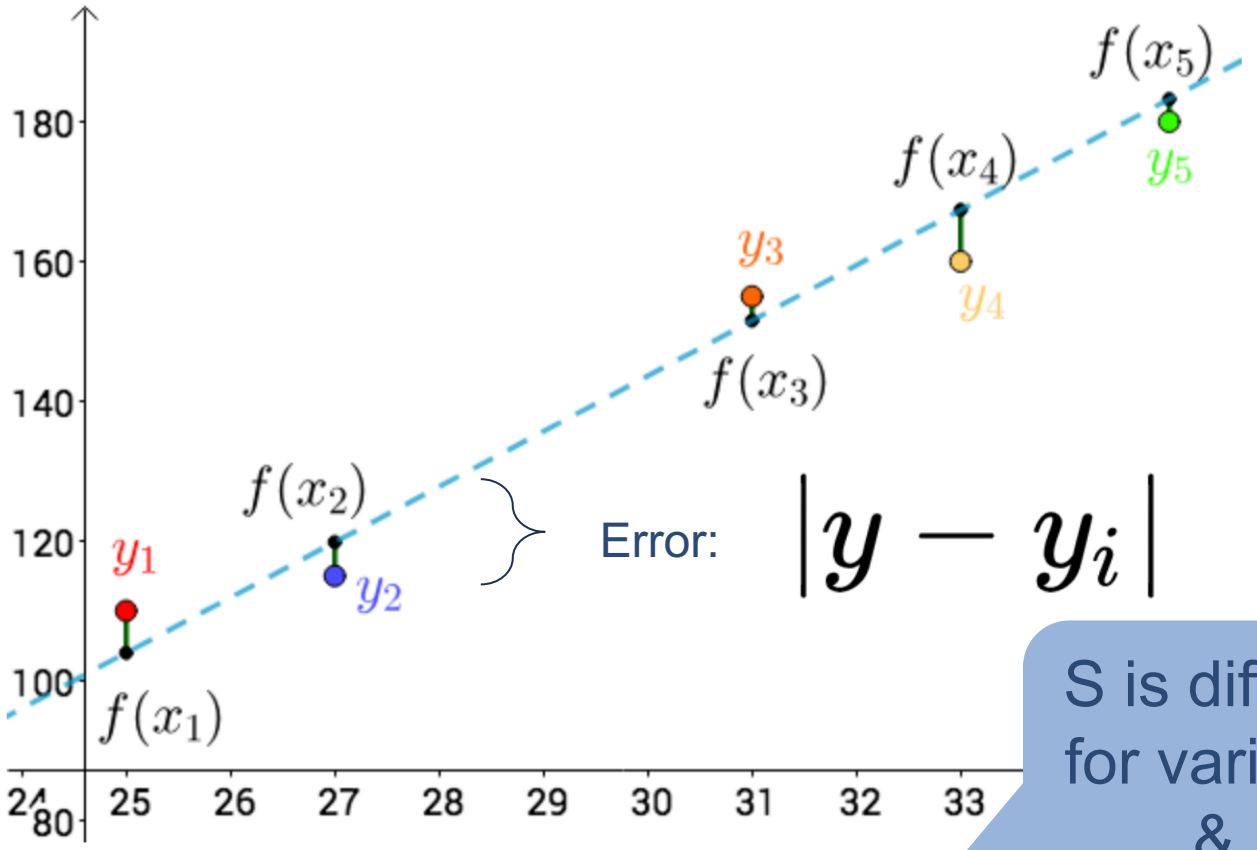
Temp (°C)	sales
25	110
27	115
31	155
33	160
35	180



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 is different for various a & b

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

Empirical risk minimization (least squares

S is different for various a & b

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

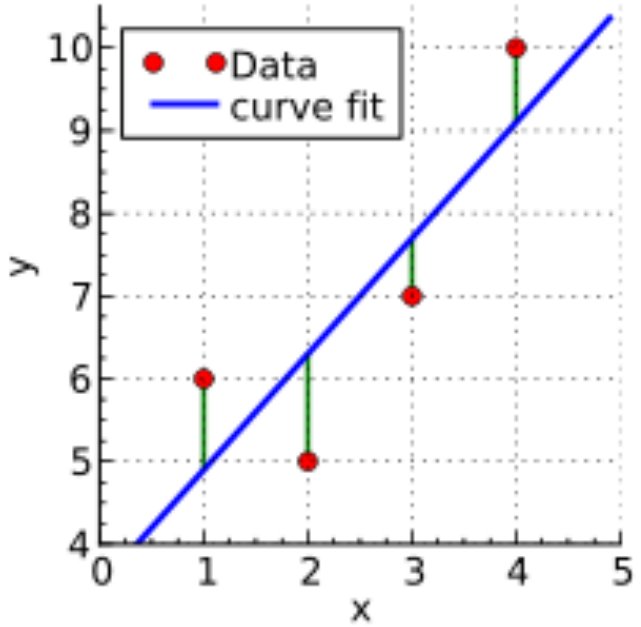
When

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

S_{ϵ^2} is minimum

Empirical risk minimization (least squares method)

- Example: data set (x,y): {(1,6), (2,5), (3,7), (4,10)}, We hope to find a straight line that best matches these four points



$$y = \beta_1 + \beta_2 x$$

$$\begin{aligned}
\beta_1 + 1\beta_2 &= 6 \\
\beta_1 + 2\beta_2 &= 5 \\
\beta_1 + 3\beta_2 &= 7 \\
\beta_1 + 4\beta_2 &= 10
\end{aligned}$$

$$\begin{aligned}
S(\beta_1, \beta_2) = &[6 - (\beta_1 + 1\beta_2)]^2 + [5 - (\beta_1 + 2\beta_2)]^2 \\
&+ [7 - (\beta_1 + 3\beta_2)]^2 + [10 - (\beta_1 + 4\beta_2)]^2
\end{aligned}$$

Empirical risk minimization (least squares method)

- To get the minimum value, we can use partial derivatives of

$$S(\beta_1, \beta_2)$$

$$\frac{\partial S}{\partial \beta_1} = 0 = 8\beta_1 + 20\beta_2 - 56$$

$$\frac{\partial S}{\partial \beta_2} = 0 = 20\beta_1 + 60\beta_2 - 154$$

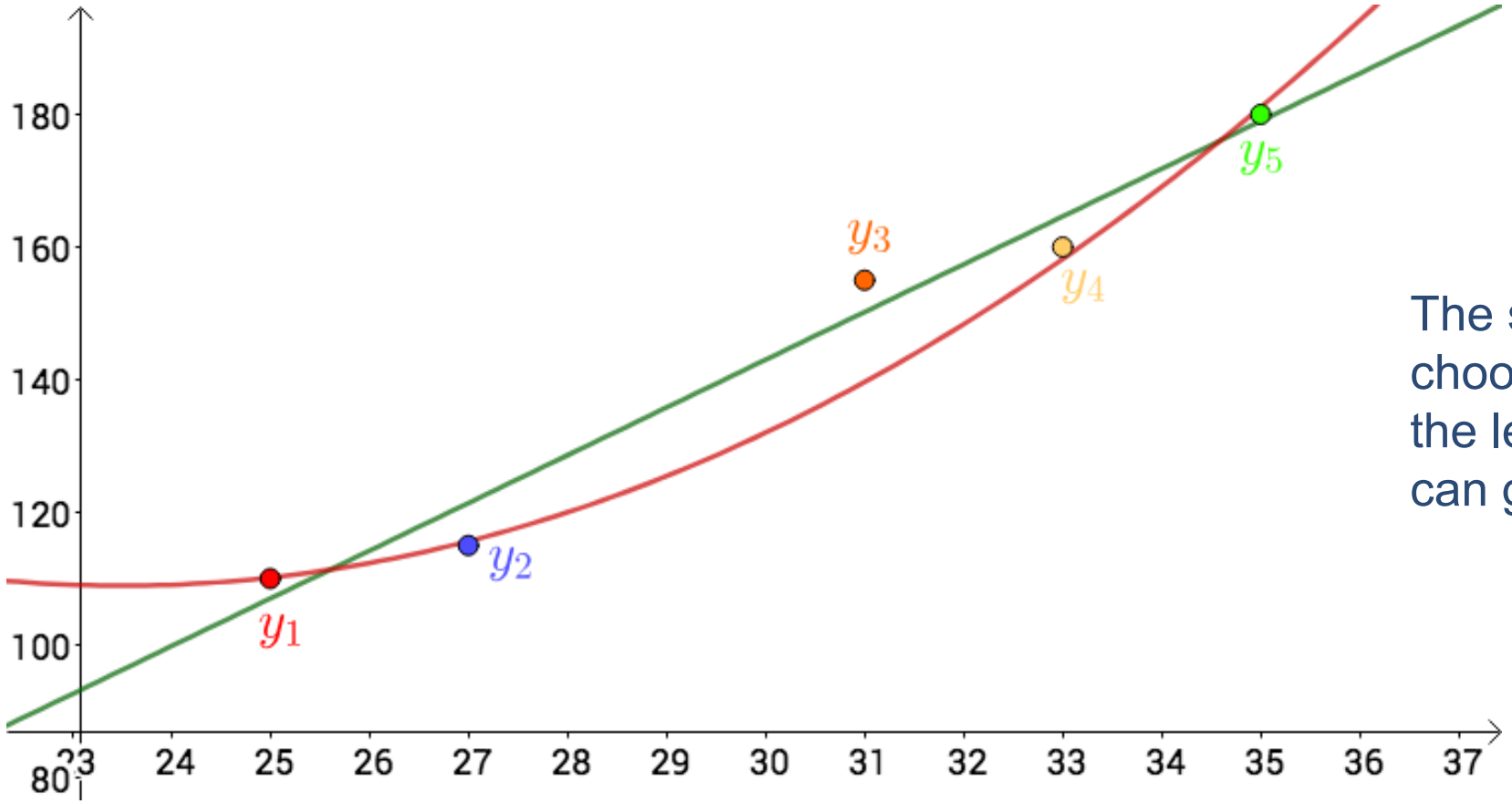
In this way, a system of equations with only two unknowns is obtained, which can be easily solved:

$$\beta_1 = 3.5$$

$$\beta_2 = 1.4$$

That is to say straight $y = 3.5 + 1.4x$ Is the best.

