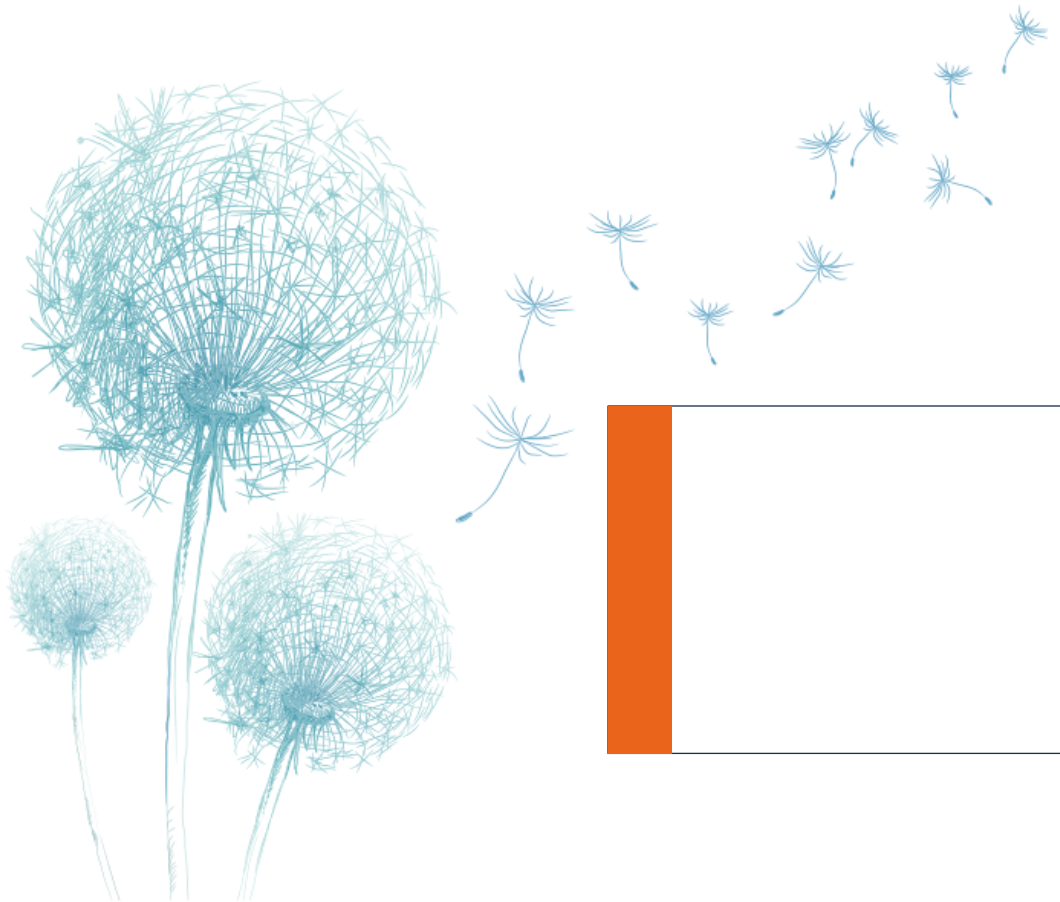
Neural networks and deep learning

Machine learning overview

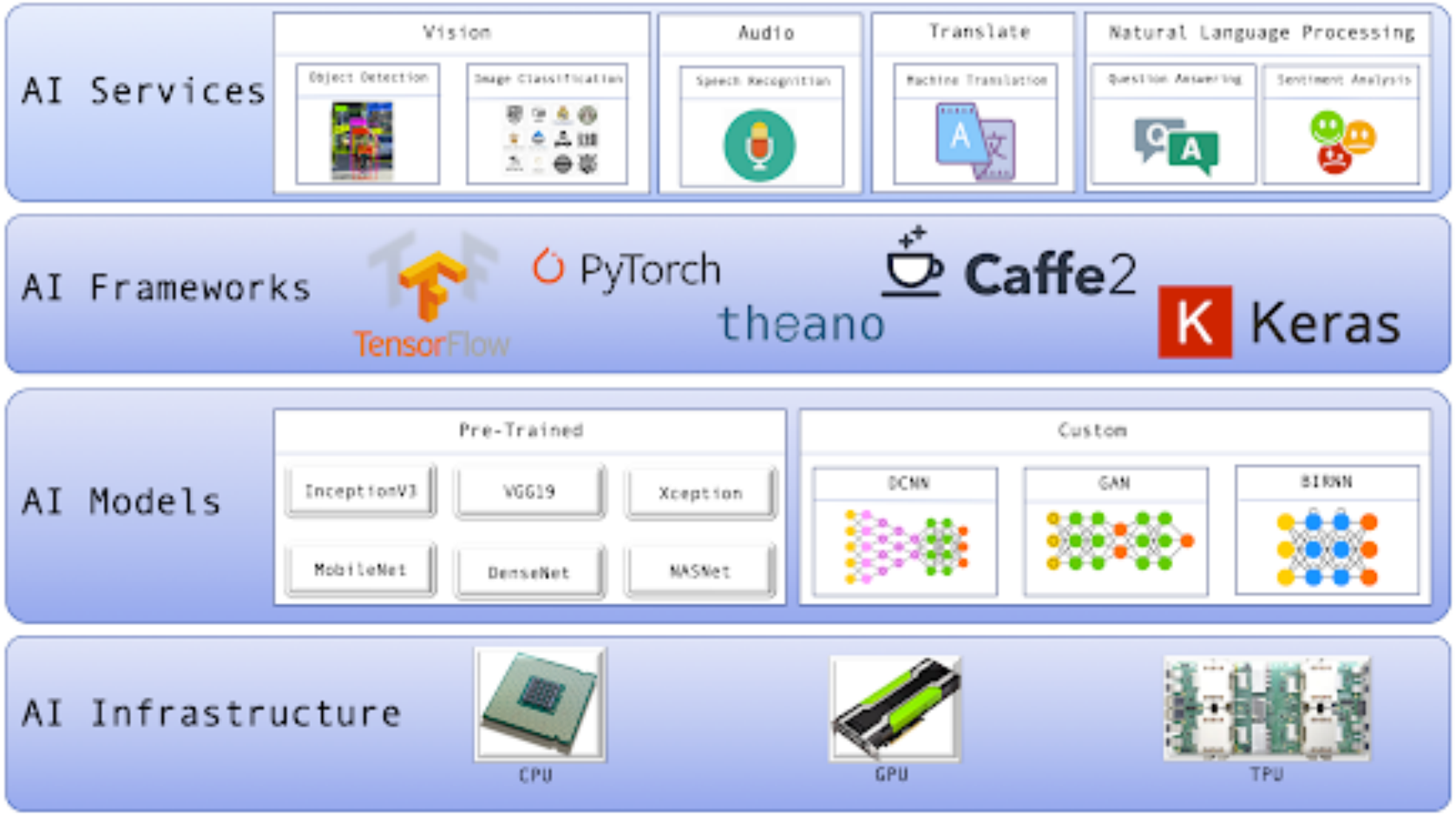
Kun Suo
Computer Science, Kennesaw State University
https://kevinsuo.github.io/



Hardware

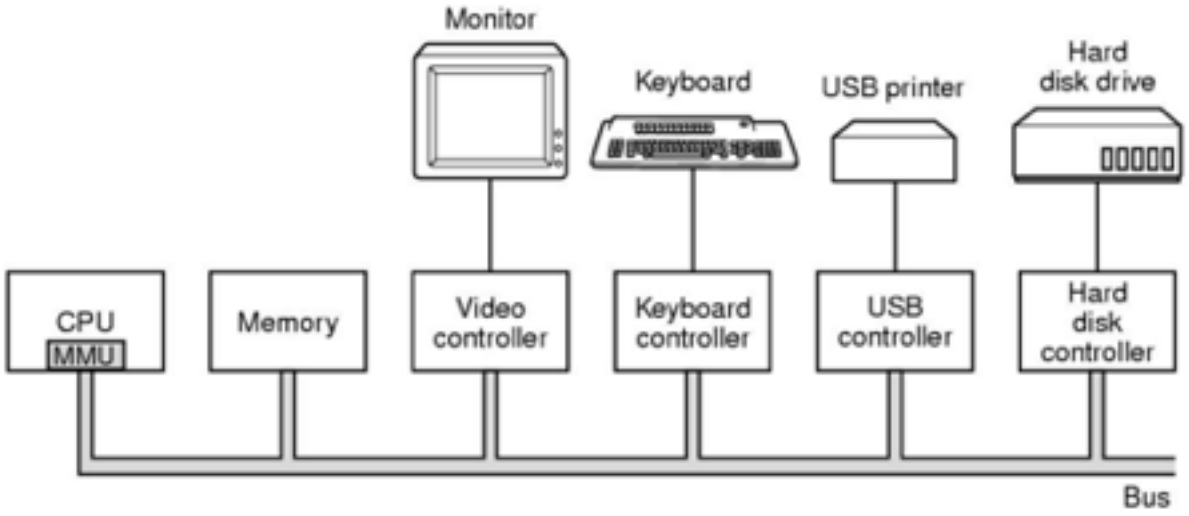
Computer Hardware Review

Artificial intelligence stack



Computer Hardware Review

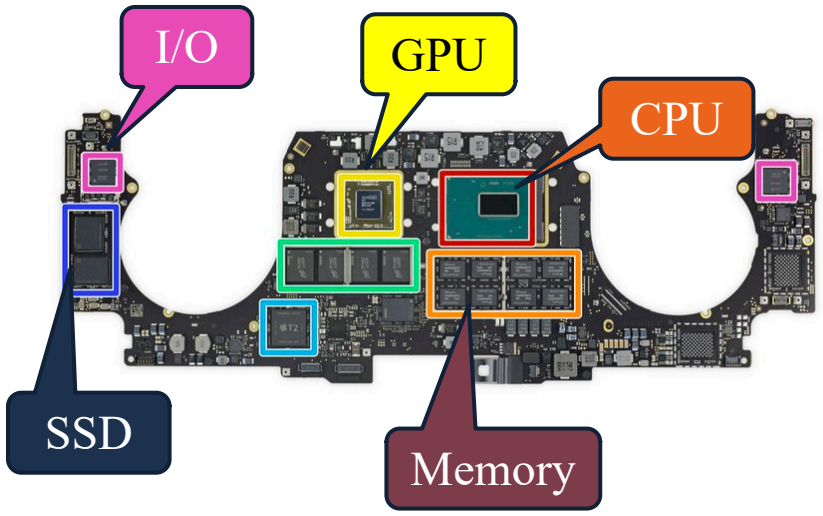
Basic components of a simple personal computer



- CPU: data processing
- Memory: volatile data storage
- Disk: persistent data storage
- NIC: inter-machine communication
- Bus: intra-machine communication

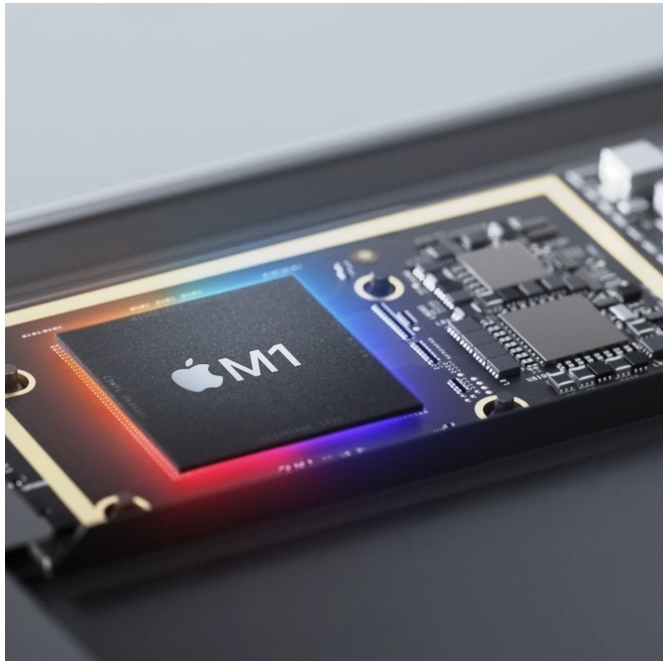
Computer Hardware Review

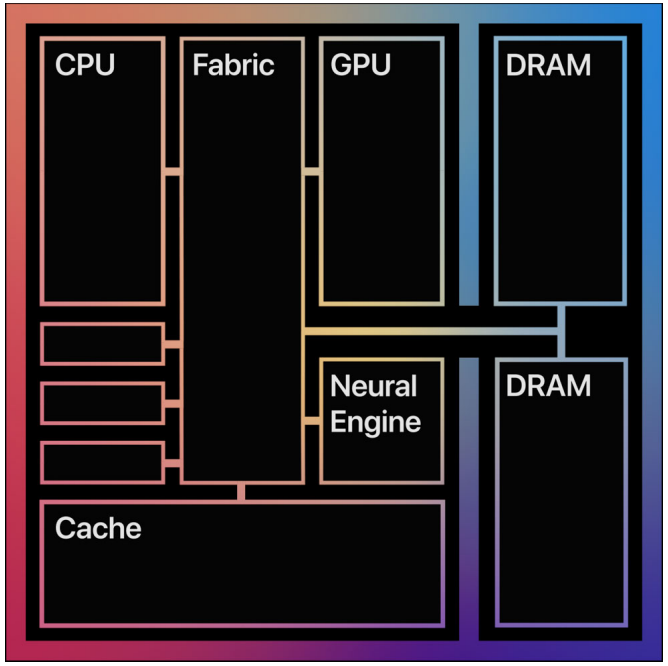
Computer Hardware Review

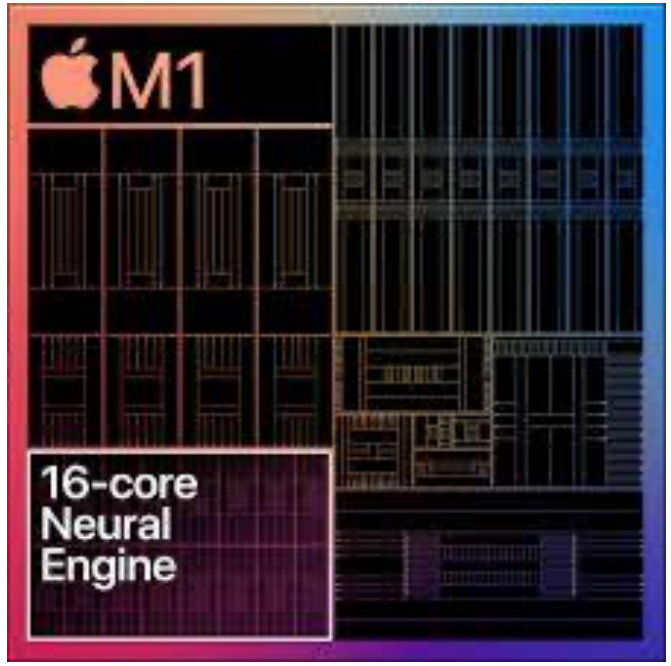


Computer Hardware Review

- Apple M1 chip

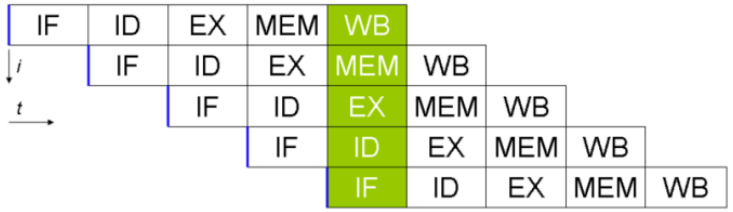


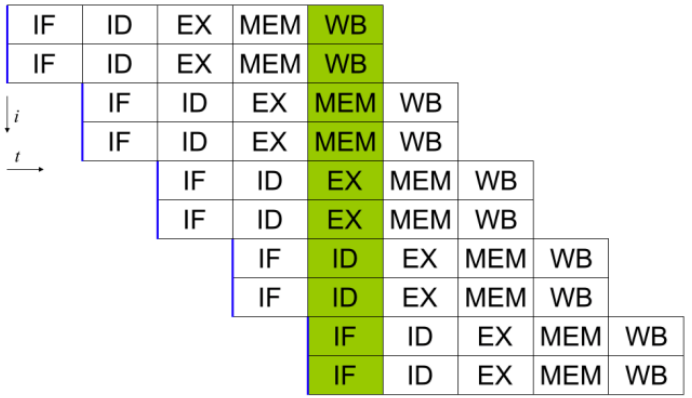


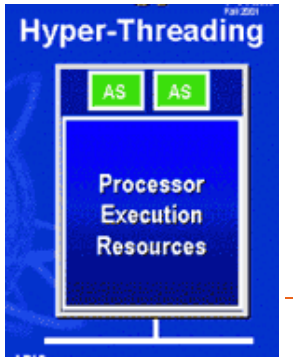


Central Processing Unit (CPU)