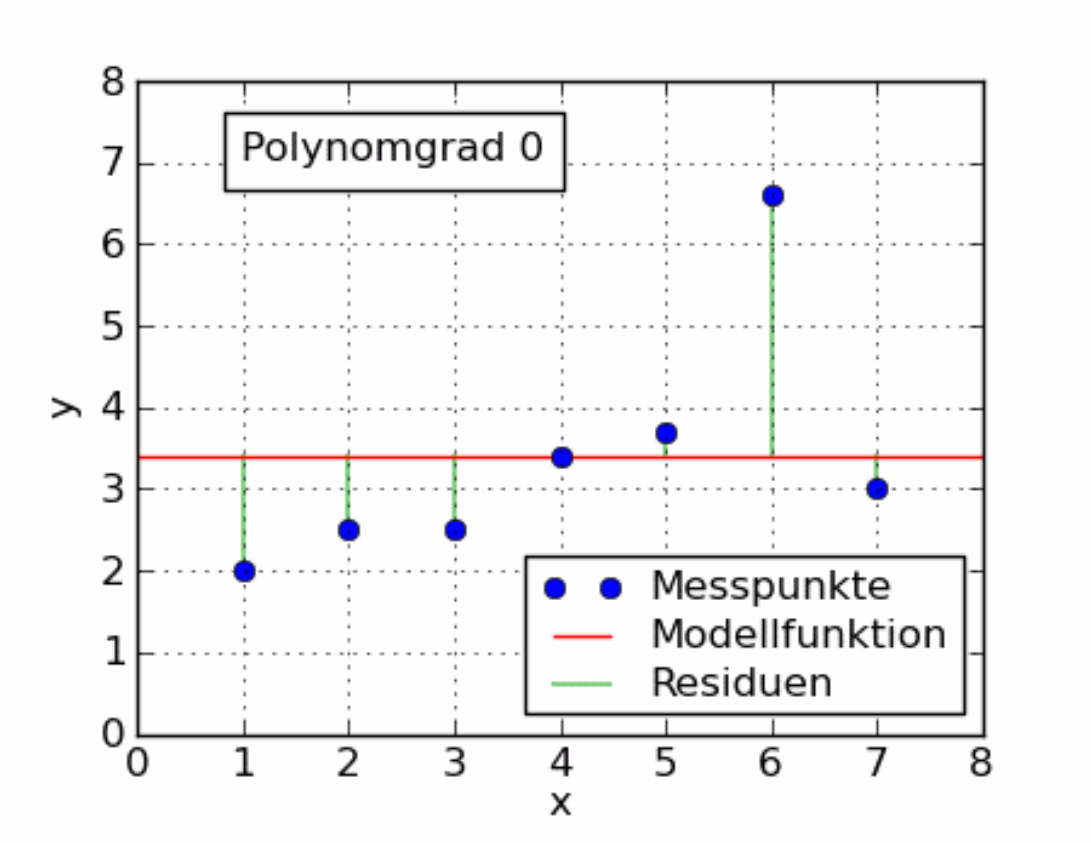
Empirical risk minimization (least squares method)

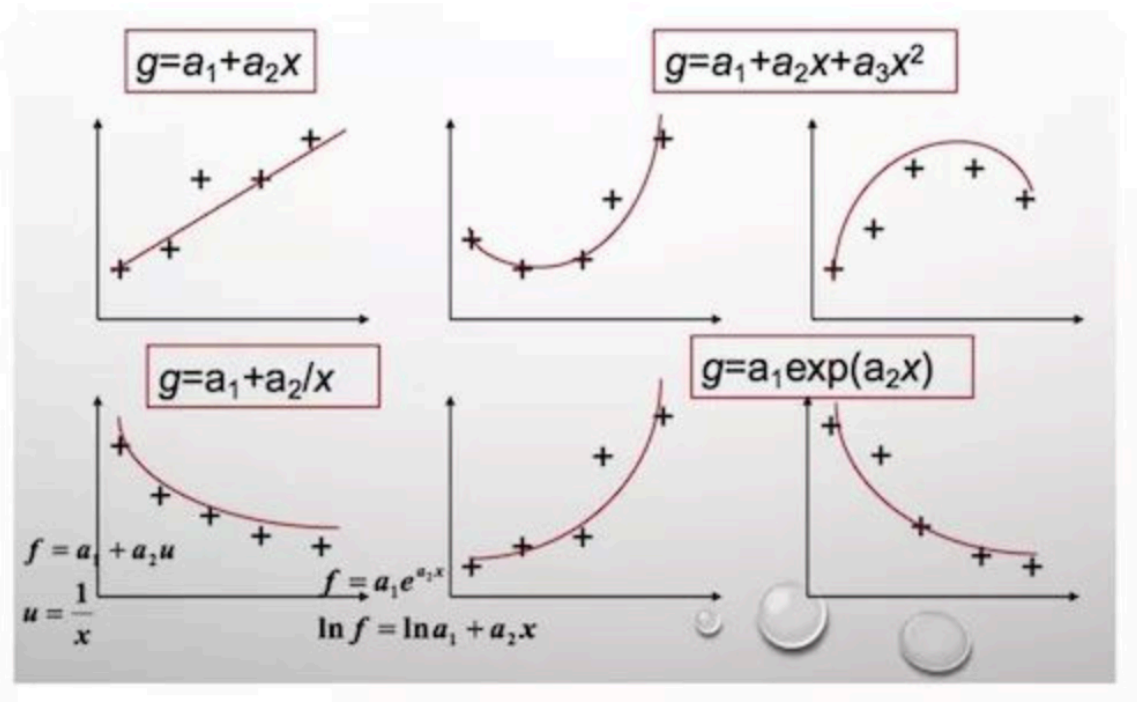


The same set of data, choose different f(x), through the least square method we can get different fitting curves

f(x) is called base function

Empirical risk minimization (least squares method)





Empirical risk minimization (least squares method)

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

- Study criteria

$$\mathcal{R}(\mathbf{w}) = \sum_{n=1}^{N} \mathcal{L}(y^{(n)}, f(\mathbf{x}^{(n)};\mathbf{w}))$$

$$= \frac{1}{2}\sum_{n=1}^{N}\left(y^{(n)} - \mathbf{w}^{\mathrm{T}}\mathbf{x}^{(n)}\right)^2$$

$$= \frac{1}{2}\|\mathbf{y} - X^{\mathrm{T}}\mathbf{w}\|^2,$$

$Suppose\ \mathrm{L}(a,b) = \frac{1}{2} * (a-b)^2$

- Least squares method is a mathematical optimization technique that finds the best function match of the data by minimizing the sum of squares of errors

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.

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

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

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

Some basic concepts about probability

Bernoulli Distribution

- In an experiment, the probability of occurrence of event A is μ, and the probability of not occurring is 1 − μ. 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,...,n}, and the corresponding distribution

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

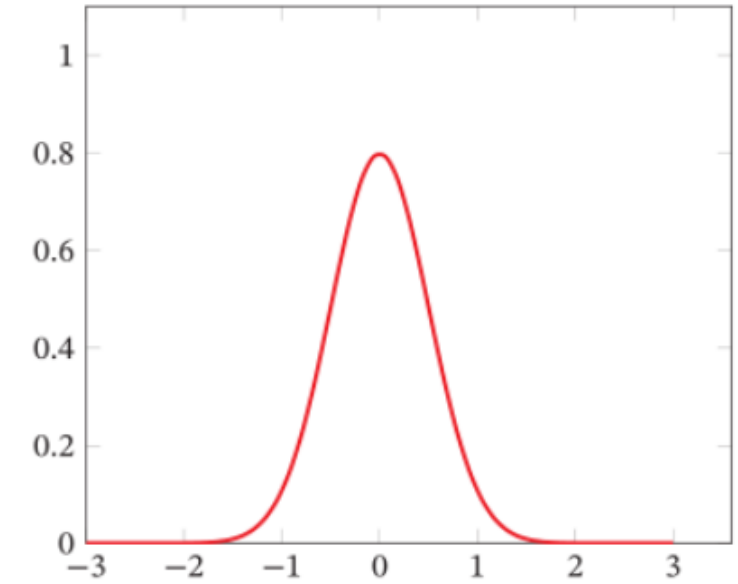
Some basic concepts about probability

Gaussian Distribution (normal distribution)

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

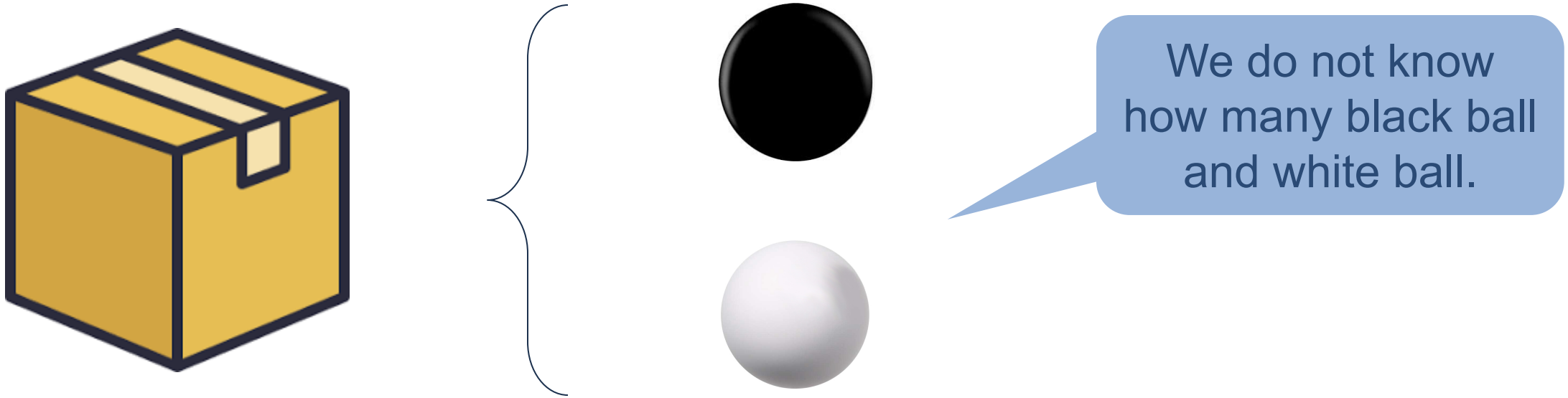
$$X \sim \mathcal{N}(\mu, \sigma^2) \qquad p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Expectation is μ
Variance is σ^2