- Components
  - Arithmetic Logic Unit (ALU) -> Compute and data
  - Control Unit (CU) -> control device and system
- Clock rate
  - The speed at which a CPU is running
- Data storage
  - General-purpose registers: EAX, EBX …
  - Special-purpose registers: PC (program counter), SP (stack), IR (instruction register) …
- Parallelism
  - Instruction-level parallelism
  - Thread-level parallelism
    - Hyper-threading: duplicate units that store architectural states
    - Replicated: registers. Partitioned: ROB, load buffer…
      Shared: reservation station, caches

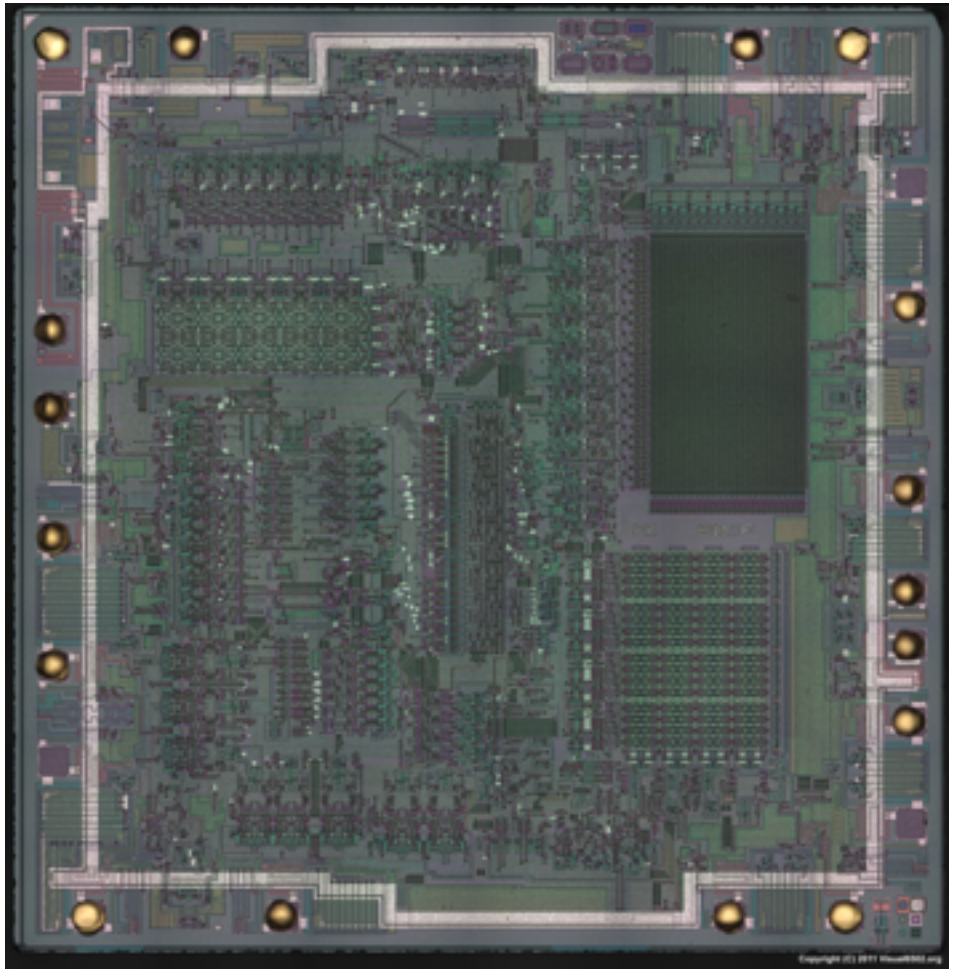






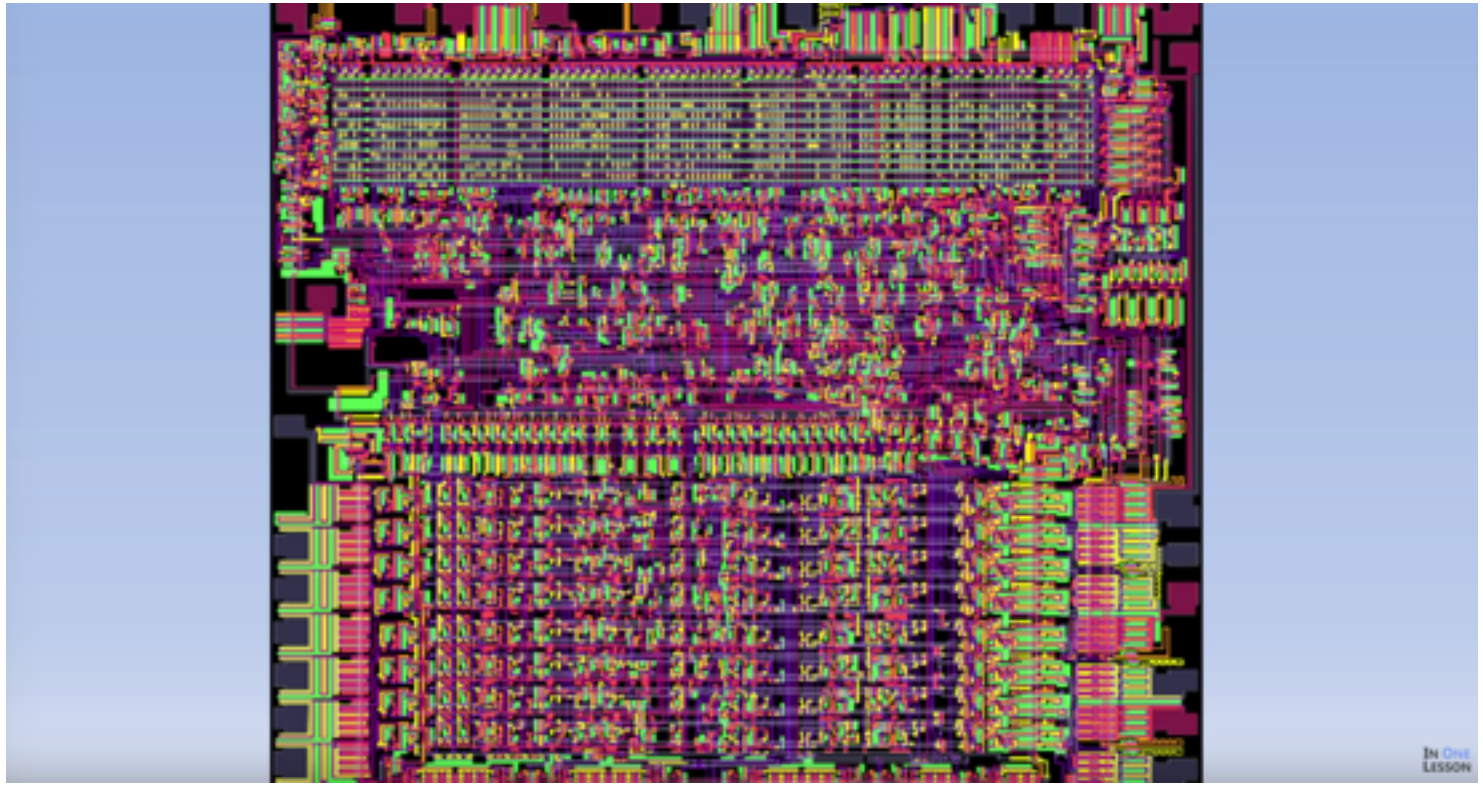
What's inside of CPU?

- http://www.visual6502.org/

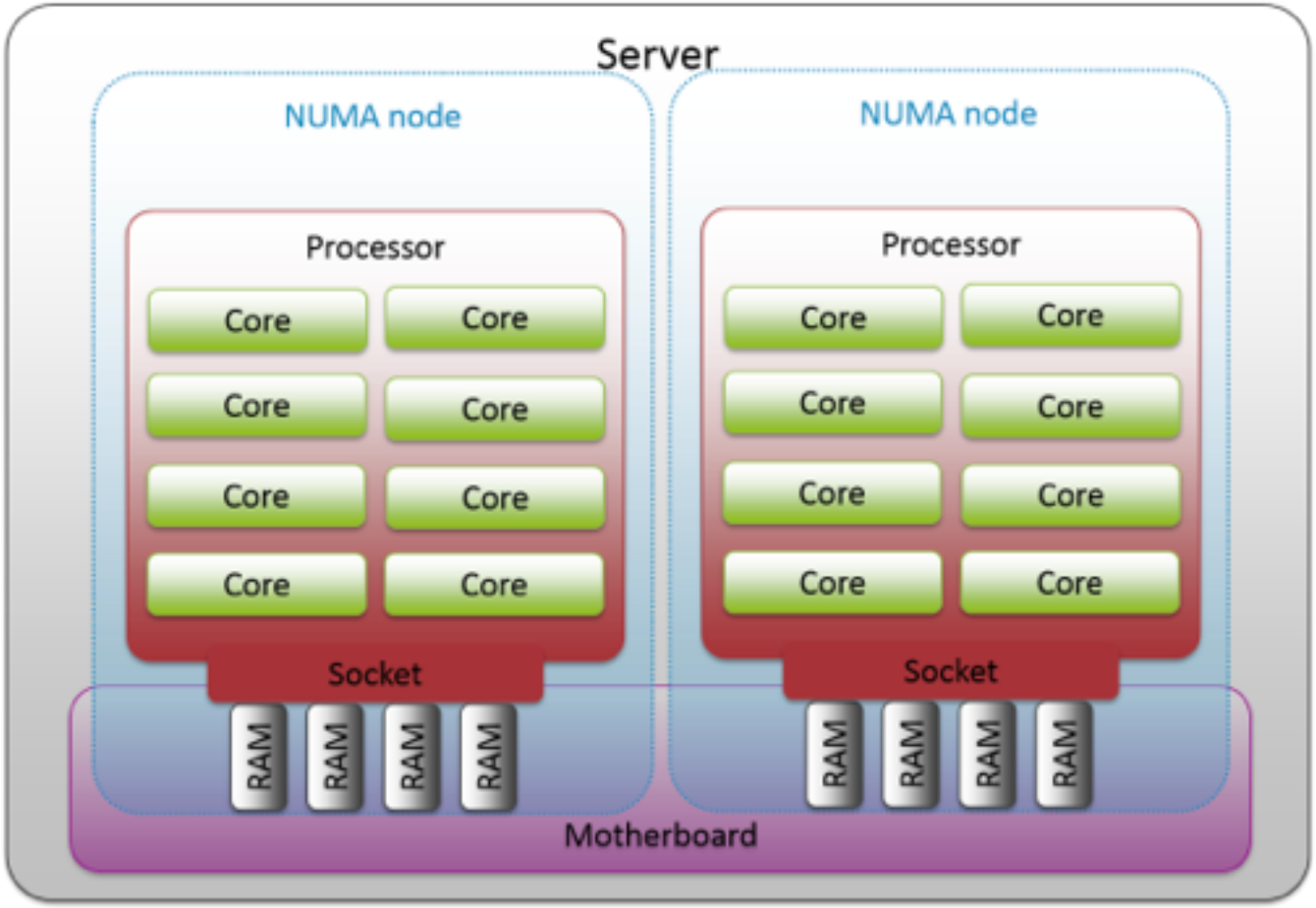


How CPU works?

- https://youtu.be/cNN_tTXABUA?t=494



NUMA node vs Socket vs Core relationship



CPU information

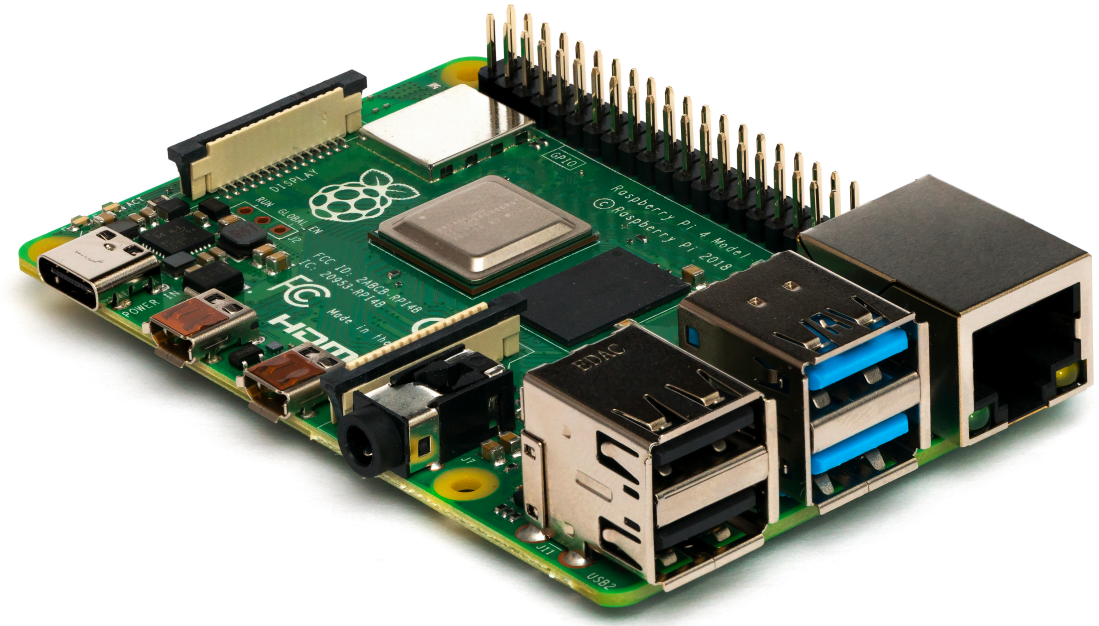
- lscpu

```
administrator@ubuntuvm-1604 ~> lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                2
On-line CPU(s) list:   0,1
Thread(s) per core:    1
Core(s) per socket:    1
Socket(s):             2
NUMA node(s):          1
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 79
Model name:            Intel(R) Xeon(R) CPU E5-2698 v4 @ 2.20GHz
Stepping:              1
CPU MHz:               2199.998
BogoMIPS:              4399.99
Hypervisor vendor:     VMware
Virtualization type:   full
L1d cache:             32K
L1i cache:             32K
L2 cache:              256K
L3 cache:              51200K
NUMA node0 CPU(s):     0,1
Flags:                 fpu vme de pse tsc msr pae mce cx8 apic sep m
all nx pdpe1gb rdtscp lm constant_tsc arch_perfmon nopl xtopology ts
id sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave av
lt invpcid_single pti ssbd ibrs ibpb stibp fsgsbase tsc_adjust bmi1
 flush_l1d arch_capabilities
```