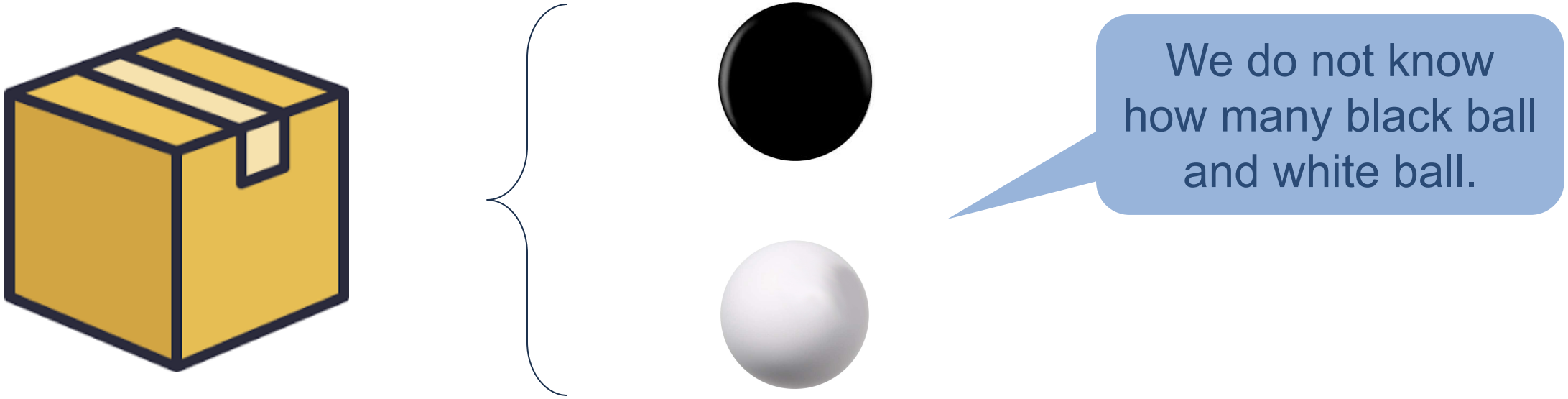


Maximum likelihood estimation

- 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



Maximum likelihood estimation



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(Data | M), where Data is all the data, and M is the given model

P(Data | M)

= P(x1,x2,...,x100|M)

= P(x1|M)P(x2|M)...P(x100|M)

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

What value of p takes, the value of P(Data |M) is the largest

Maximum likelihood estimation

What value of p takes, the value of P(Data |M) is the largest

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

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

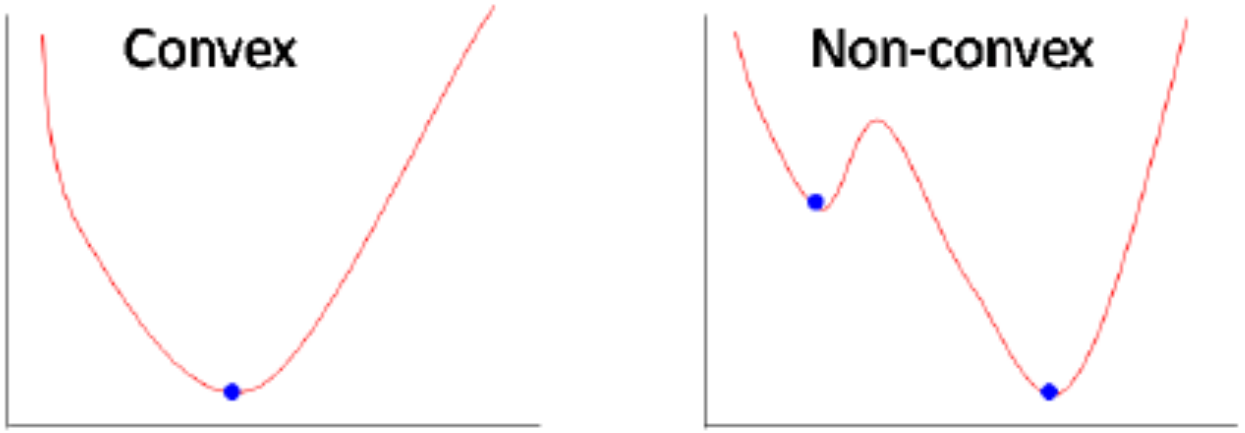
$$70 * p^{69}\,(1-p)^{30} = p^{70} * 30 * (1-p)^{29}$$

$$70 * (1-p) = p * 30$$

➔ P = 0.7

Optimization problem

- When you set up the study criteria, machine learning problem $f(\boldsymbol{x}, \theta^*)$ is turned into an optimization problem. (Here θ is the parameter for the model)



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

Optimization methods

- 1) Gradient descent
- 2) Stop early
- 3) Stochastic gradient descent
- 4) Mini-batch gradient descent

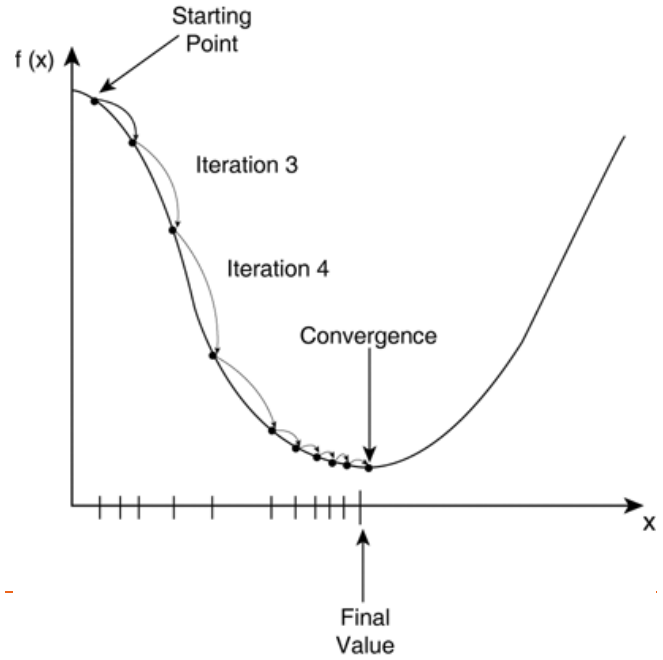
Gradient Descent

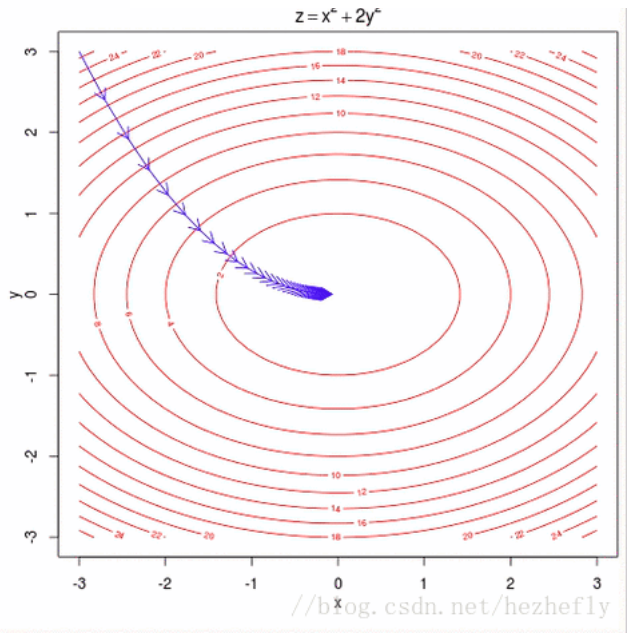
- Gradient descent method: the simplest and most widely used optimization algorithm in machine learning

First fix θ_0, then calculate θ_{t+1} by iterative method to minimize the loss function

$$\theta_{t+1} = \theta_t - \alpha \frac{\partial \mathcal{R}(\theta)}{\partial \theta_t}$$

$$= \theta_t - \alpha \frac{1}{N} \sum_{i=1}^{N} \frac{\partial \mathcal{L}\left(\theta_t; x^{(i)}, y^{(i)}\right)}{\partial \theta}.$$

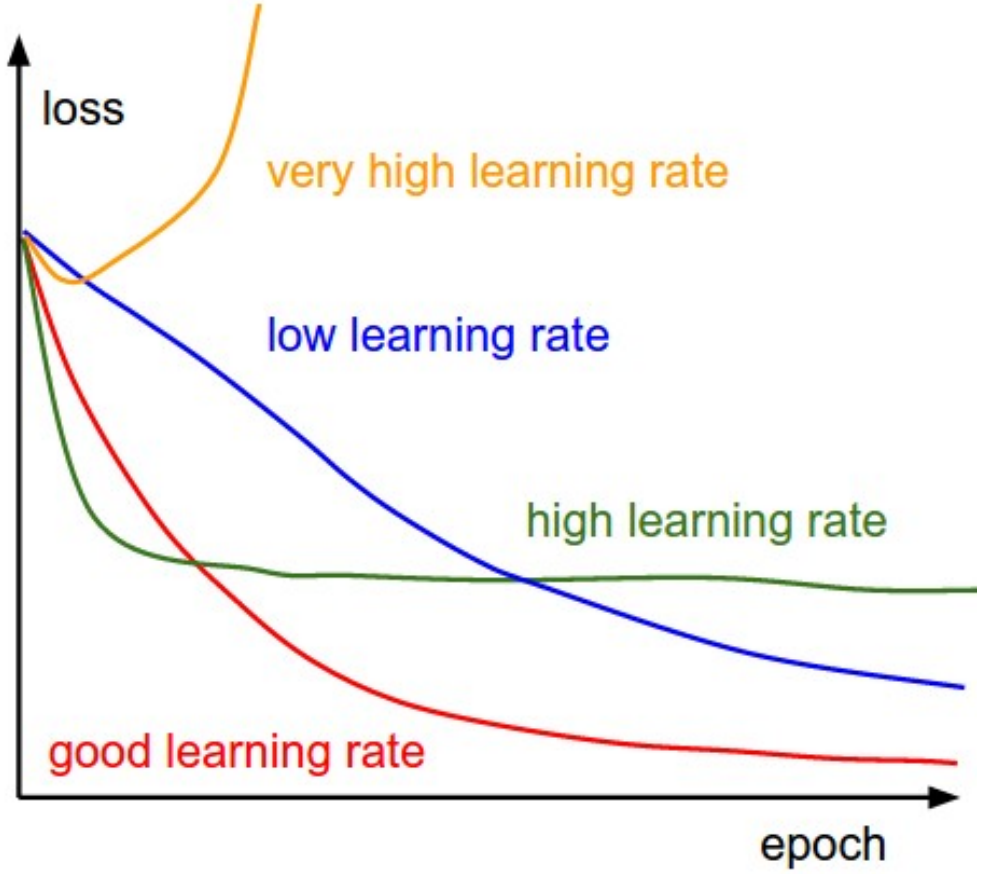


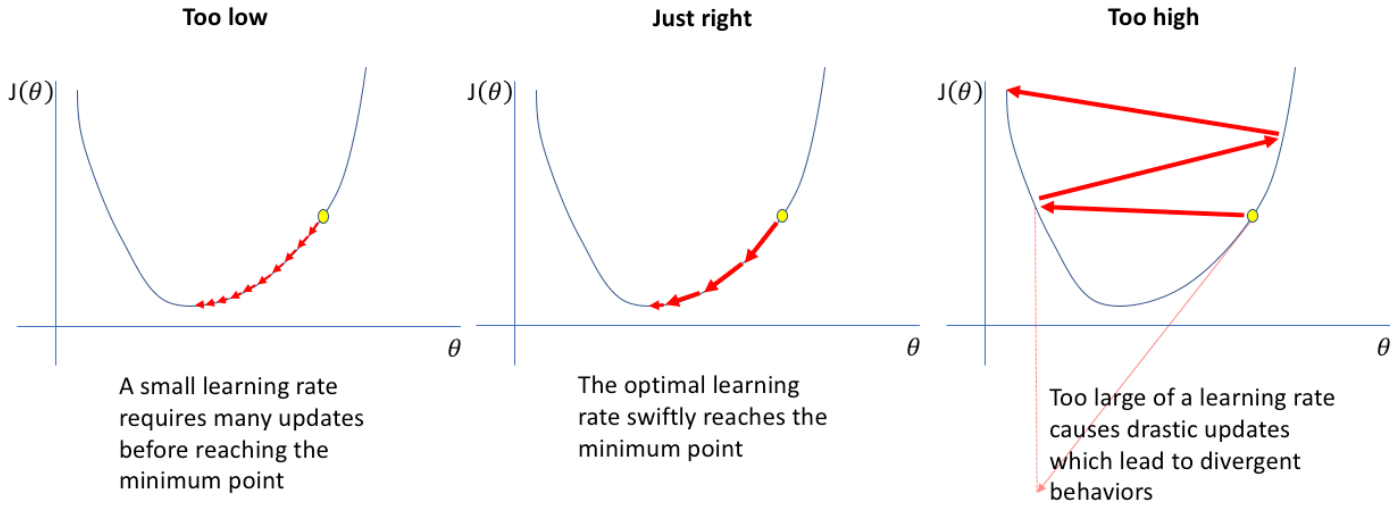


Search step α is also called *Learning Rate*

Learning rate is a very important hyperparameter!

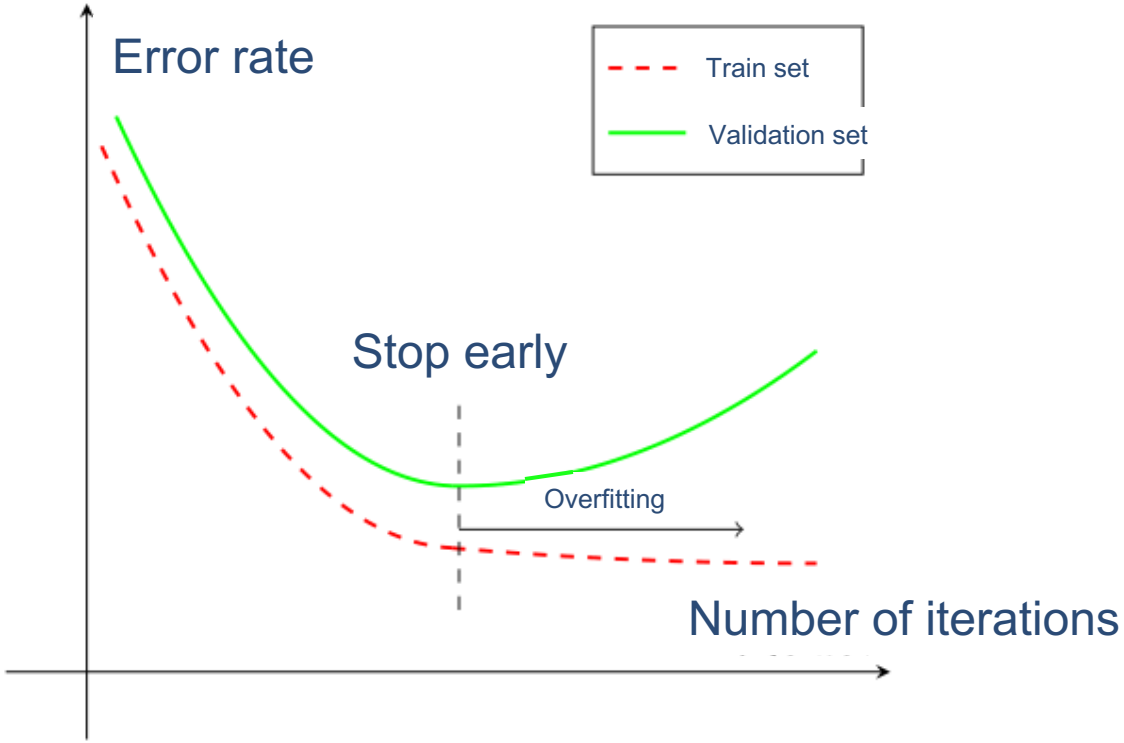
Define the model structure or optimization strategy





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. This policy is called stop early.



Stochastic Gradient Descent

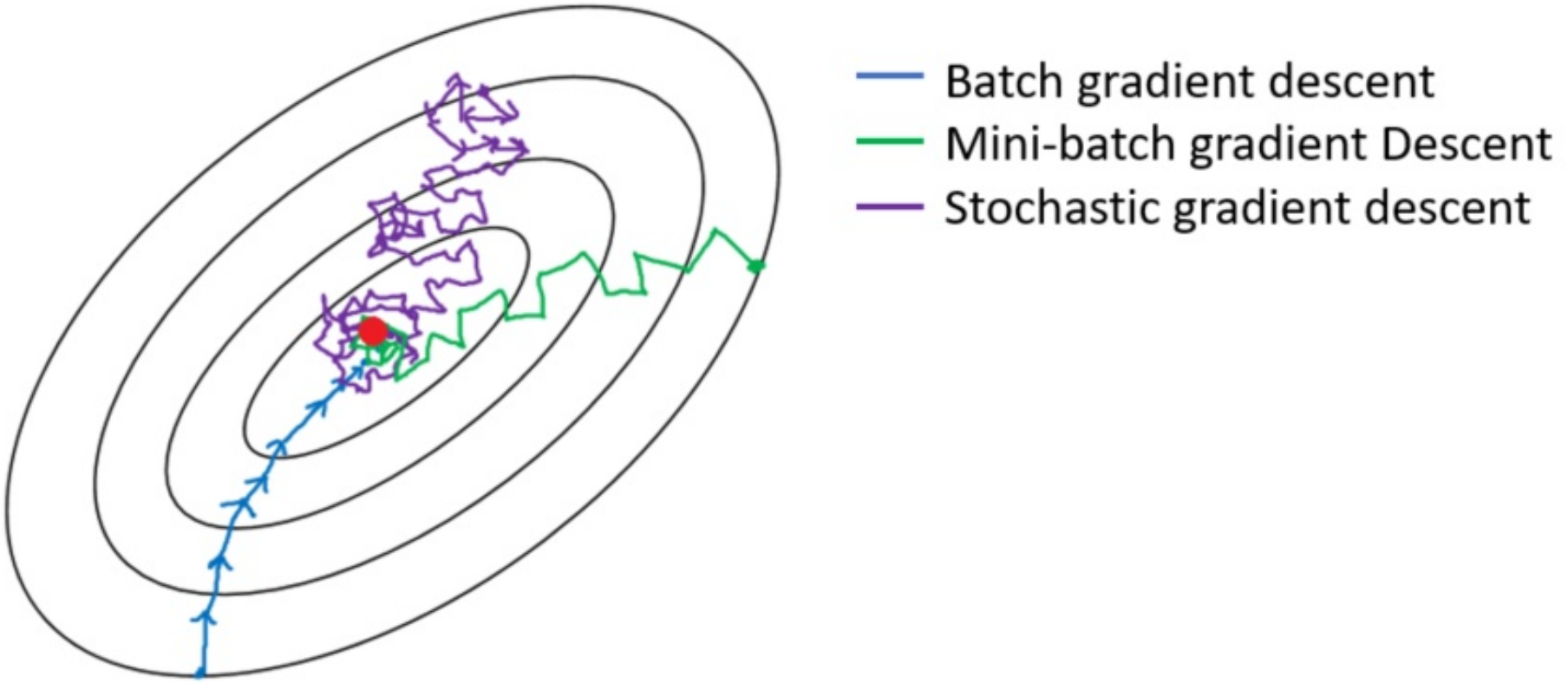
- Stochastic Gradient Descent (SGD) is also called incremental gradient descent, and each sample is updated based on only one sample