CPU information

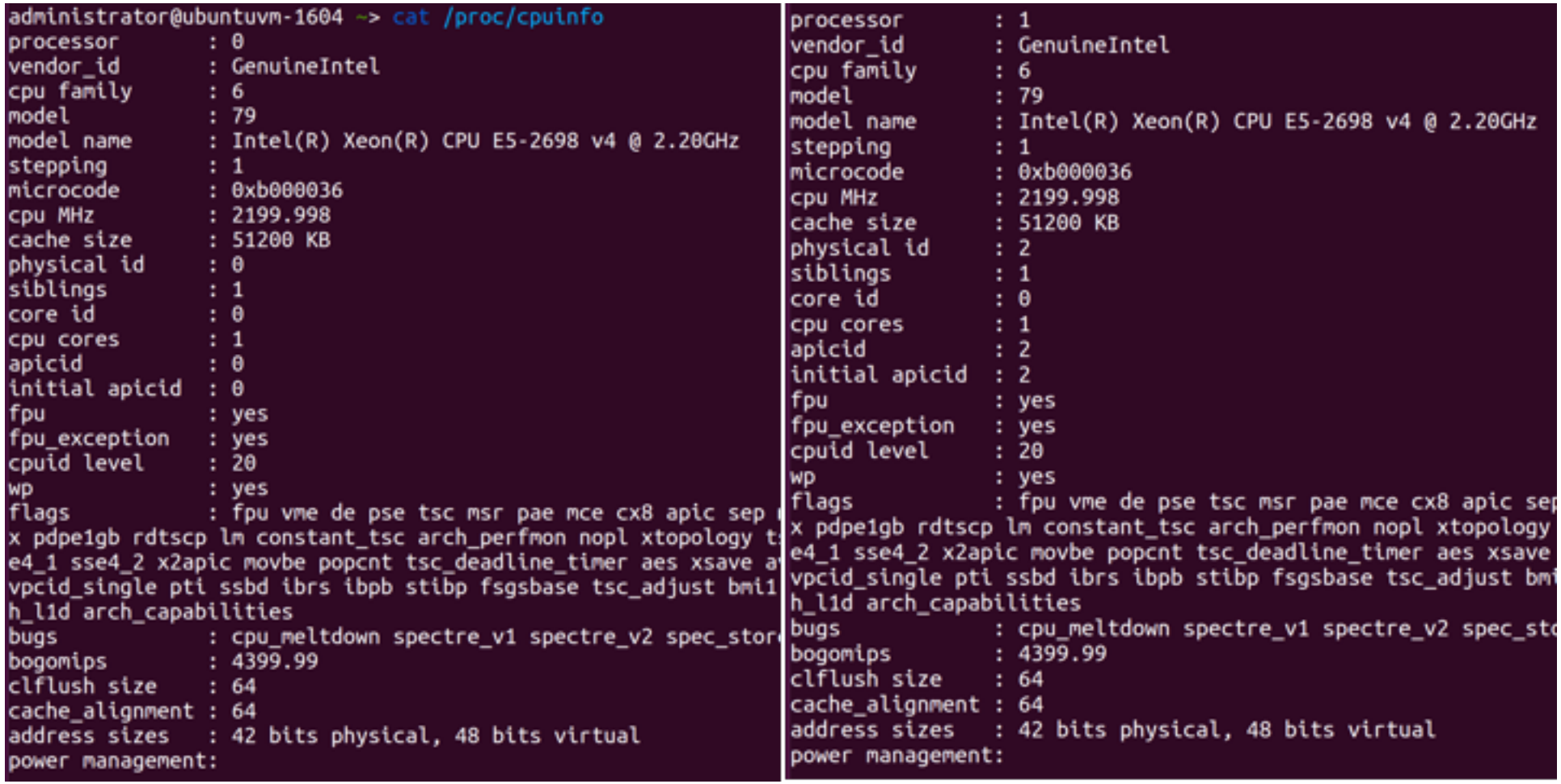
- lscpu

```
pi@raspberrypi:~ $ lscpu
Architecture:            armv7l
Byte Order:              Little Endian
CPU(s):                  4
On-line CPU(s) list:     0-3
Thread(s) per core:      1
Core(s) per socket:      4
Socket(s):               1
Vendor ID:               ARM
Model:                   3
Model name:              Cortex-A72
Stepping:                r0p3
CPU max MHz:             1500.0000
CPU min MHz:             600.0000
BogoMIPS:                108.00
Flags:                   half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva idivt vfpd32 lpae evtstrm crc32
```



CPU information

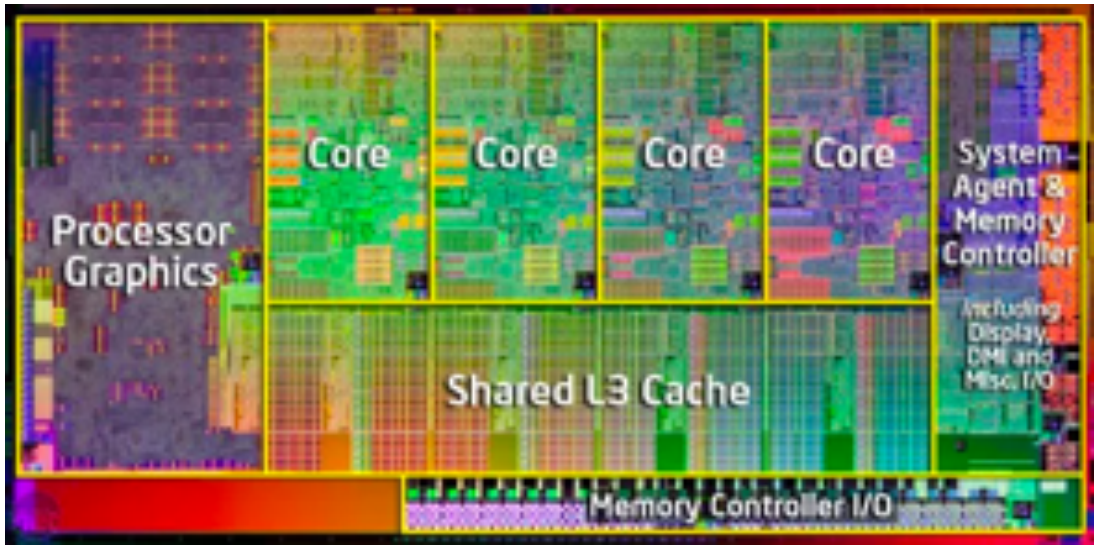
- $ cat /proc/cpuinfo

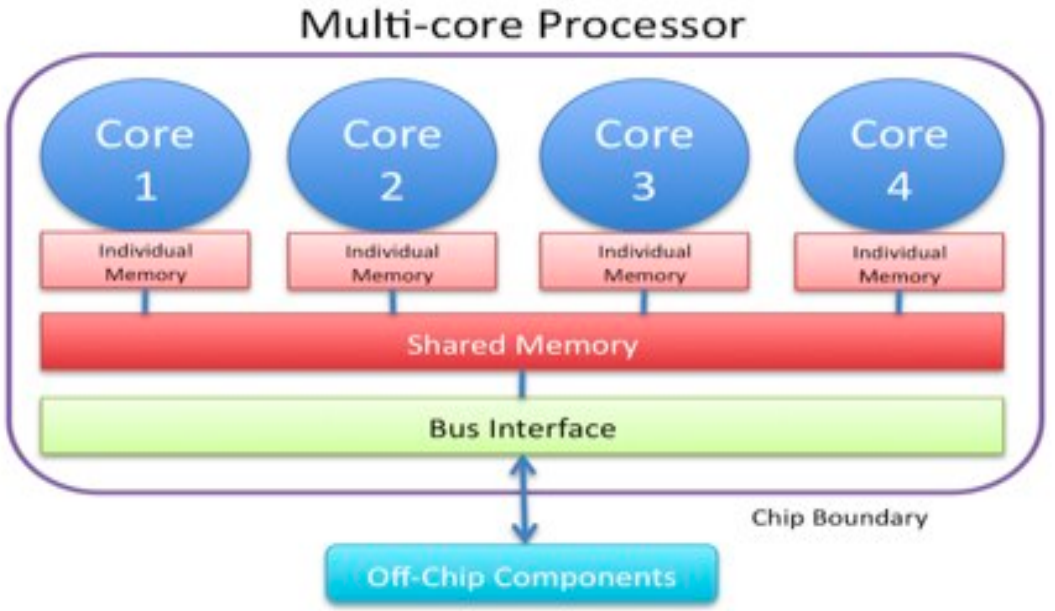


```
administrator@ubuntuvm-1604 ~> cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 79
model name      : Intel(R) Xeon(R) CPU E5-2698 v4 @ 2.20GHz
stepping        : 1
microcode       : 0xb000036
cpu MHz         : 2199.998
cache size      : 51200 KB
physical id     : 0
siblings        : 1
core id         : 0
cpu cores       : 1
apicid          : 0
initial apicid  : 0
fpu             : yes
fpu_exception   : yes
cpuid level     : 20
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep
x pdpe1gb rdtscp lm constant_tsc arch_perfmon nopl xtopology t
e4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave a
vpcid_single pti ssbd ibrs ibpb stibp fsgsbase tsc_adjust bmi1
h_l1d arch_capabilities
bugs            : cpu_meltdown spectre_v1 spectre_v2 spec_stor
bogomips        : 4399.99
clflush size    : 64
cache_alignment : 64
address sizes   : 42 bits physical, 48 bits virtual
power management:

processor       : 1
vendor_id       : GenuineIntel
cpu family      : 6
model           : 79
model name      : Intel(R) Xeon(R) CPU E5-2698 v4 @ 2.20GHz
stepping        : 1
microcode       : 0xb000036
cpu MHz         : 2199.998
cache size      : 51200 KB
physical id     : 2
siblings        : 1
core id         : 0
cpu cores       : 1
apicid          : 2
initial apicid  : 2
fpu             : yes
fpu_exception   : yes
cpuid level     : 20
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep
x pdpe1gb rdtscp lm constant_tsc arch_perfmon nopl xtopology
e4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave
vpcid_single pti ssbd ibrs ibpb stibp fsgsbase tsc_adjust bmi
h_l1d arch_capabilities
bugs            : cpu_meltdown spectre_v1 spectre_v2 spec_st
bogomips        : 4399.99
clflush size    : 64
cache_alignment : 64
address sizes   : 42 bits physical, 48 bits virtual
power management:
```

Multi-Core Processors (SMP)

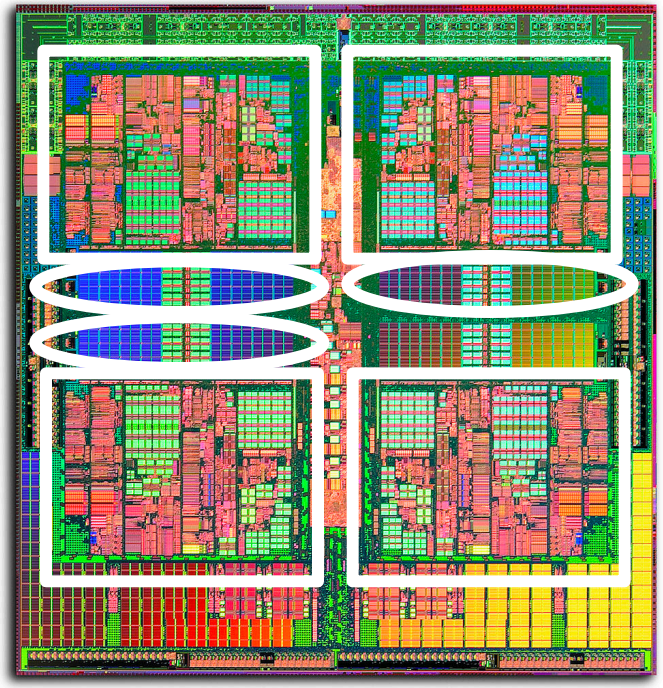
- Multiple CPUs on a single chip

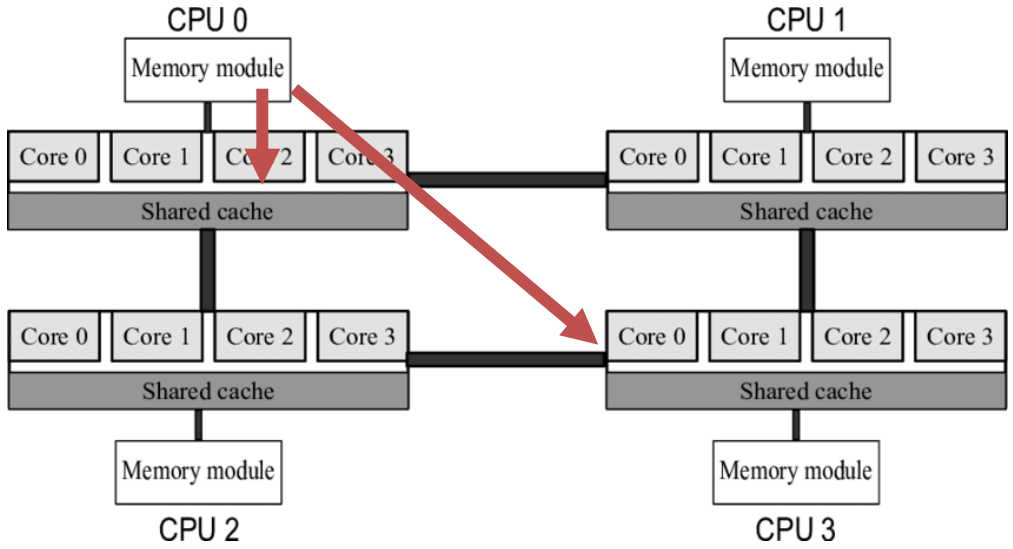




Multi-Core Processors (NUMA)

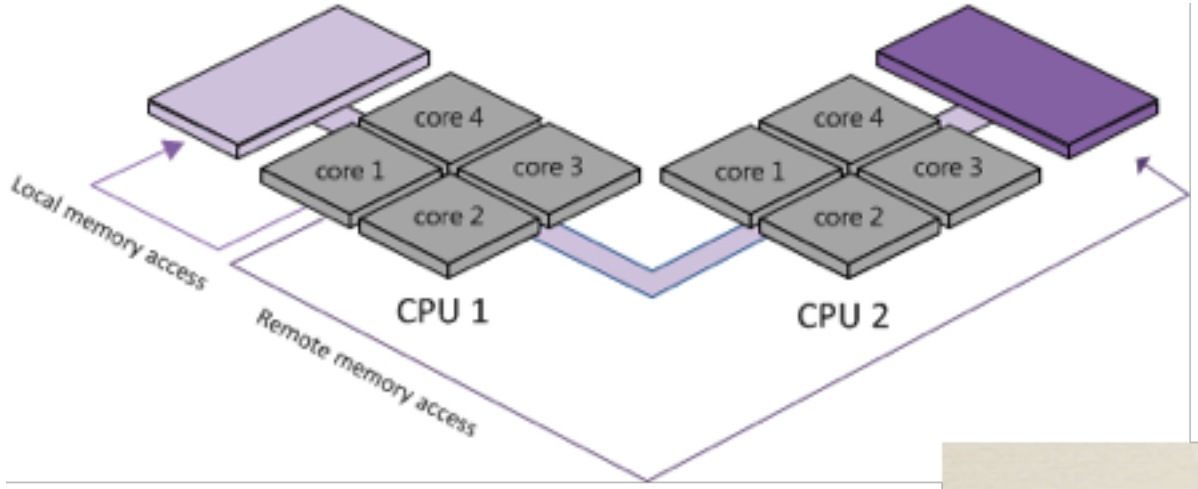
- Multiple CPUs on a single chip

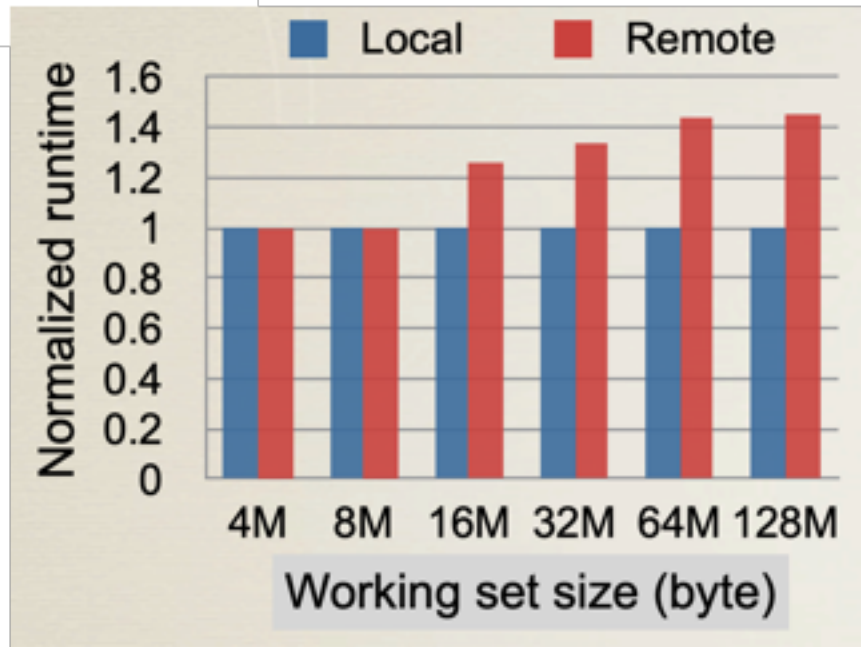




Non-uniform memory access (**NUMA**)

Multi-Core Processors (NUMA)





Check CPU topology

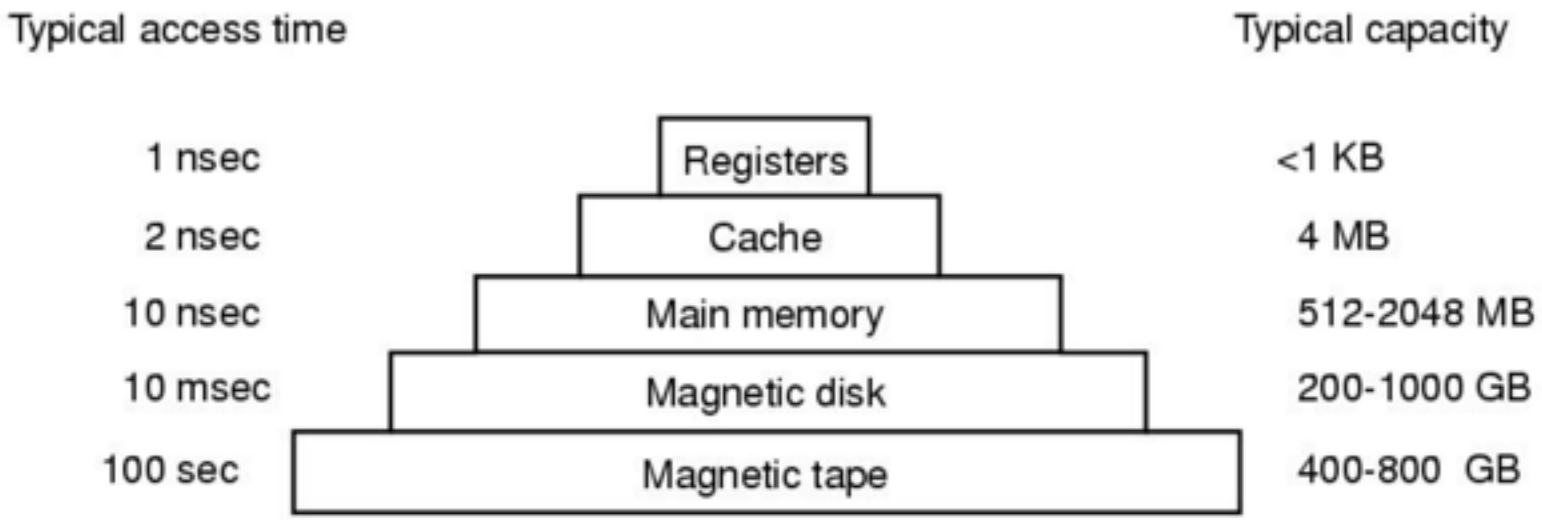
- $ likwid-topology –g

- NUMA example: https://github.com/RRZE-HPC/likwid/wiki/TutorialNUMA

```
ksuo@ksuo-VirtualBox ~/likwid-5.0.0> likwid-topology -g
--------------------------------------------------------------------------------
CPU name:       Intel(R) Core(TM) i9-9880H CPU @ 2.30GHz
CPU type:       Intel Coffeelake processor
CPU stepping:   13
********************************************************************************
Hardware Thread Topology
********************************************************************************
Sockets:                1
Cores per socket:       4
Threads per core:       1
--------------------------------------------------------------------------------
HWThread        Thread          Core            Socket          Available
0               0               0               0               *
1               0               1               0               *
2               0               2               0               *
3               0               3               0               *
--------------------------------------------------------------------------------
Socket 0:               ( 0 1 2 3 )
--------------------------------------------------------------------------------
********************************************************************************
Graphical Topology
********************************************************************************
Socket 0:
+-------------------------------------------+
| +---------+ +---------+ +---------+ +---------+ |
| |    0    | |    1    | |    2    | |    3    | |
| +---------+ +---------+ +---------+ +---------+ |
| +---------+ +---------+ +---------+ +---------+ |
| |  32 kB  | |  32 kB  | |  32 kB  | |  32 kB  | |
| +---------+ +---------+ +---------+ +---------+ |
| +---------+ +---------+ +---------+ +---------+ |
| | 256 kB  | | 256 kB  | | 256 kB  | | 256 kB  | |
| +---------+ +---------+ +---------+ +---------+ |
| +---------+ +---------+ +---------+ +---------+ |
| |  16 MB  | |  16 MB  | |  16 MB  | |  16 MB  | |
| +---------+ +---------+ +---------+ +---------+ |
+-------------------------------------------+
```

Memory

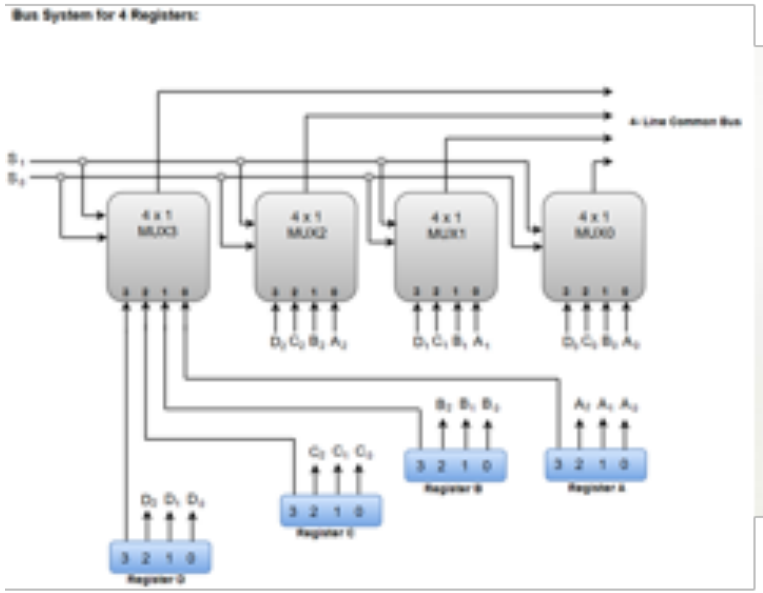
- A typical memory hierarchy

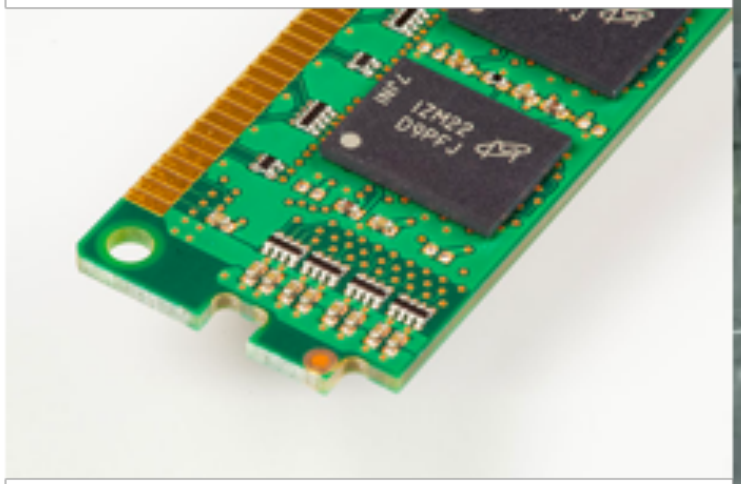


Minimize the access time vs. Cost

Memory

- A typical memory hierarchy





Memory information

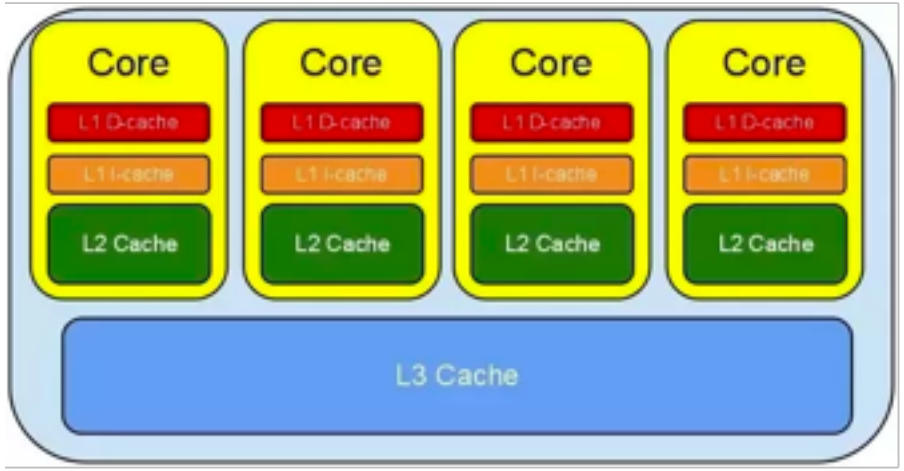
- free

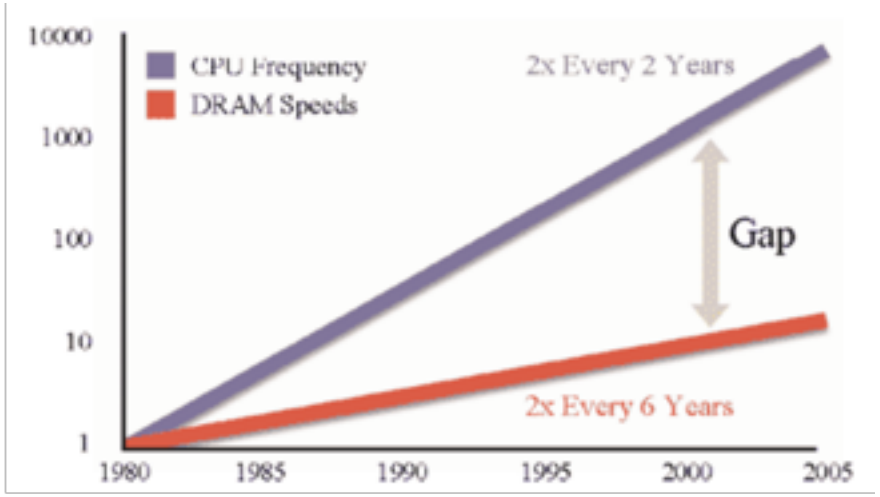
```
administrator@ubuntuvm-1604 ~> free
              total        used        free      shared  buff/cache   available
Mem:        8168756     1348160     3031888       97424     3788708     6383036
Swap:        998396           0      998396
```

- The free command displays:
  - Total amount of free and used physical memory
  - Total amount of swap memory in the system
  - Buffers and caches used by the kernel

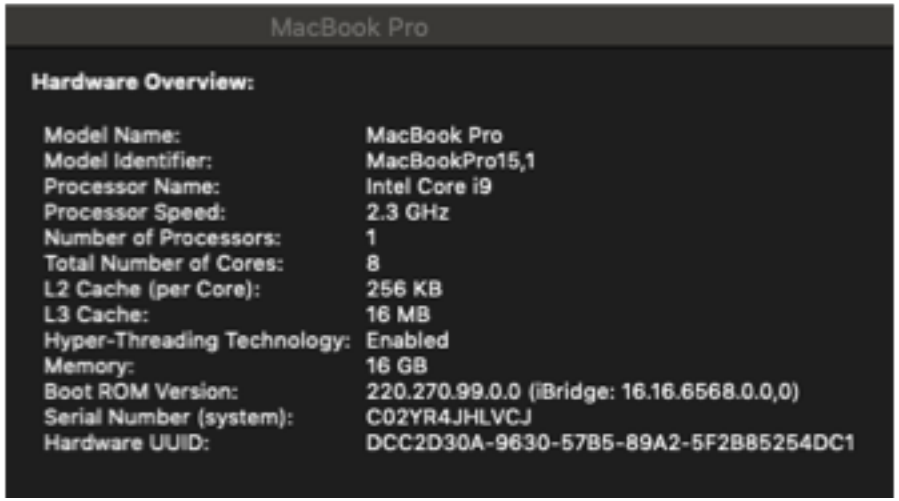
Cache

- Why Cache is important?





A larger size than registers
A much faster speed than memory
Tradeoff between performance and cost



Check Cache info

- $ likwid-topology –g
- $ lscpu | grep cache

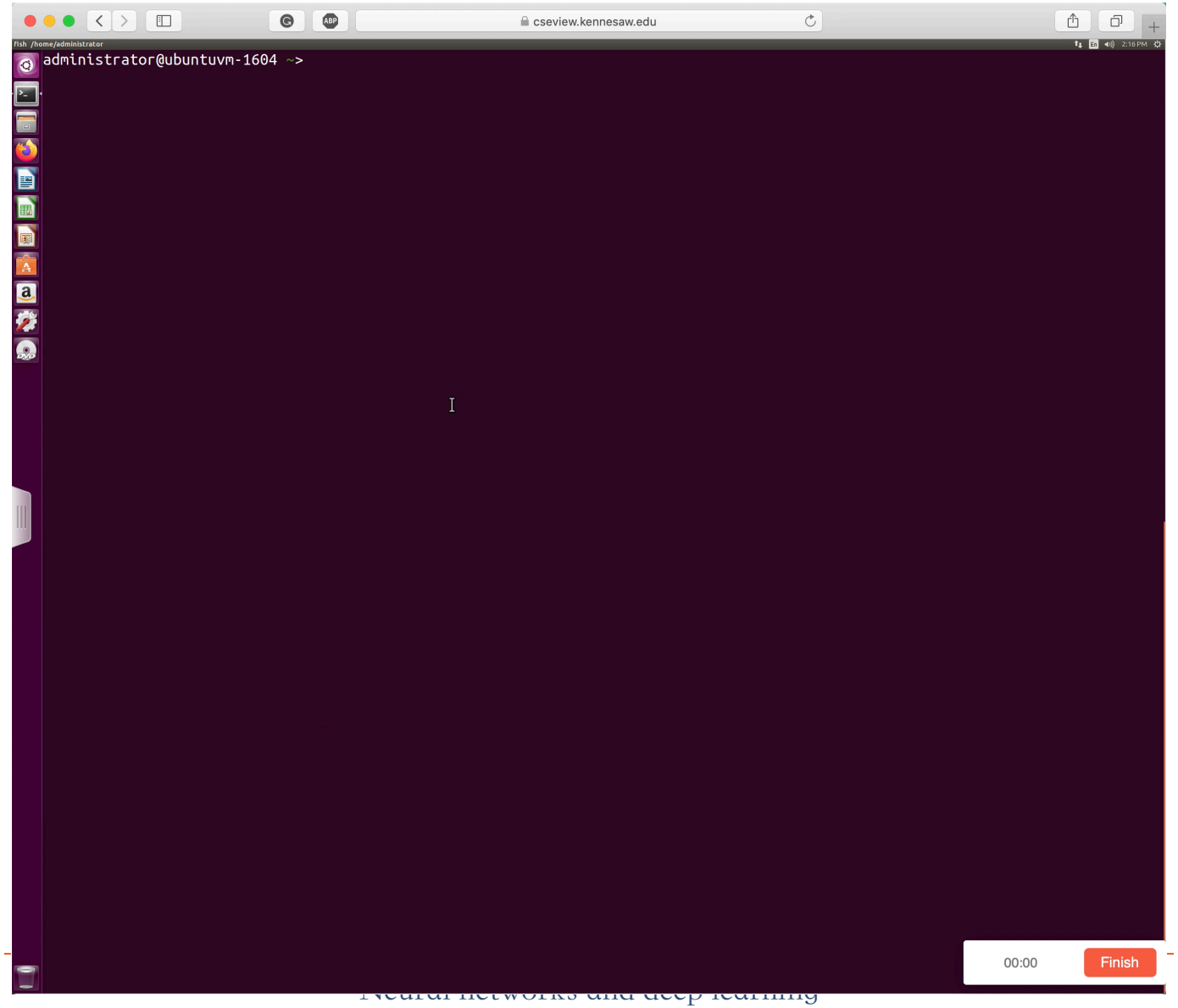
```
********************************************************************************
Cache Topology
********************************************************************************
Level:                  1
Size:                   32 kB
Cache groups:           ( 0 ) ( 1 ) ( 2 ) ( 3 )
--------------------------------------------------------------------------------
Level:                  2
Size:                   256 kB
Cache groups:           ( 0 ) ( 1 ) ( 2 ) ( 3 )
--------------------------------------------------------------------------------
Level:                  3
Size:                   16 MB
Cache groups:           ( 0 ) ( 1 ) ( 2 ) ( 3 )
--------------------------------------------------------------------------------
********************************************************************************
NUMA Topology
********************************************************************************
NUMA domains:           1
--------------------------------------------------------------------------------
Domain:                 0
Processors:             ( 0 1 2 3 )
Distances:              10
Free memory:            1967.79 MB
Total memory:           3942.19 MB
--------------------------------------------------------------------------------
```

```
ksuo@ksuo-VirtualBox ~/likwid-5.0.0> lscpu | grep cache
L1d cache:             32K
L1i cache:             32K
L2 cache:              256K
L3 cache:              16384K
```

Memory information

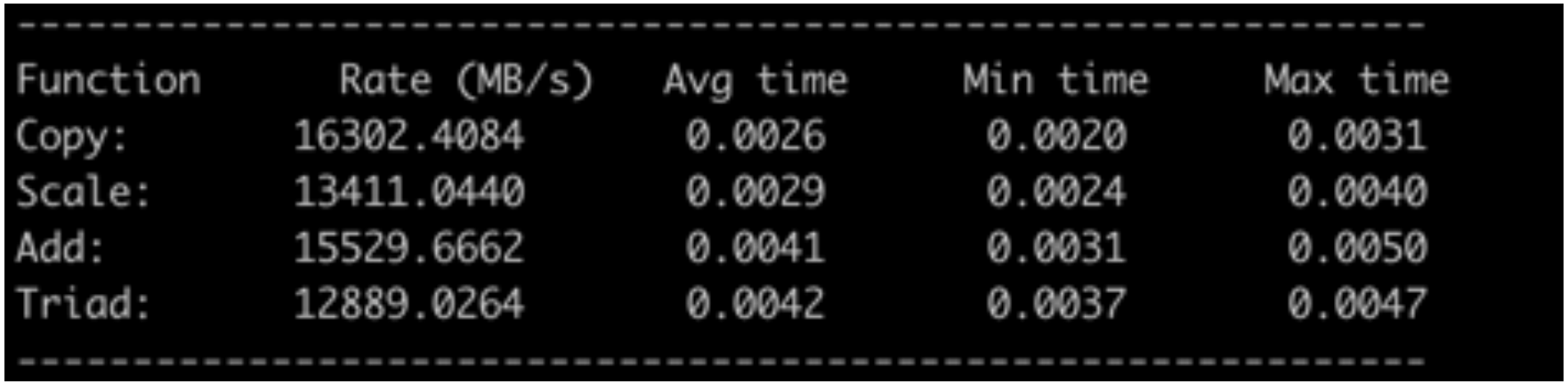
- lstopo: show the CPU cache and logical CPU layout
- Intall: $ sudo apt-get install hwloc
- Run: $ lstopo

Memory information



Test: what is the speed of your memory?