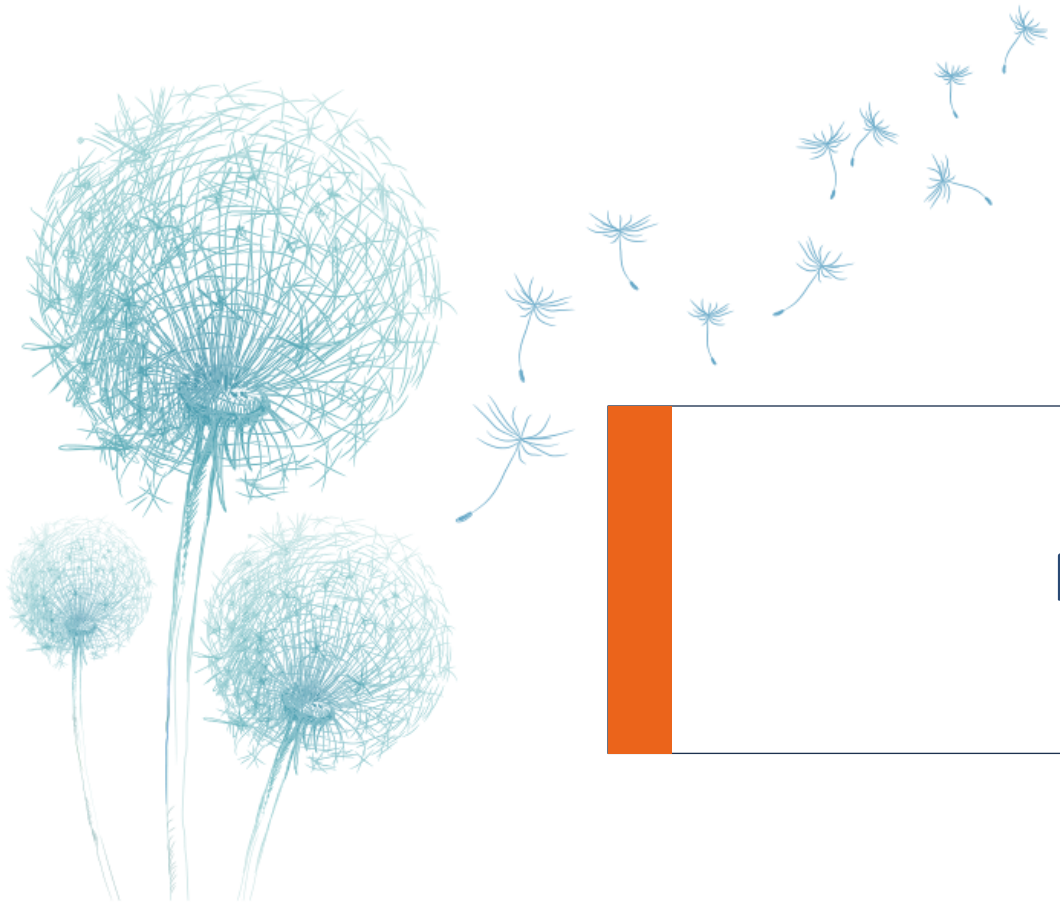
$$\theta_{t+1} = \theta_t - \alpha \frac{\partial \mathcal{L}\left(\theta_t; x^{(t)}, y^{(t)}\right)}{\partial \theta},$$

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

$$\theta_{t+1} \leftarrow \theta_t - \alpha \frac{1}{K} \sum_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{S}_t} \frac{\partial \mathcal{L}\left(\boldsymbol{y}, f(\boldsymbol{x};\theta)\right)}{\partial \theta}.$$

Stochastic Gradient Descent





Model evaluation

Generalization error

Expected risk

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

Empirical risk

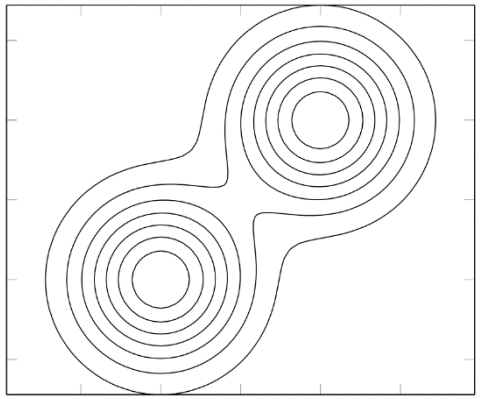
$$\mathcal{R}_{\mathcal{D}}^{emp}(\theta) = \frac{1}{N}\sum_{n=1}^{N}\mathcal{L}(y^{(n)}, f(x^{(n)}, \theta))$$

global

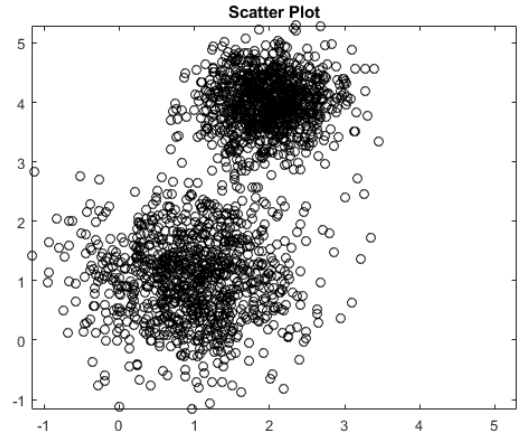
idealized

cannot be obtained

Real distribution



$\neq$



local

realistic

can be obtained

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

Generalization error: can measure whether a machine learning model can generalize well to unknown data. The goal of machine learning is to reduce generalization errors.

How to reduce generalization errors?

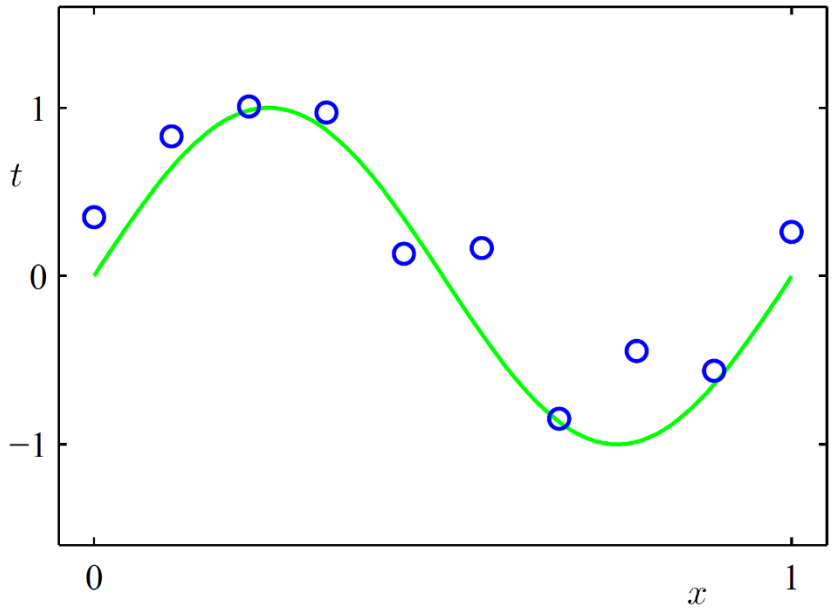
Optimization

Minimal empirical risk

Regularization

Reduce model complexity

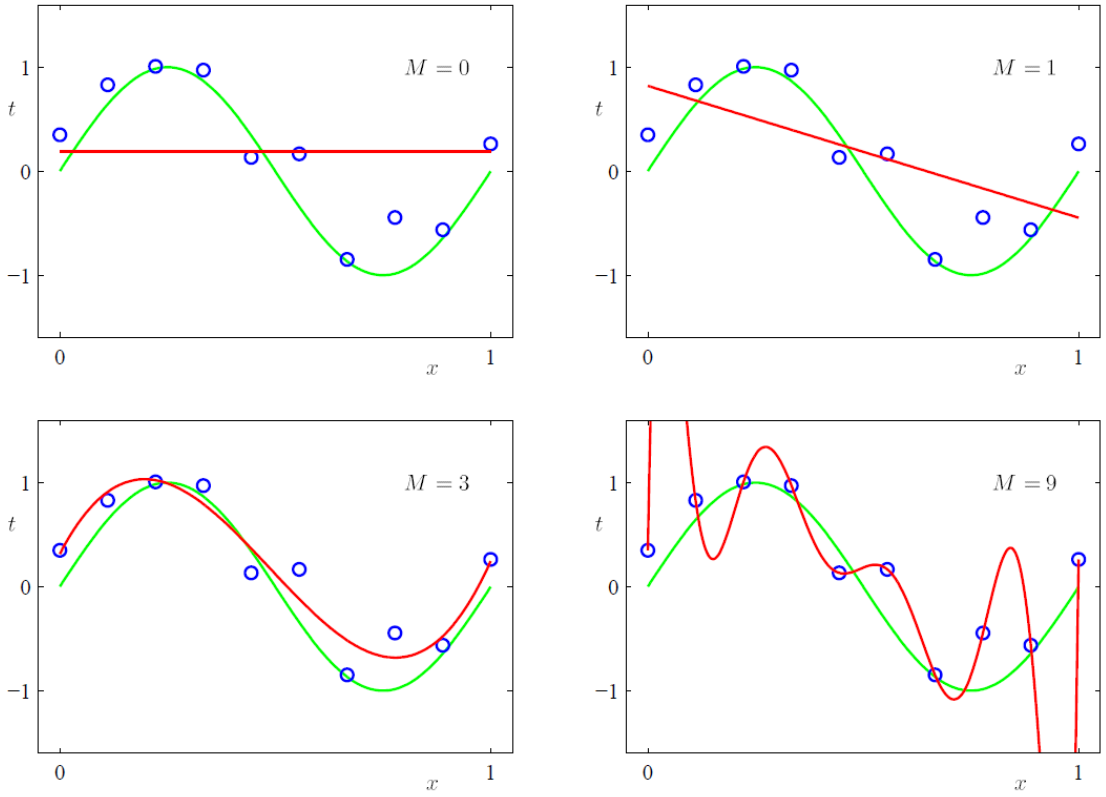
Polynomial Curve Fitting Example



Model $$y(x, \boldsymbol{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M$$

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

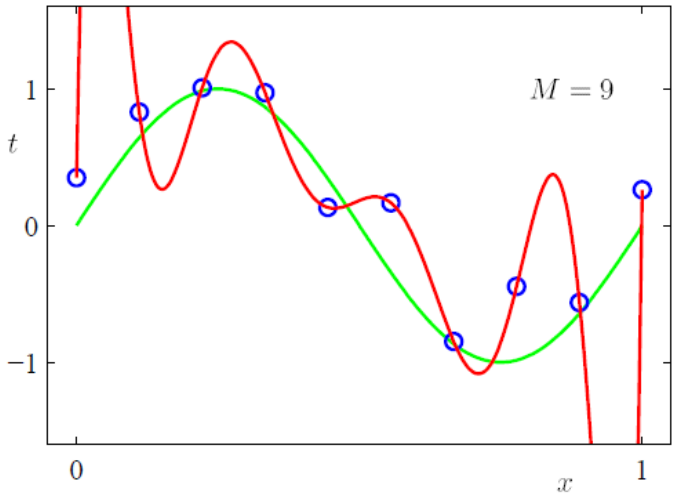
Which Degree of Polynomial?