- STREAM Benchmark:
  - it can measure the performance of the memory system, including bandwidth and latency.
  - https://www.cs.virginia.edu/stream/



```
-------------------------------------------------------------
Function      Rate (MB/s)   Avg time     Min time     Max time
Copy:       16302.4084       0.0026       0.0020       0.0031
Scale:      13411.0440       0.0029       0.0024       0.0040
Add:        15529.6662       0.0041       0.0031       0.0050
Triad:      12889.0264       0.0042       0.0037       0.0047
-------------------------------------------------------------
```

Test: what is the speed of your memory?

Copy

```
void tuned_STREAM_Copy()
{
    int j;
        for (j=0; j<N; j++)
            c[j] = a[j];
}
```

Read from one memory cell

Write to another memory cell

Test: what is the speed of your memory?

```
Scale
void tuned_STREAM_Scale(double scalar)
{
    int j;
    for (j=0; j<N; j++)
        b[j] = scalar*c[j];
}
```

Read from one memory cell

Make a multiply operation

Write to another memory cell

Test: what is the speed of your memory?

Sum

```
void tuned_STREAM_Add()
{
    int j;
    for (j=0; j<N; j++)
        c[j] = a[j]+b[j];
}
```

Read from two memory cell

Make an add operation

Write to another memory cell

Test: what is the speed of your memory?

Triad

```
void tuned_STREAM_Triad(double scalar)
{
    int j;
    for (j=0; j<N; j++)
        a[j] = b[j]+scalar*c[j];
}
```

Read from two memory cell

Make an add operation and a multiply operation

Write to another memory cell

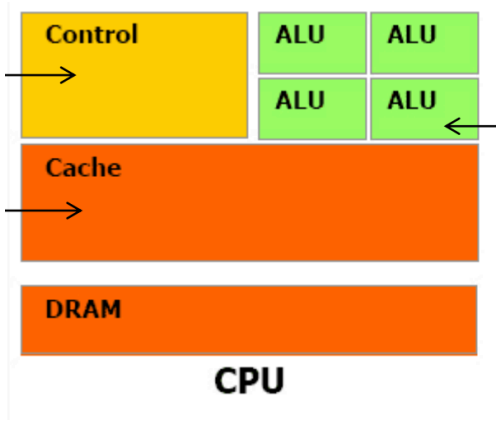
Test: what is the speed of your memory?

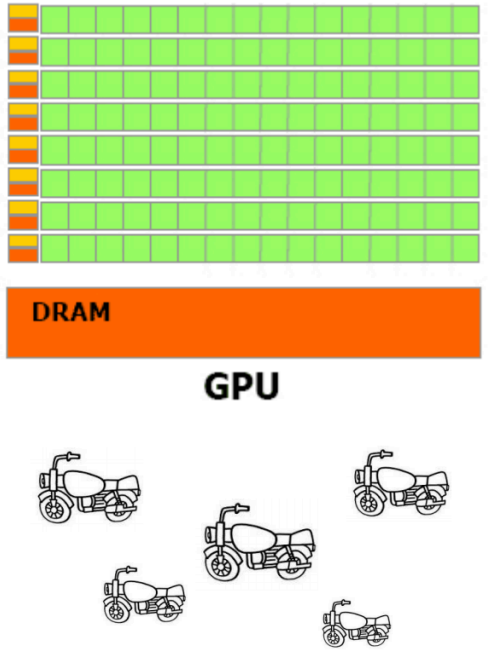
- 1, Download STREAM Benchmark
  - $ wget https://www.nersc.gov/assets/Trinity--NERSC-8-RFP/Benchmarks/Jan9/stream.tar

- 2, Compile & run
  - $ tar xvf stream.tar
    $ gcc stream.c -o stream
    $ ./stream

GPU

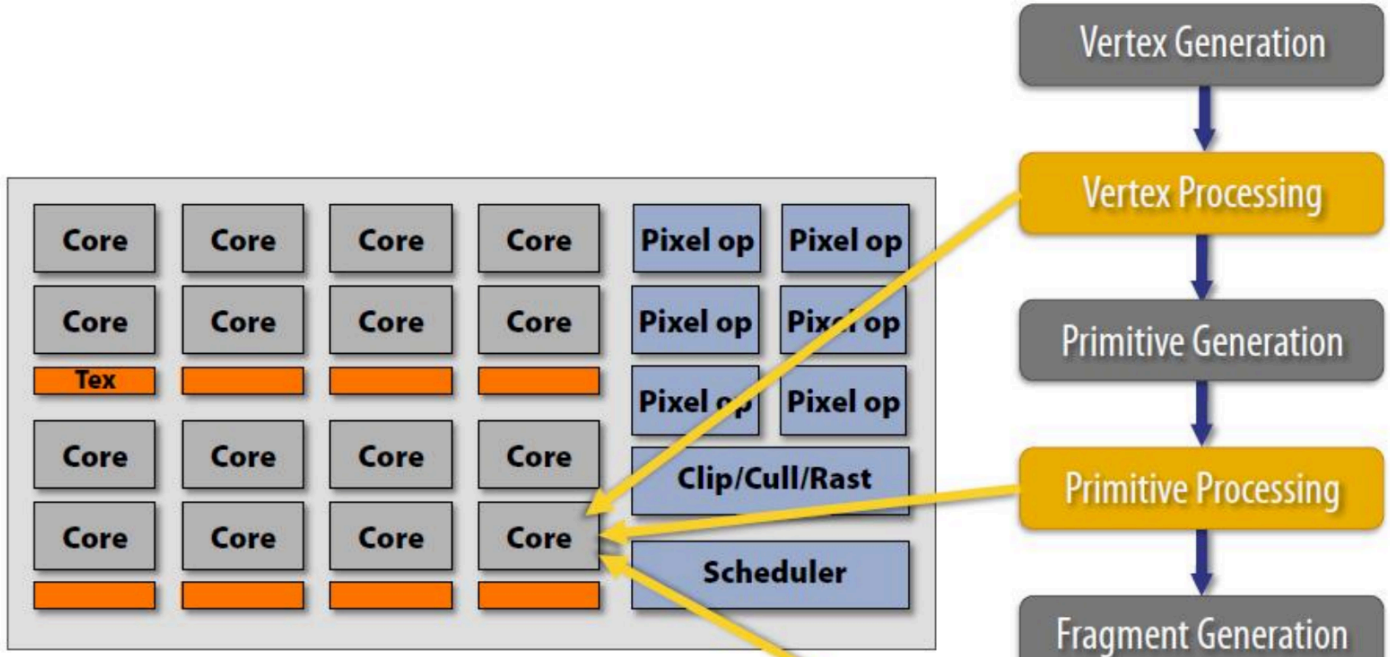
- Graphical Processing Unit
  - The task is to composite and display images with millions of pixels on the screen
  - Good at large numbers, simple, and inherently difficult to do complex calculations (logic, process control)
- GPU has many cores





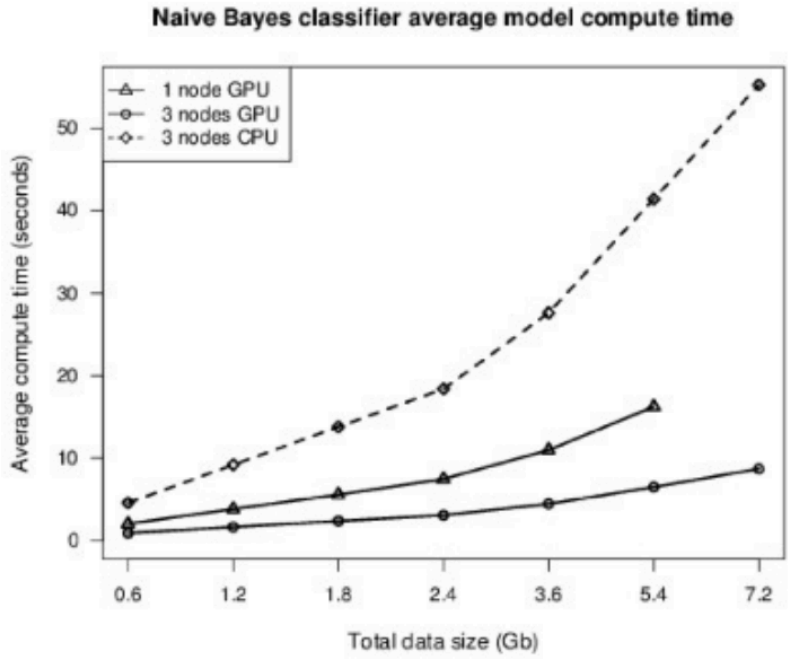
GPU

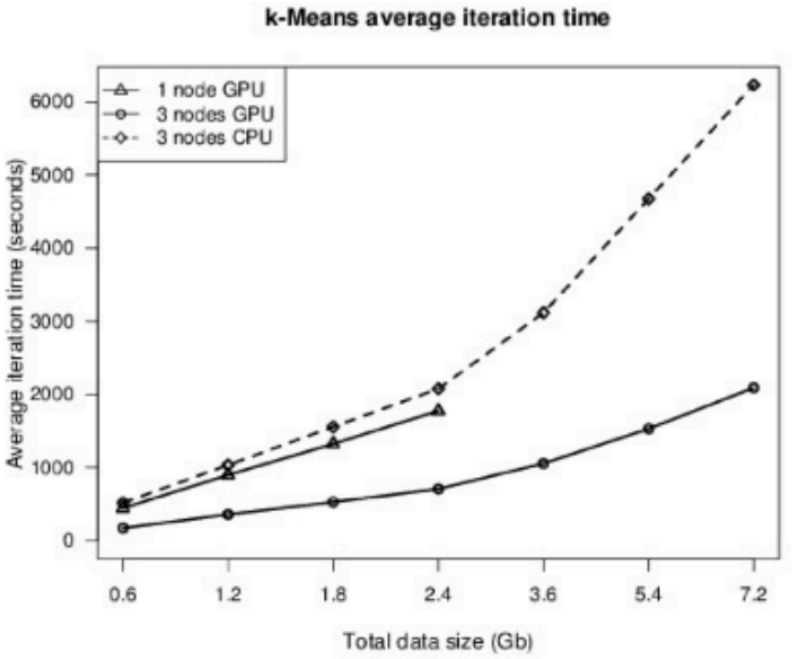
- Each pixel can be given to A CORE (core) processing



GPU

SPEEDUPS



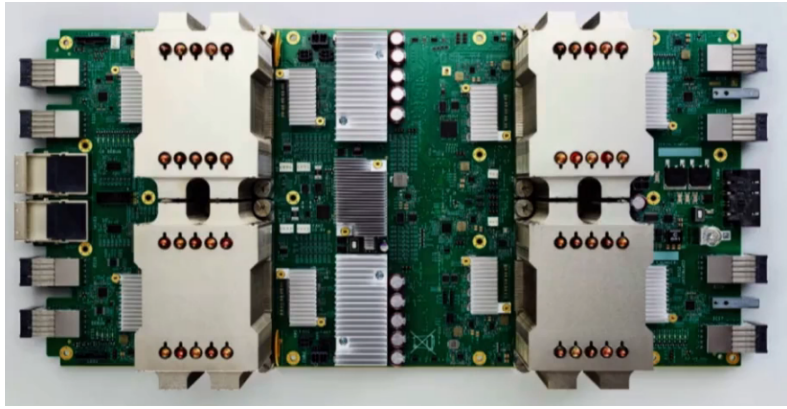


Up to 12x speedup for a same task when nodes are equipped with GPU

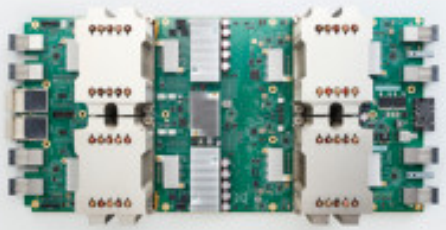
GPU

TPU: Tensor Processing Unit

- Proposal: Design a custom ASIC for the inference phase of NN (training still happens using GPUs)
- Principles:
  - improve cost-performance by 10X compared to GPUs
  - simple design for response time guarantees(single-thread, no prefetching, no OOO etc.)
- Characteristics:
  - More like a co-processor to reduce time-to-market delays
    - Host sends instructions to TPU
- connected through PCIe I/O bus



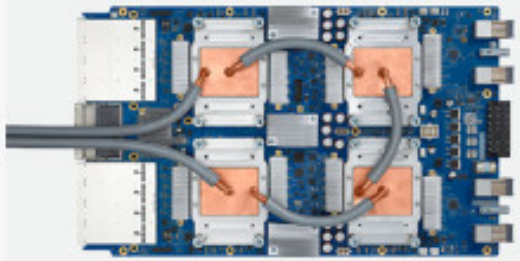
TPU: Tensor Processing Unit



Cloud TPU v2

180 teraflops

64 GB High Bandwidth Memory (HBM)



Cloud TPU v3 Alpha

420 teraflops

128 GB HBM

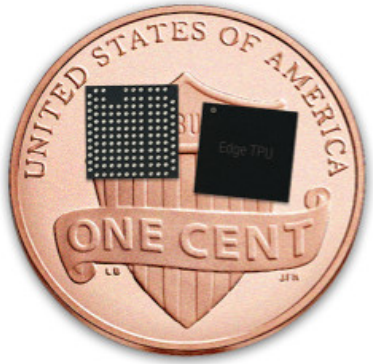
Cloud TPU v2 Pod Alpha

11.5 petaflops

4 TB HBM

2-D toroidal mesh network

Edge TPU



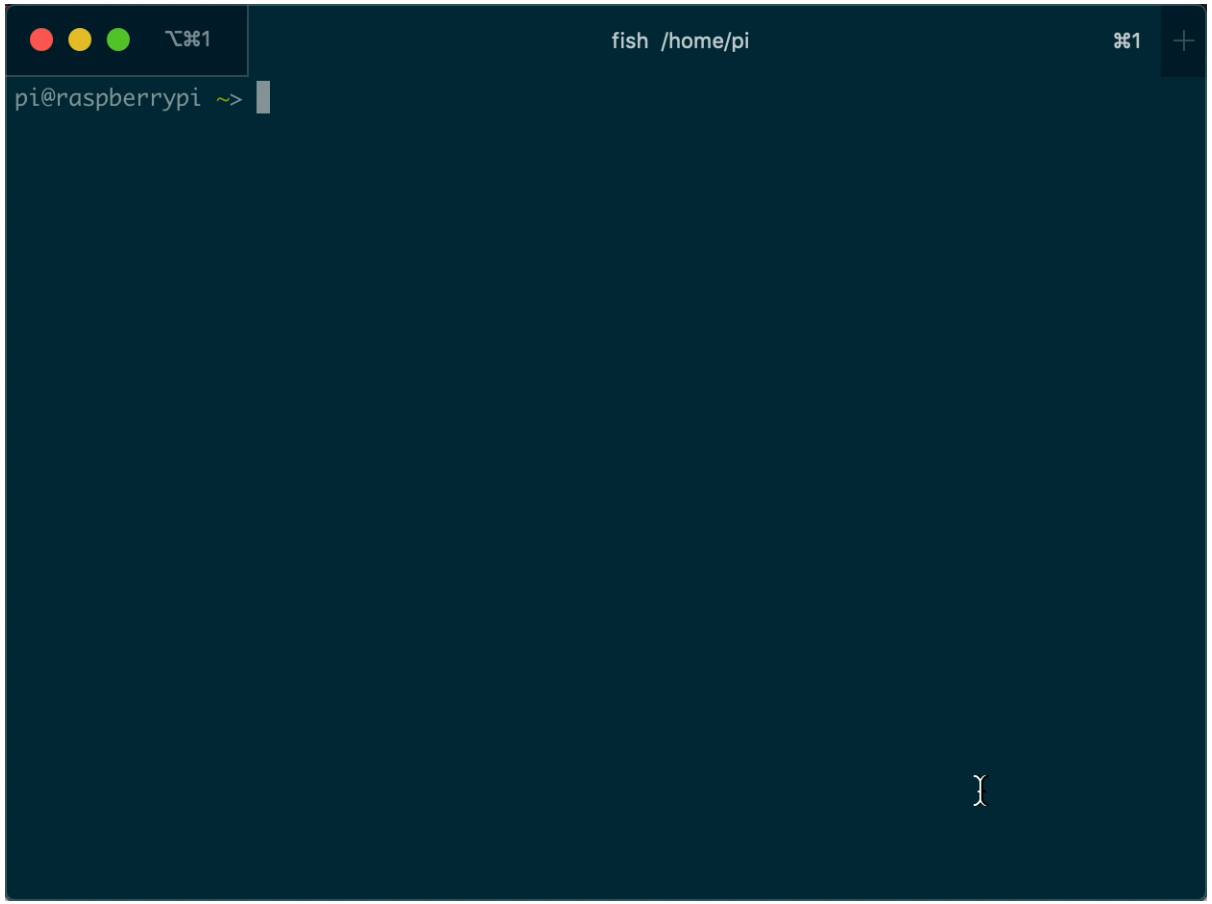
An interesting video introducing hardware

Test: Your VM hardware specs?

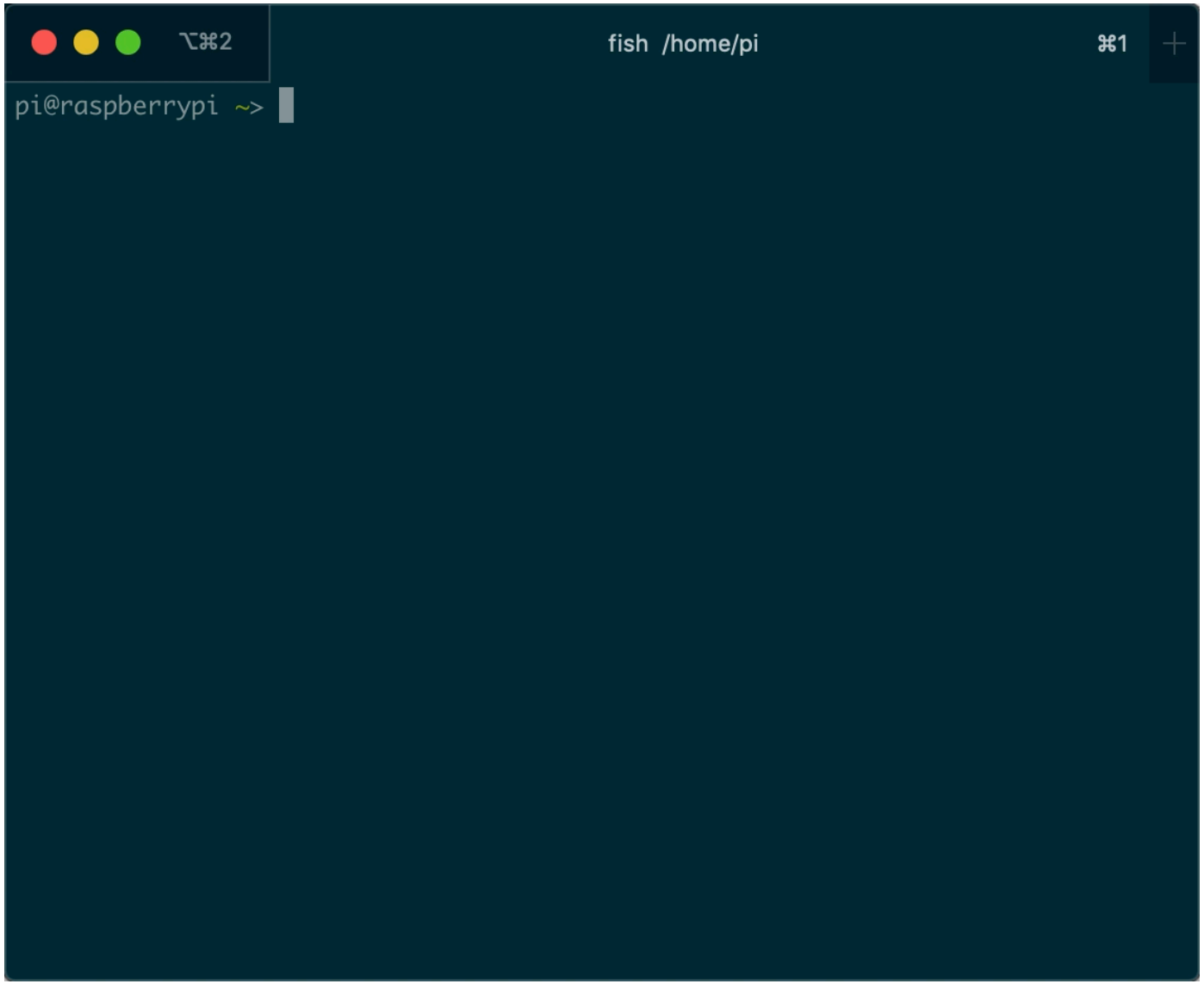
- CPU? → How many cores? Frequency? Topology?
- Cache? → Size?
- Memory? → Size? Speed?
- ~~Disk? → Size?~~
- ~~Network? → Latency? Bandwidth?~~

How to get my machine spec?

- Inxi: https://www.tecmint.com/inxi-command-to-find-linux-system-information/



How to get my machine spec? One command for All!



AI benchmark

- AI Benchmark Alpha is an open-source python library for evaluating AI performance of various hardware platforms, including CPUs, GPUs and TPUs. The benchmark is relying on TensorFlow machine learning library and is providing a lightweight and accurate solution for assessing inference and training speed for key Deep Learning models.

- http://ai-benchmark.com/

- https://arxiv.org/pdf/1910.06663.pdf

AI benchmark

- In total, AI Benchmark consists of 42 tests and 19 sections provided below:
  - MobileNet-V2 [classification]
  - Inception-V3 [classification]
  - Inception-V4 [classification]
  - Inception-ResNet-V2 [classification]
  - ResNet-V2-50 [classification]
  - ResNet-V2-152 [classification]
  - VGG-16 [classification]
  - SRCNN 9-5-5 [image-to-image mapping]
  - VGG-19 [image-to-image mapping]
  - ResNet-SRGAN [image-to-image mapping]
  - ResNet-DPED [image-to-image mapping]
  - U-Net [image-to-image mapping]
  - Nvidia-SPADE [image-to-image mapping]
  - ICNet [image segmentation]
  - PSPNet [image segmentation]
  - DeepLab [image segmentation]
  - Pixel-RNN [inpainting]
  - LSTM [sentence sentiment analysis]
  - GNMT [text translation]

Installing AI benchmark

```
apt update -y

apt-get install python3-pip python3-dev nmon fish

pip3 install tensorflow
# or
pip3 install tensorflow-gpu

#check version
python3 -c 'import tensorflow as tf; print(tf.__version__)'  # for Python 3

pip3 install ai-benchmark
```

Run AI benchmark

- Create test.py file as:

```
from ai_benchmark import AIBenchmark
benchmark = AIBenchmark()
results = benchmark.run_micro()
```

- python3 test.py
  - https://www.youtube.com/watch?v=OlbUcSlc0Uc&ab_channel=TonyJ

Monitor the AI-benchmark and visualize the resource consumption

- $ nmon -fTt -s 1 -c 10000

- When benchmark finished, killall nmon. On the disk, you will see file similar as raspberrypi_190927_2015.nmon

- Install nmonchart
  - $ git clone https://github.com/aguther/nmonchart.git

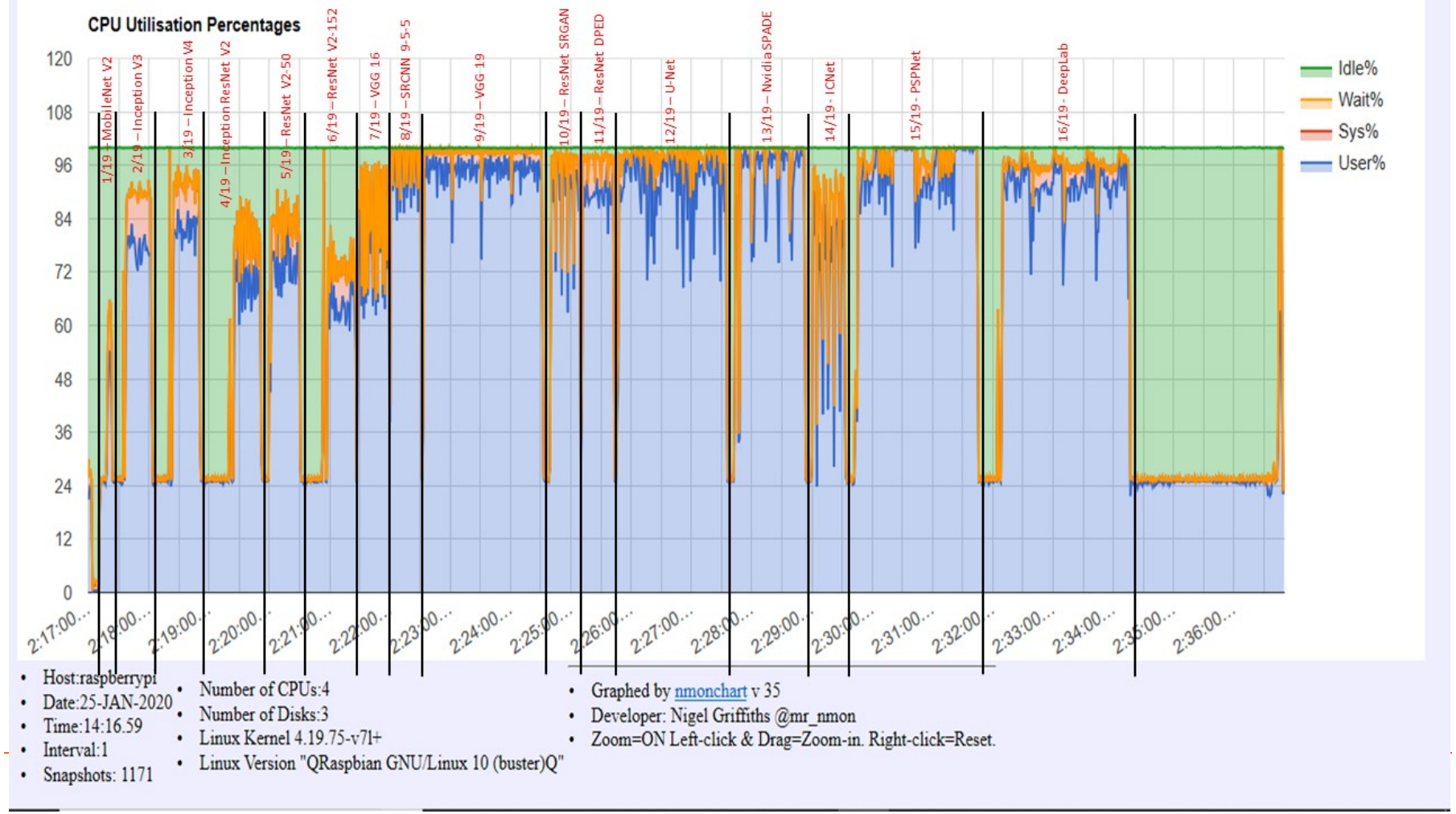
- Change nmonchart to executable
  - $ sudo chmod +x ./nmonchart/nmonchart

- Install ksh
  - $ sudo apt-get install ksh -y

- Transfer nmon file to html file
  - $ nmonchart raspberrypi_190927_2015.nmon raspberrypi_190927_2015.html