A model selection problem

M = 9 → E(w) = 0: This is overfitting

Controlling Overfitting: Regularization



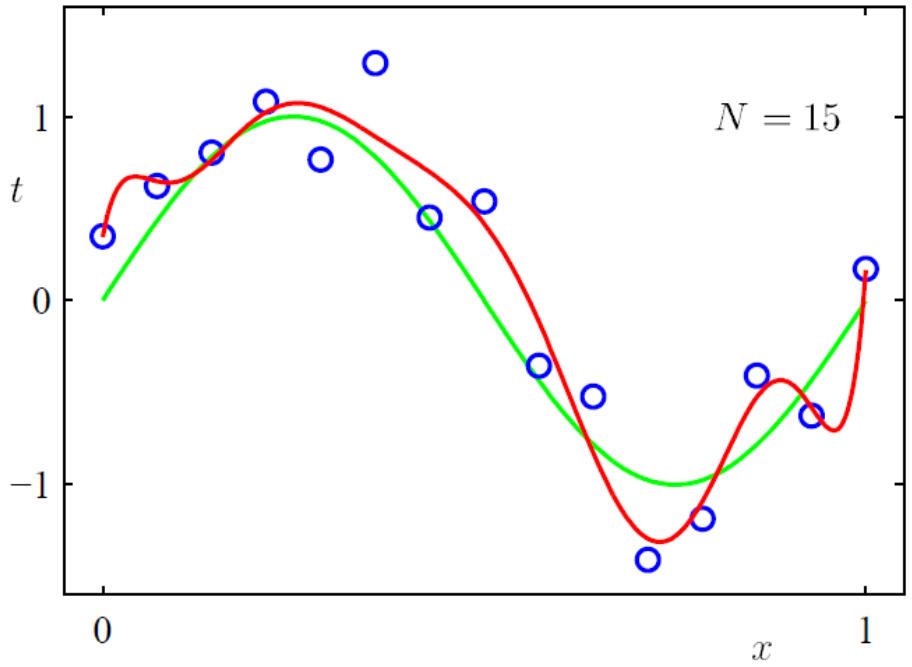
	$M = 0$	$M = 1$	$M = 3$	$M = 9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

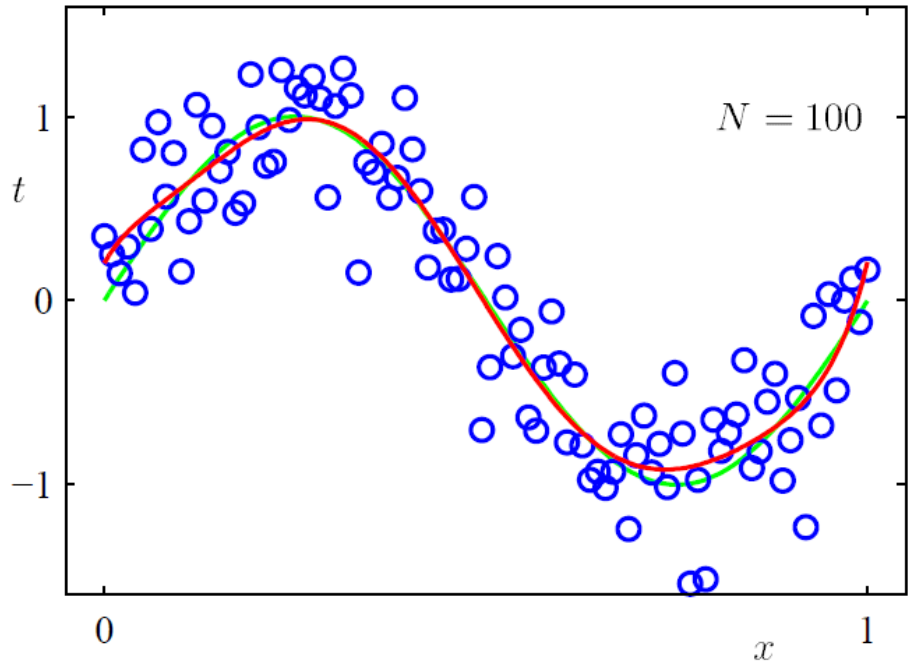
As order of polynomial M increases, so do coefficient magnitudes!

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

Penalize large coefficients

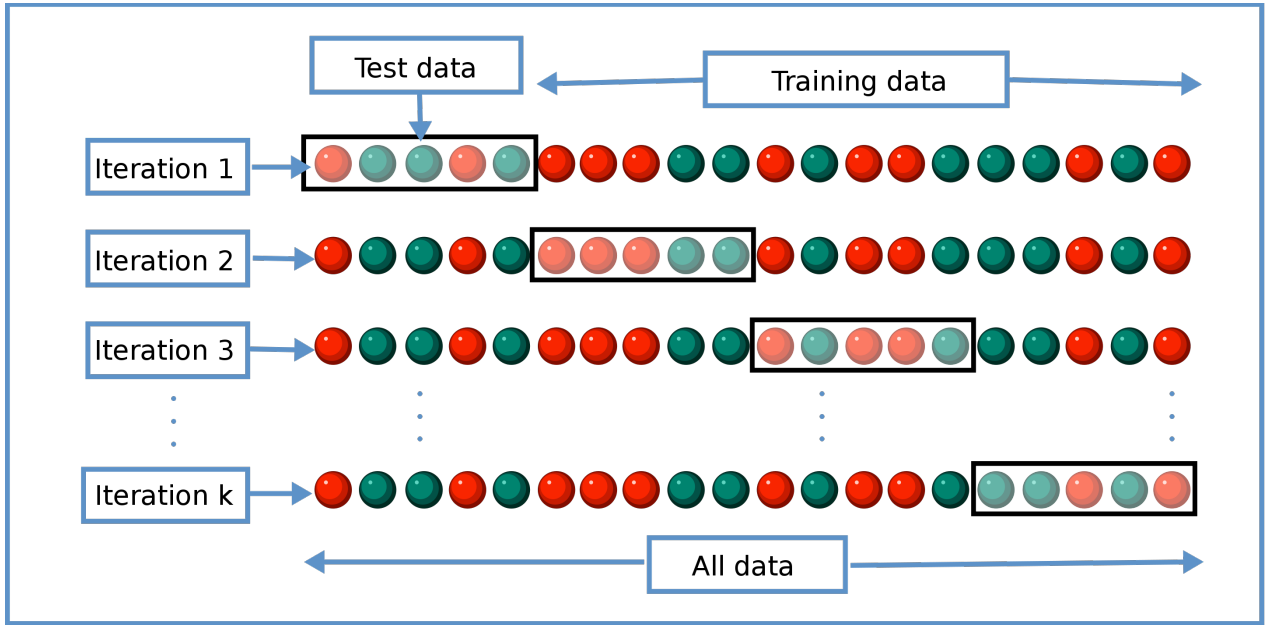
Controlling Overfitting: Dataset size





Cross Validation

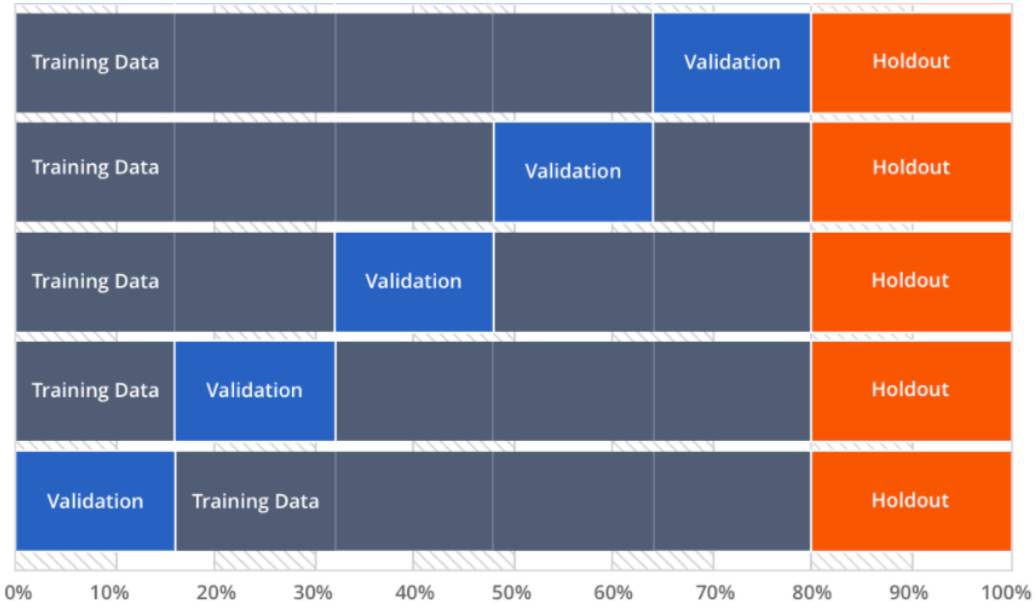
- Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample.
- Repeated use of data, split the sample data obtained, and combine them into different training sets and test sets



Why we need cross validation?

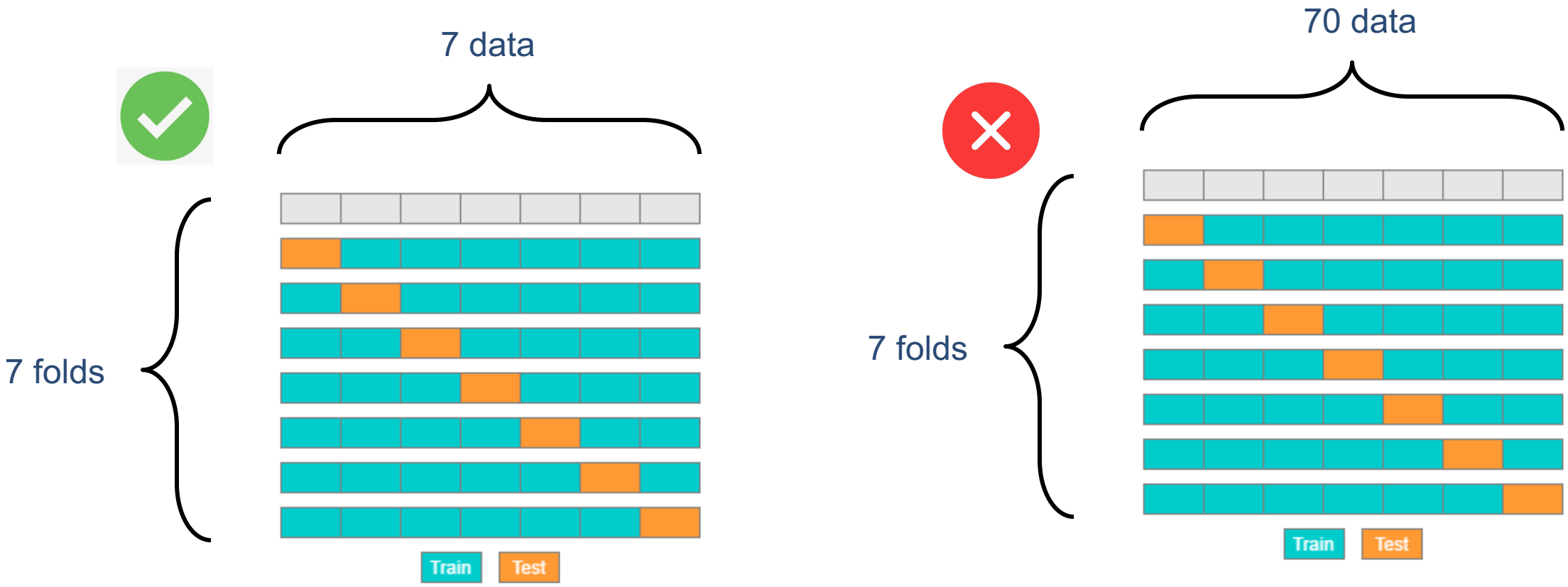
- Cross-validation is used when the amount of data is **not sufficient** (for example, the amount of data is less than 10,000), and can obtain as much effective information as possible from the limited data.