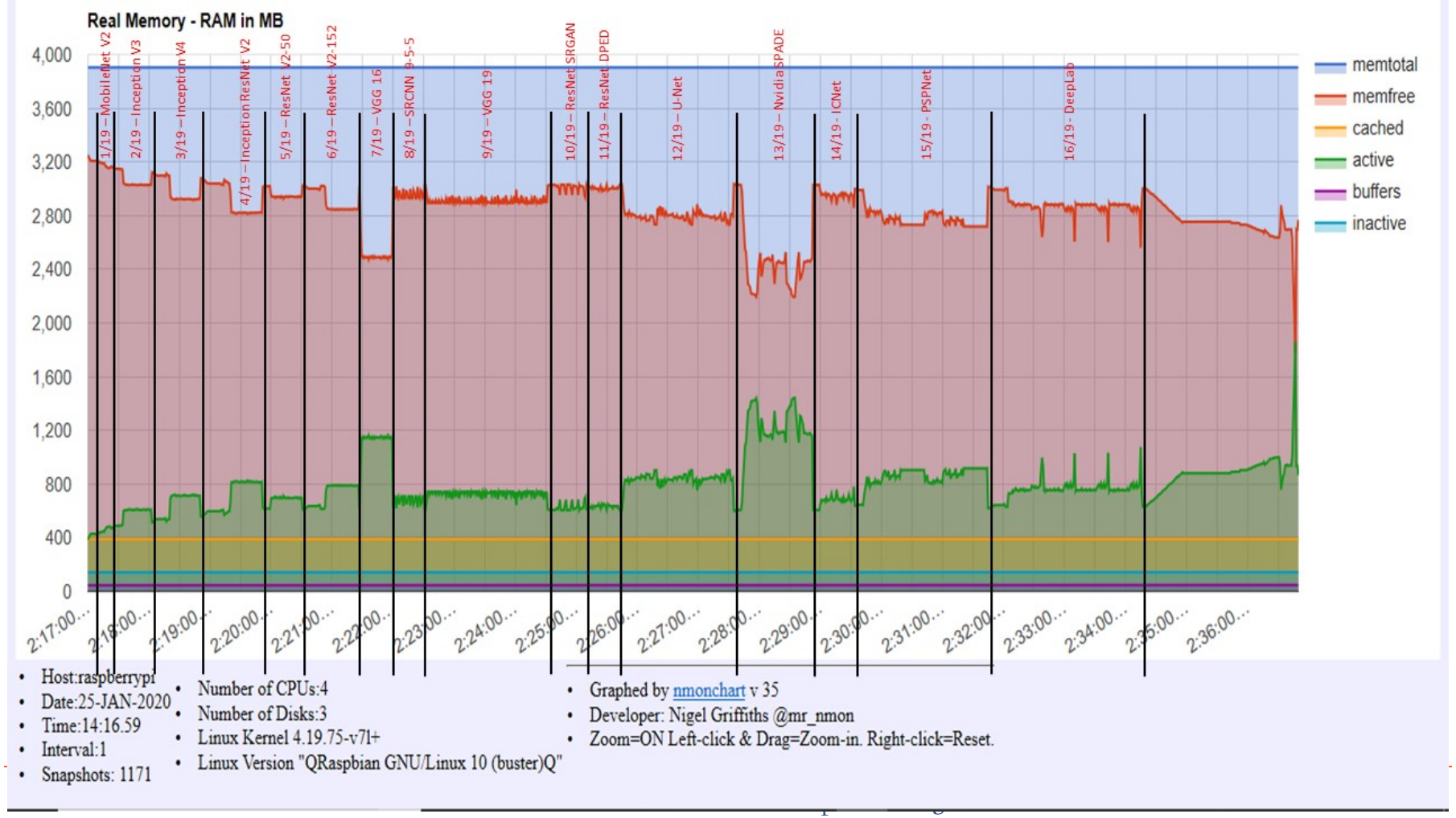
Output file

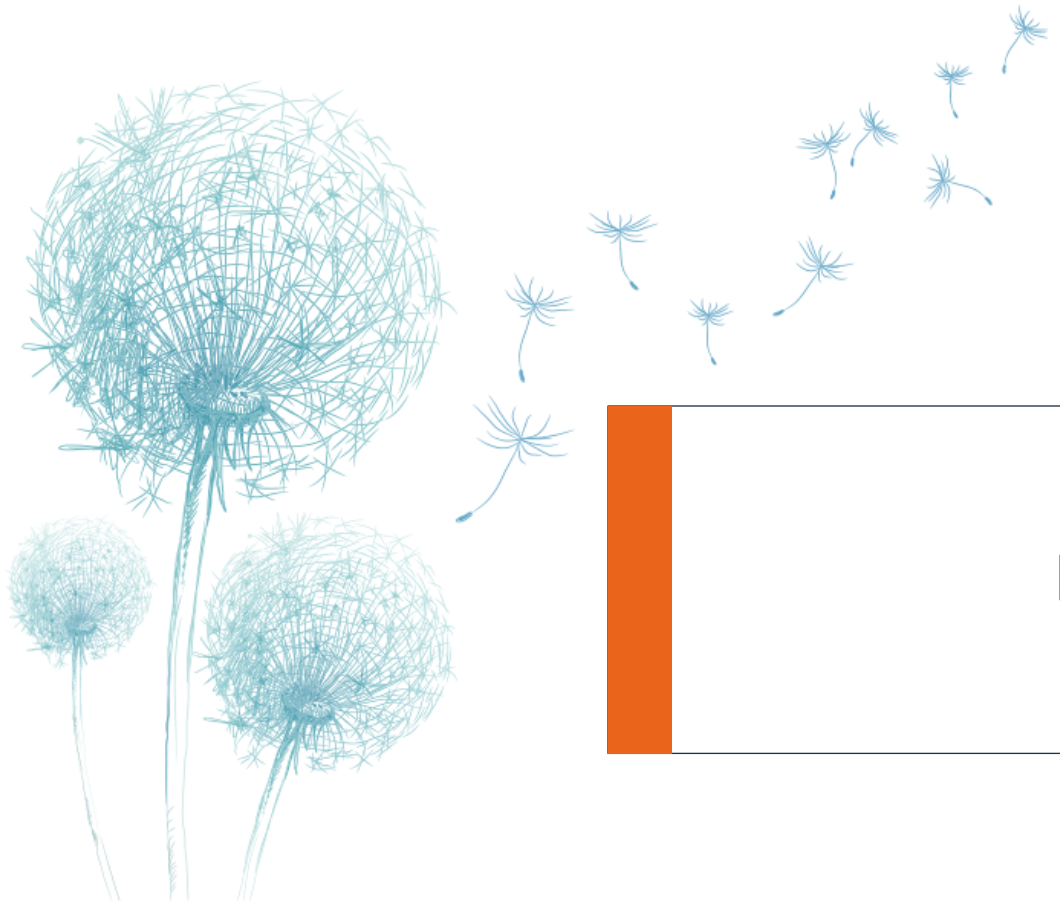
- https://kevinsuo.github.io/code-lab-files/results/rsbp4/2-raspberrypi_200125_1416.html



Output file

- https://kevinsuo.github.io/code-lab-files/results/rsbp4/2-raspberrypi_200125_1416.html

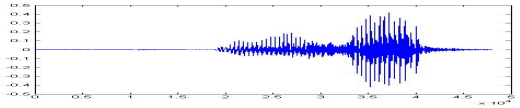




Machine learning

Machine learning ≈ build a mapping function

- Speech Recognition

$f($  $)=$ “Hello”

- Image Identification

$f($  $)=$ “Cat”

- Go

$f($  $)=$ “5-5” (position)

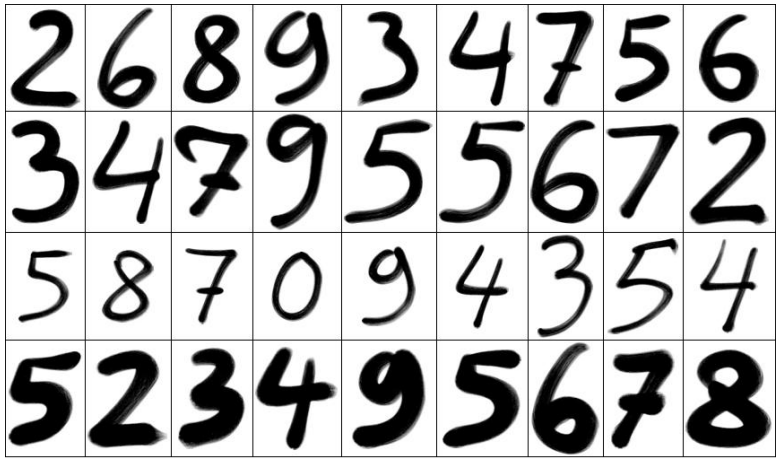
- Dialogue system

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

User input Machine

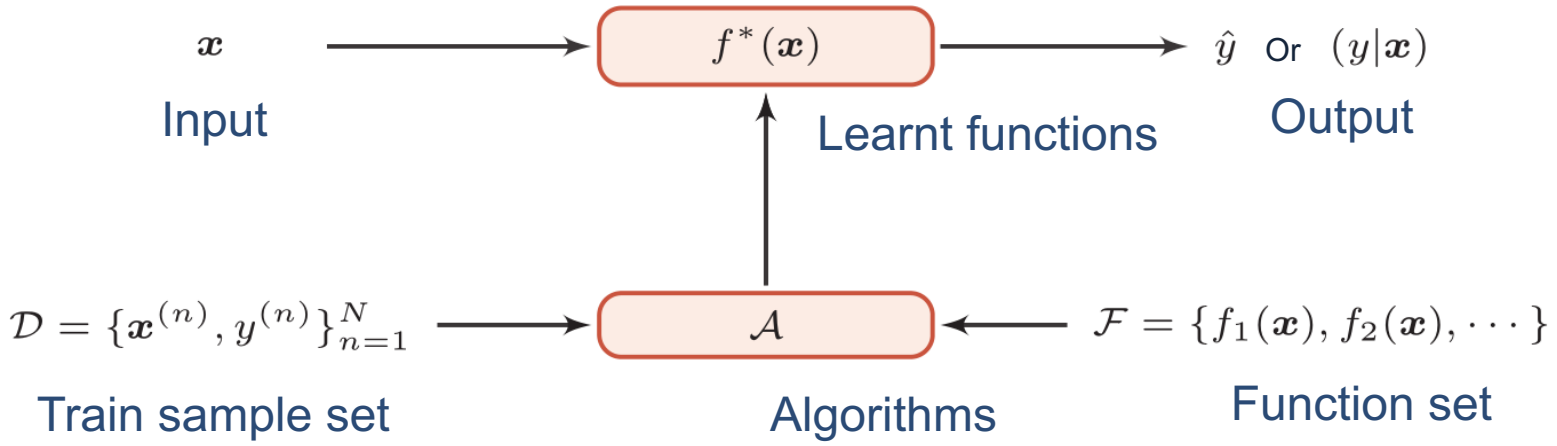
Why we need "machine learning"?

- 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.
  - Law: Decision (prediction) function



The three elements of machine learning

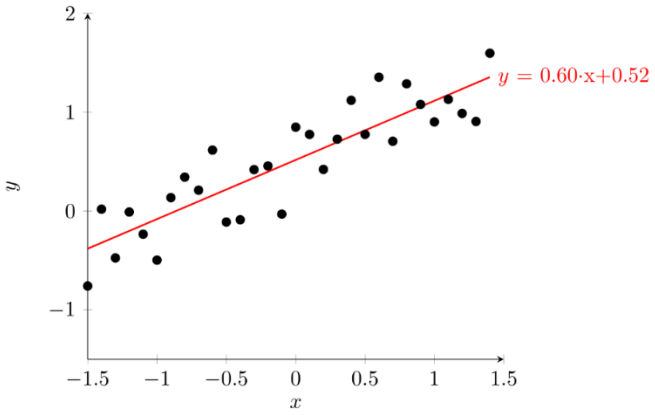
- Model
  - Linear method : $f(\mathbf{x}, \theta) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + b$
  - Generalized linear method : $f(\mathbf{x}, \theta) = \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}) + b$
    - If φ(x) is a learnable nonlinear basis function, f(x,θ) is equivalent to a neural network.

- Study criteria
  - Expected risk

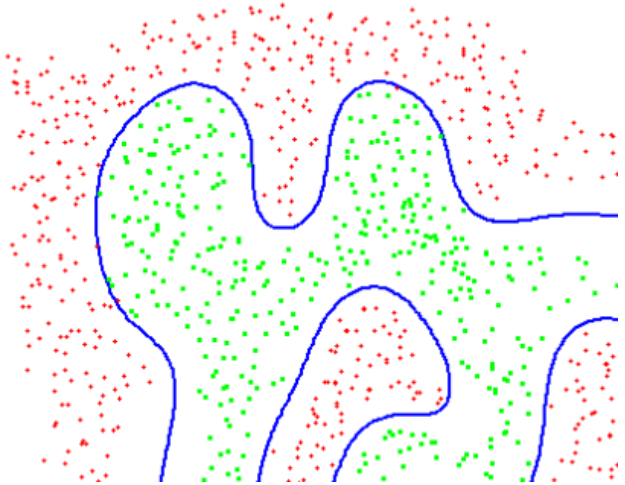
$$\mathcal{R}(f) = \mathbb{E}_{(\mathbf{x},y)\sim p(\mathbf{x},y)}[\mathcal{L}(f(\mathbf{x}), y)],$$

- Optimization
  - Gradient descent

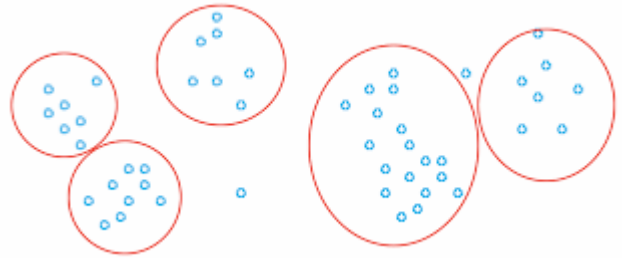
Common machine learning problems and models



regression



classification

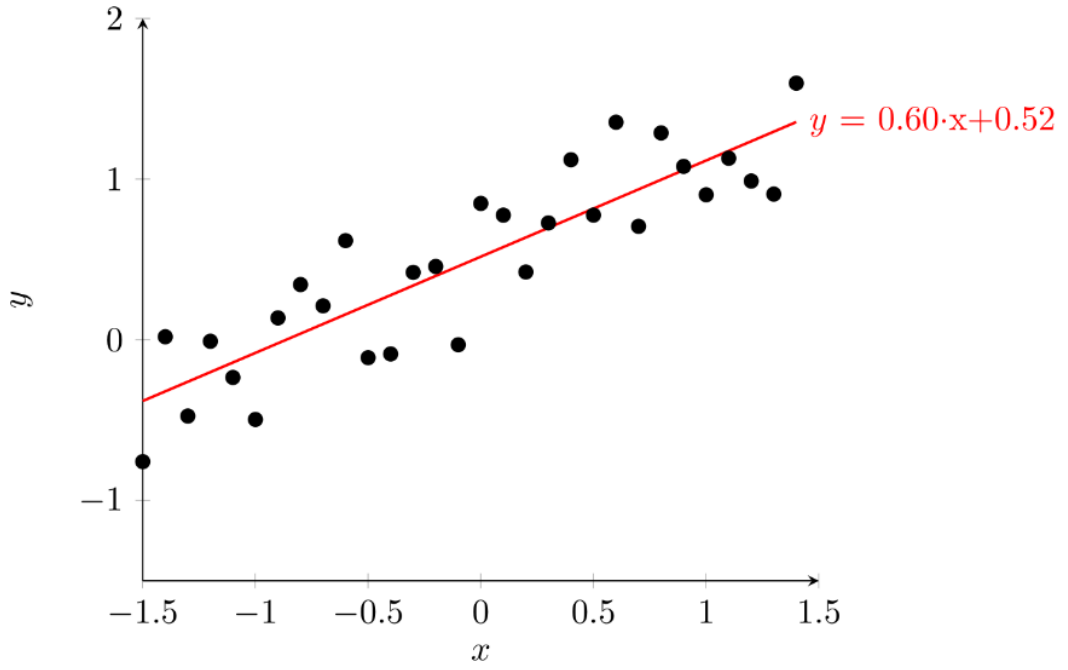


clustering

Model

- Take Linear Regression as an example
  - Model:

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



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: $\mathcal{D} = \{\mathrm{x}^{(\mathrm{n})}, y^{(\mathrm{n})}\}, i \in [1, N]$

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

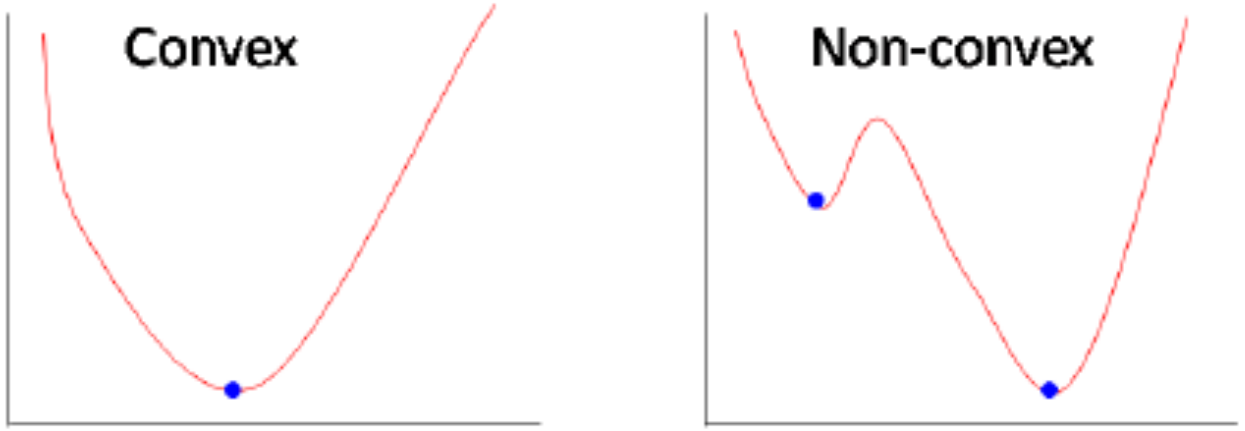
- Minimize experience risk
  - After selecting the appropriate risk function, we look for a parameter θ* to minimize the empirical risk function.