- 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.



Leave-one-out cross-validation (LOOCV)

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

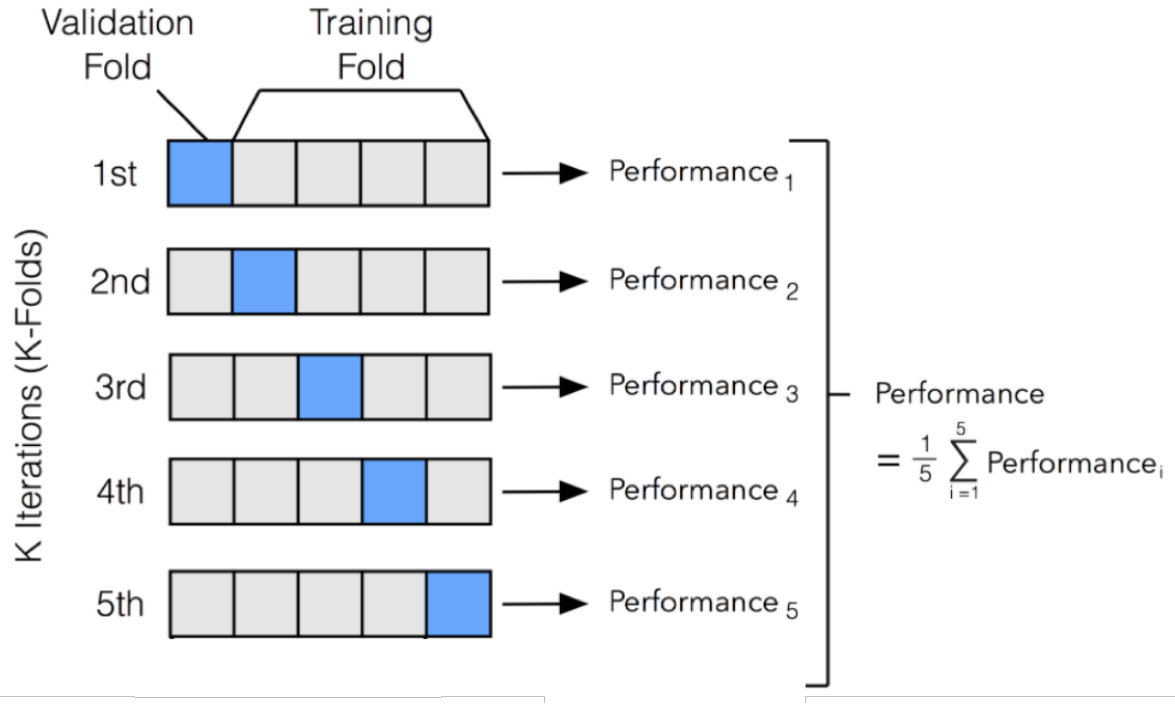


Leave-one-out cross-validation (LOOCV)

- 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 only suitable for small data sets.

k-fold Cross Validation

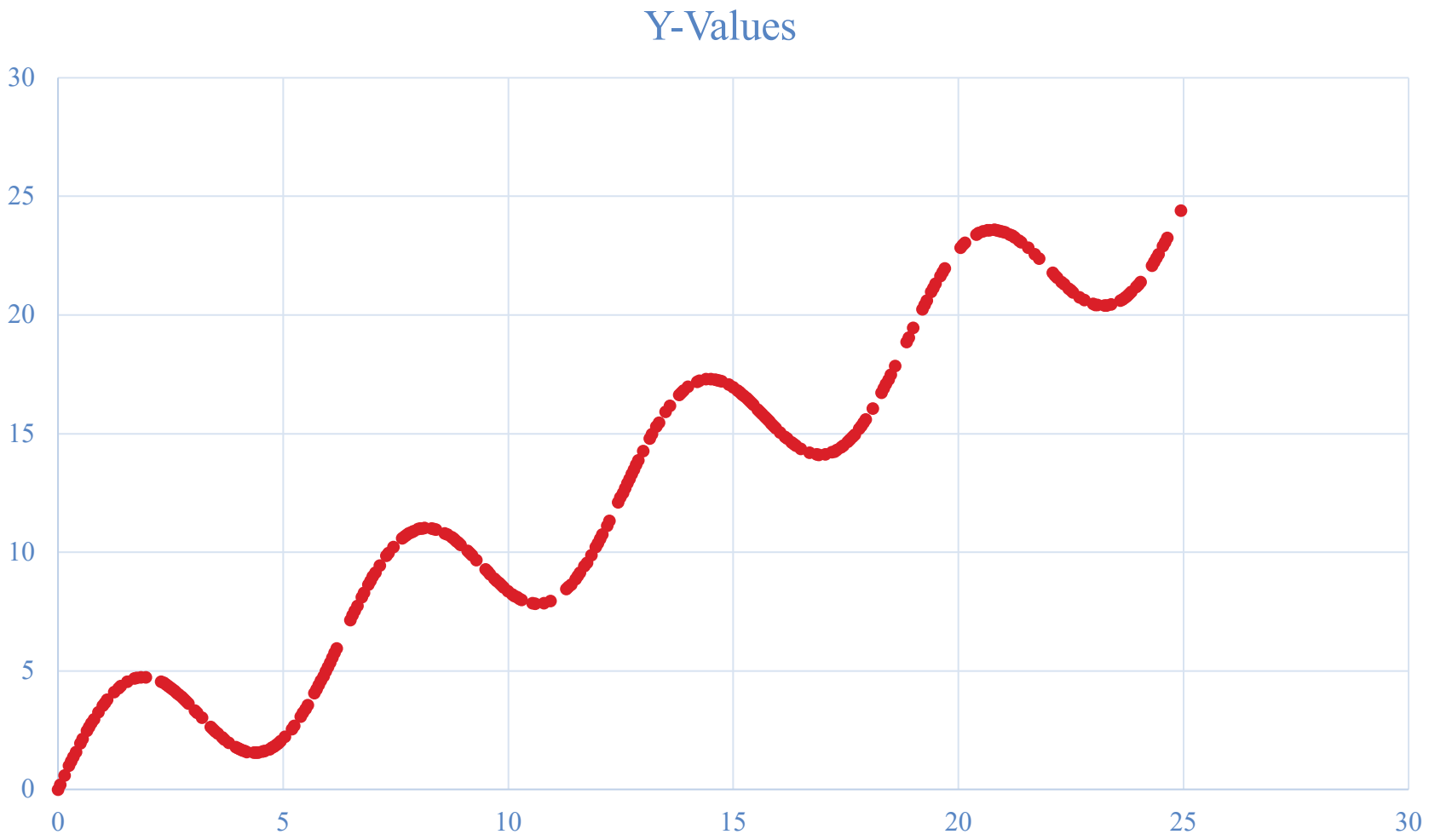
- The initial sample is divided into K sub-samples, one is used to verify the model, and k-1 are used for training
- 10-fold cross-validation is the most commonly used
- LOOCV is a special K-fold Cross Validation (K=N)



Train dataset

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

https://github.com/kevinsuo/CS7357/tree/5e79706addef78c30c262b65bc111a02382ad565/lec2



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--------------- """
```

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
```

```
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)
```

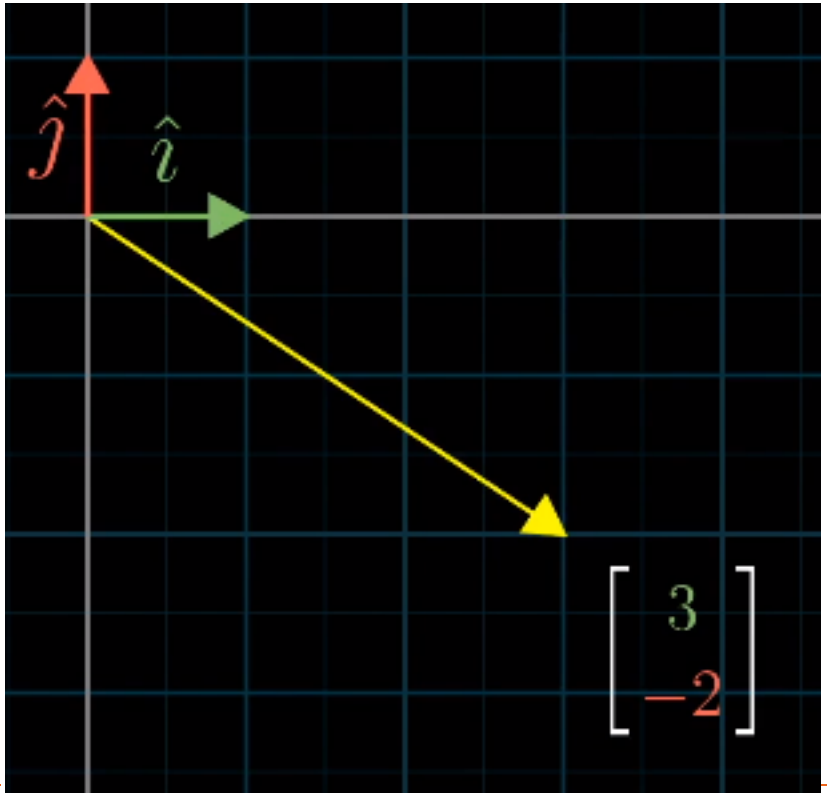
Basis function

```python
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
```

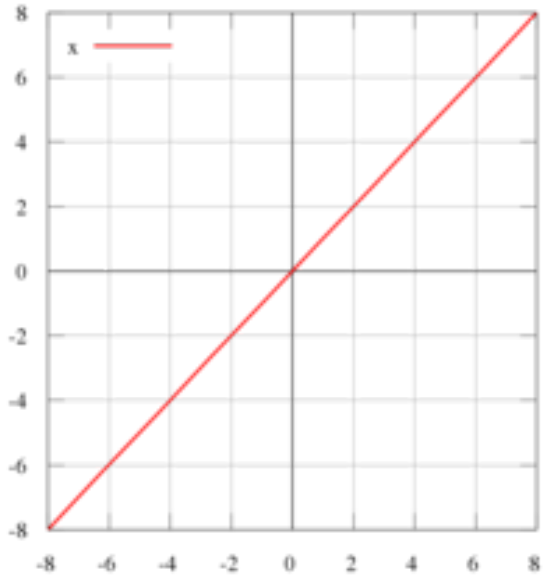
```python
"""-------------Train model--------------- """
def main(x_train, y_train):
    basis_func = gaussian_basis
    phi0 = np.expand_dims(np.ones_like(x_train), axis=1)
    phi1 = basis_func(x_train)
    phi = np.concatenate([phi0, phi1], axis=1)
    w = np.dot(np.linalg.pinv(phi), y_train)

    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

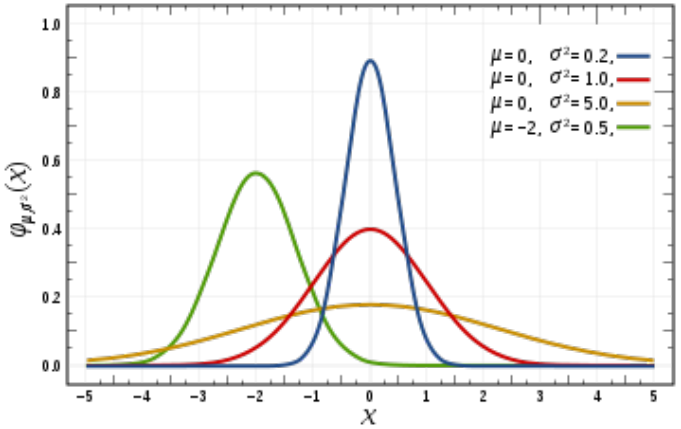


$f(x) = x$ for all elements x in M

Identity function

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

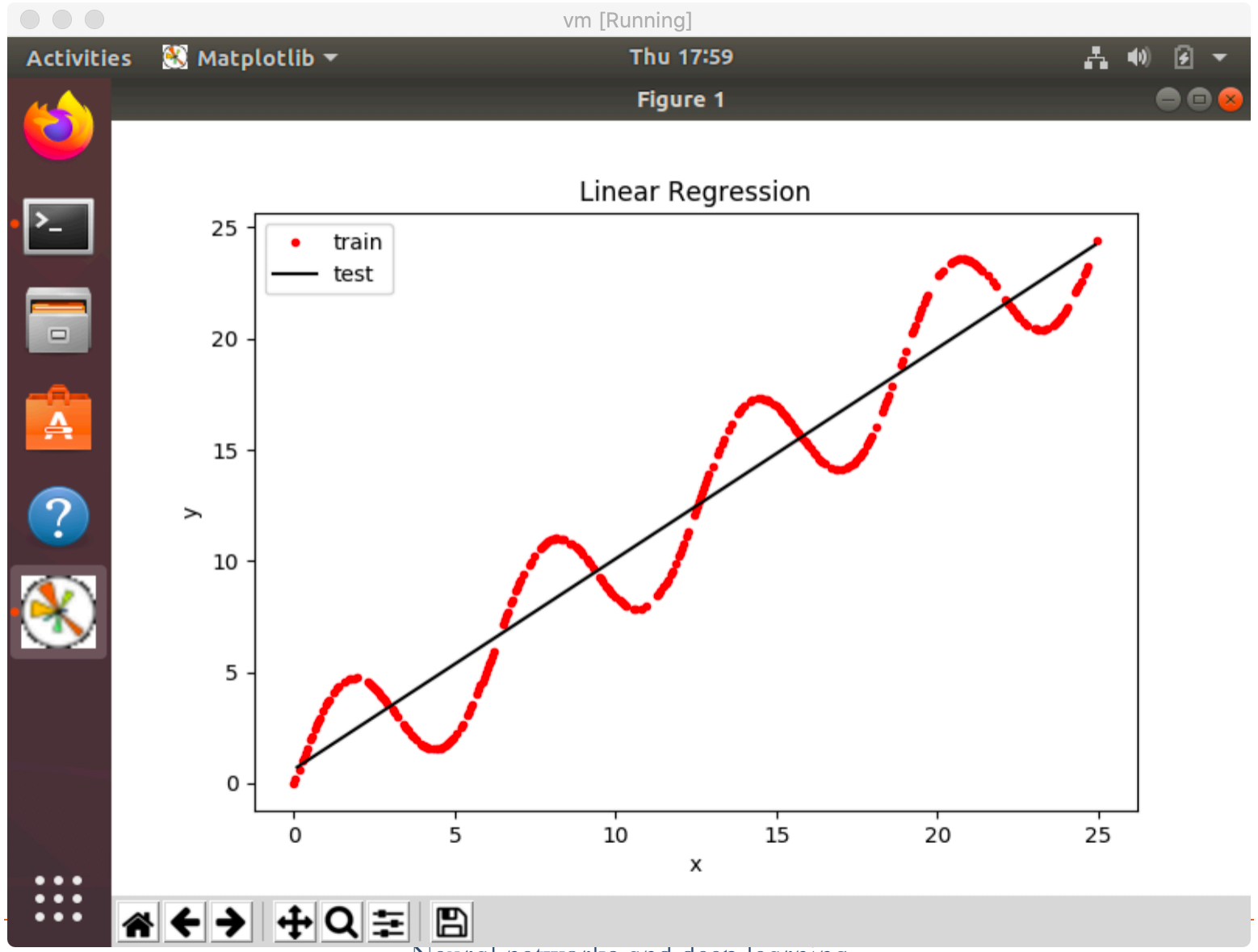
Polynomial function



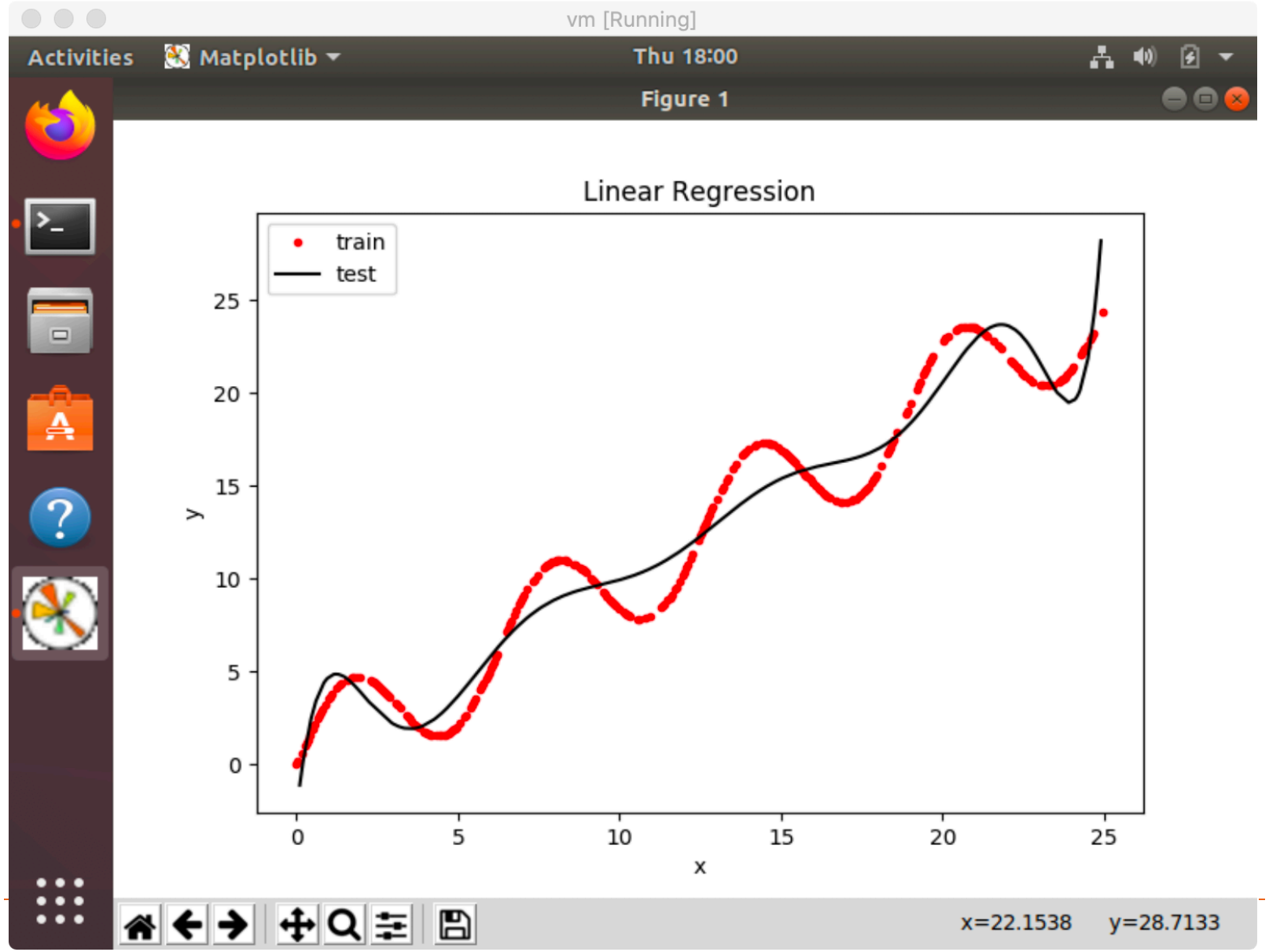
$$f(x) = a \cdot \exp\left(-\frac{(x-b)^2}{2c^2}\right)$$

Gaussian function

Execution: identity_basis



Execution: multinomial_basis



Execution: gaussian_basis