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

- Machine learning problem turned into an optimization problem

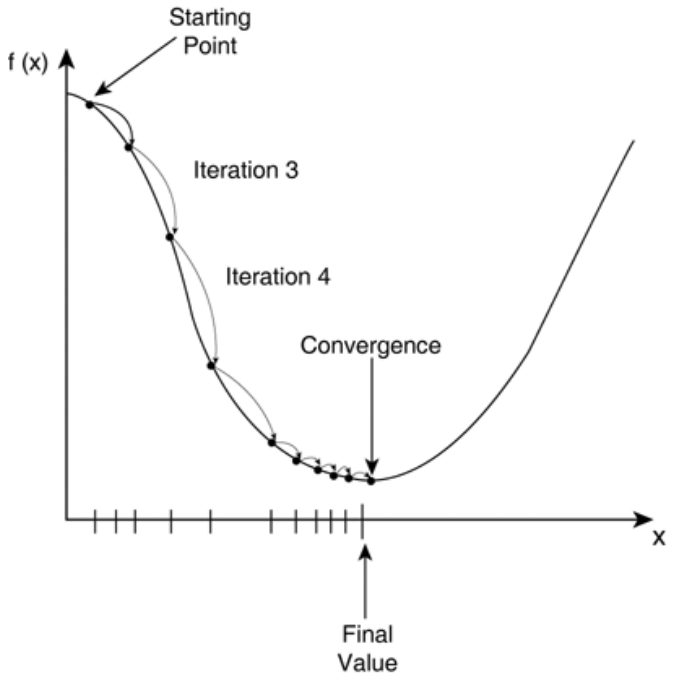
Optimization problem

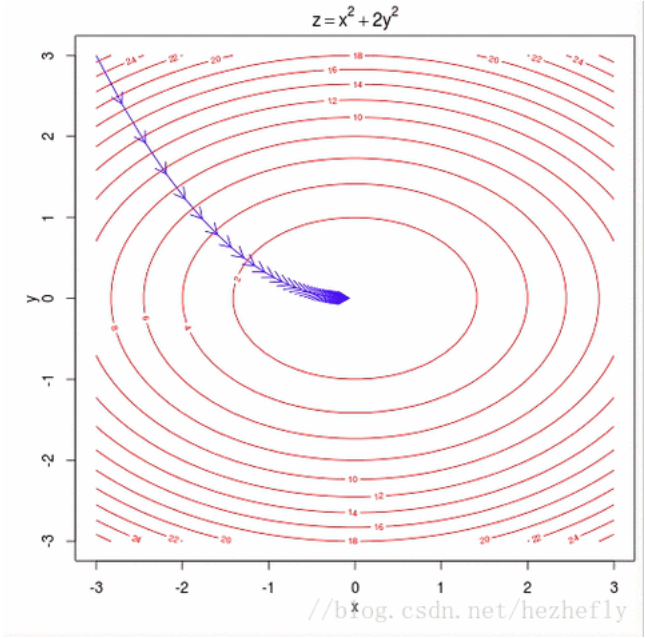
- Machine learning problem turned into an optimization problem



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

Gradient Descent



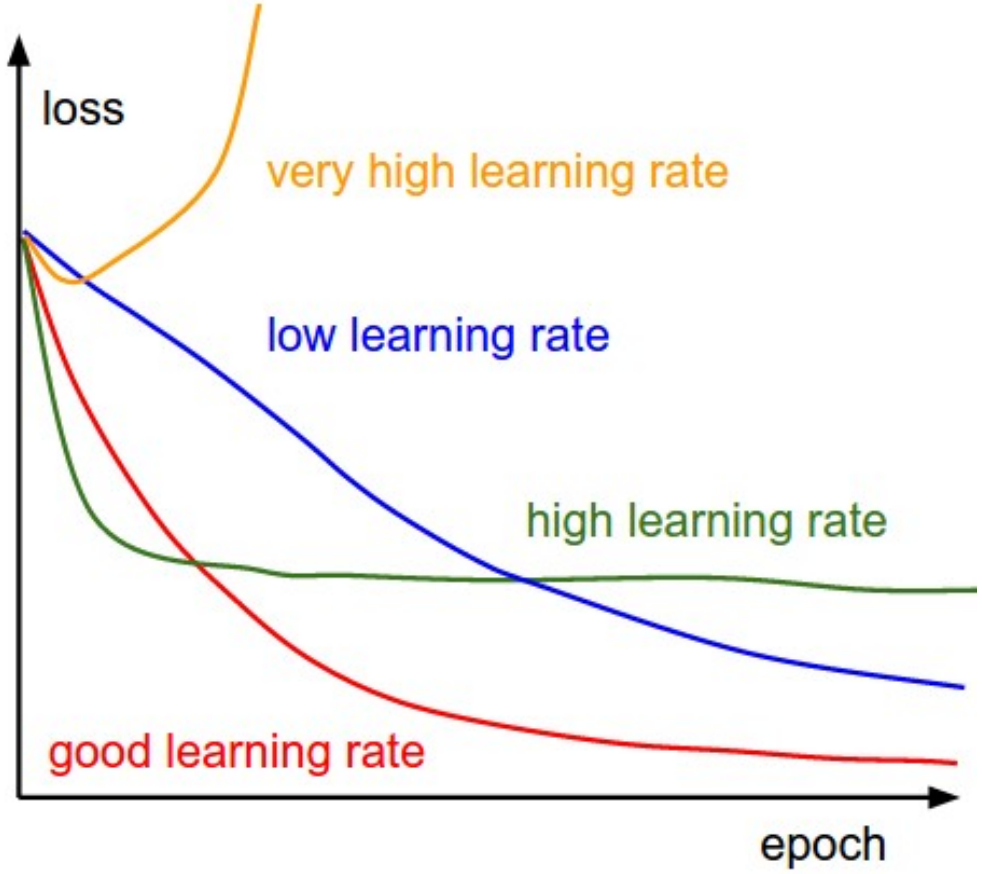


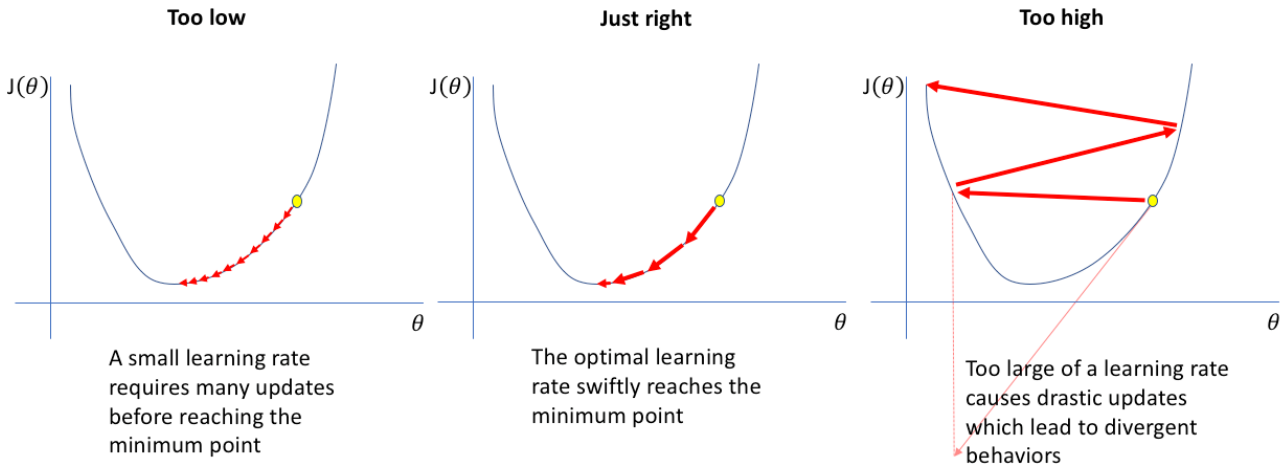
$$\theta_{t+1} = \theta_t - \alpha \frac{\partial \mathcal{R}(\theta)}{\partial \theta_t}$$

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

Search step α is also called *Learning Rate*

Learning rate is a very important hyperparameter!





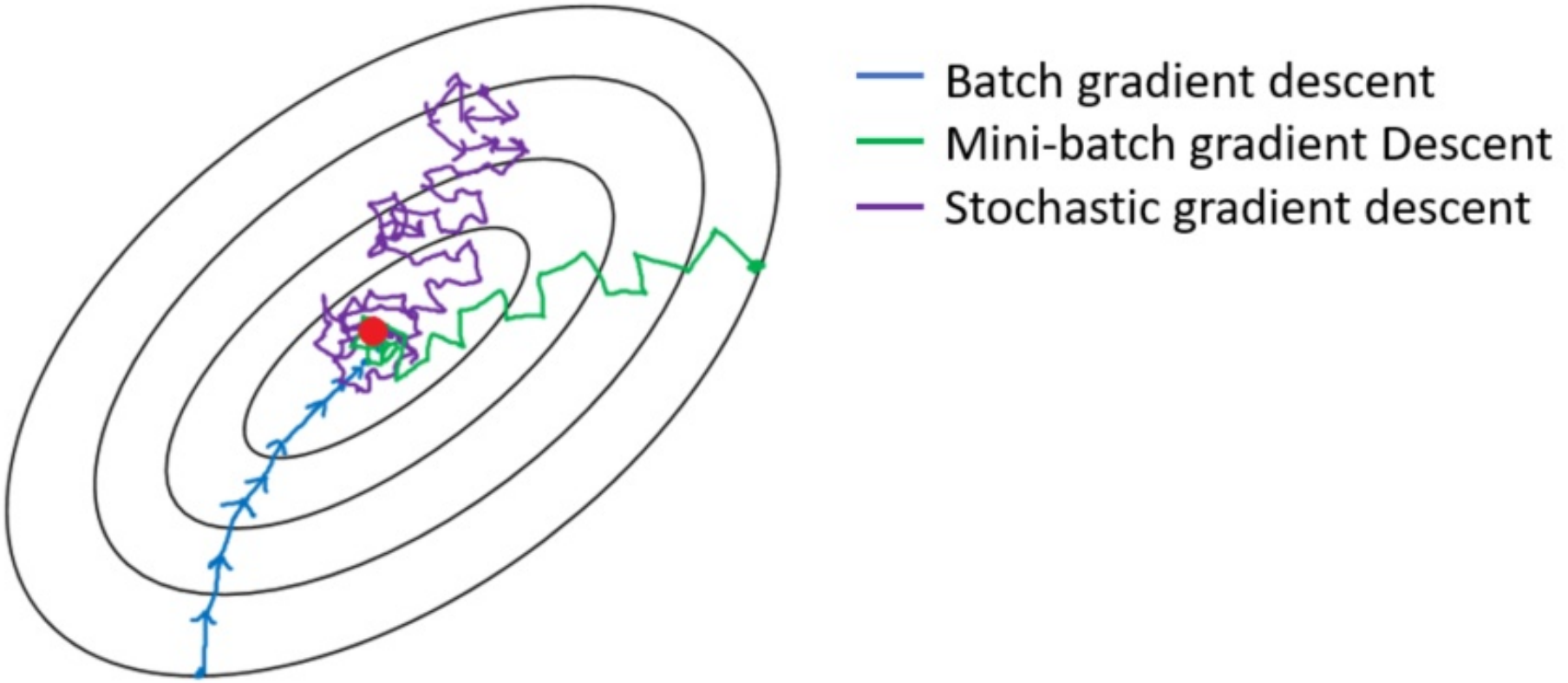
Stochastic Gradient Descent

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

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

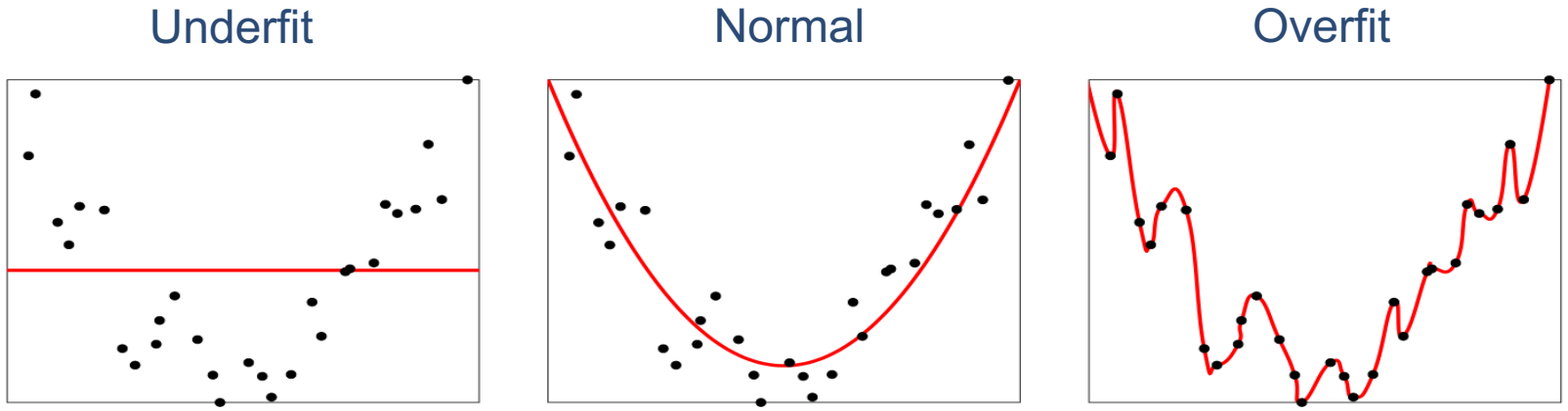
- Mini-Batch Stochastic Gradient Descent
  - Use b samples for each update. In fact, batch gradient descent is a compromise method.

Stochastic Gradient Descent



Machine learning = optimization?

Machine learning = optimization?



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

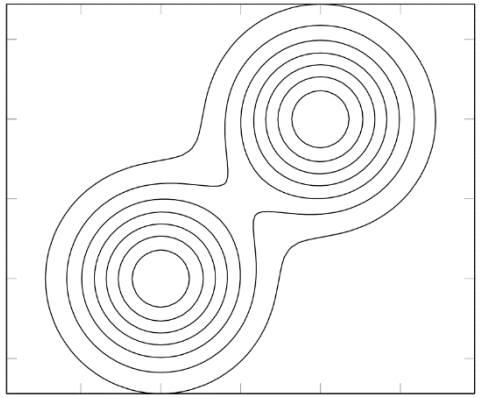
Expected risk

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

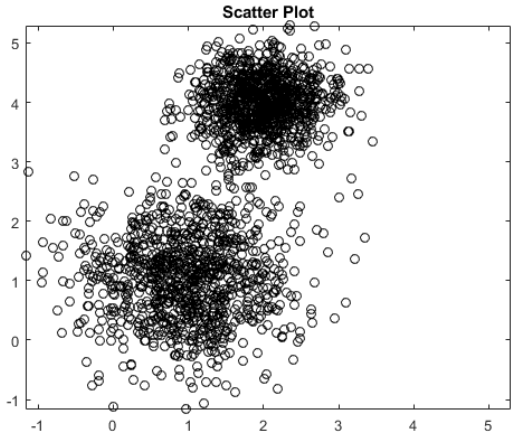
Empirical risk

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

Real distribution



$\neq$



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

Generalization error

How to reduce generalization errors?

Optimization

Minimal empirical risk

Regularization

Reduce model complexity

Regularization

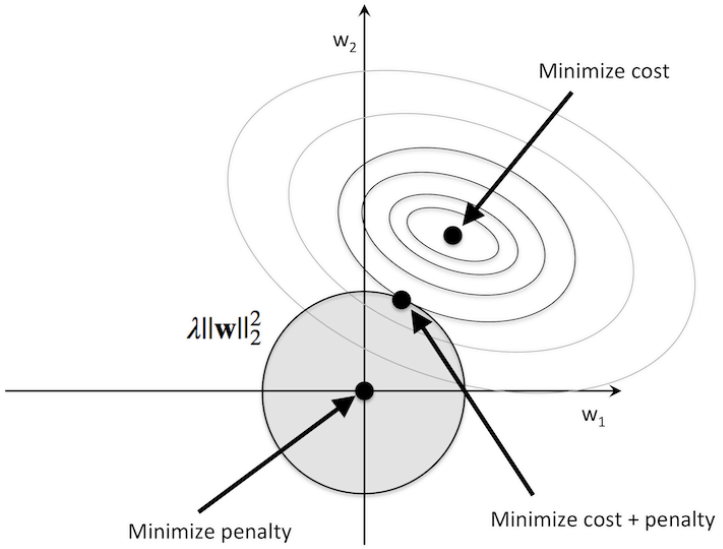
All regularization will harm the optimization

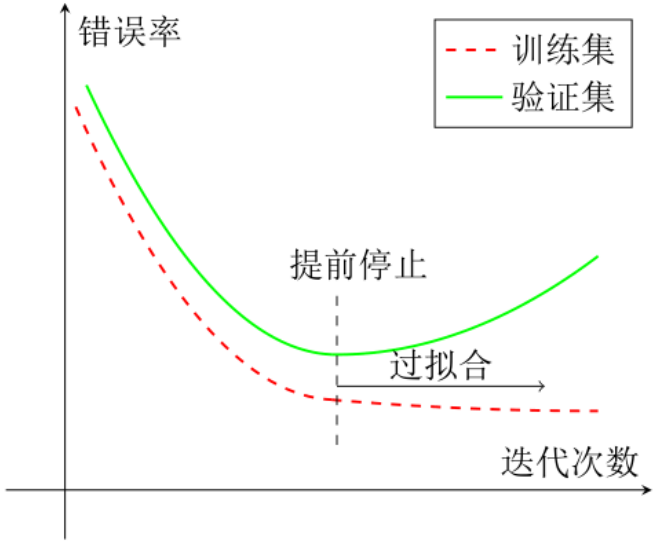
Increase optimization constraints

L1/L2 constraints, data enhancement

Interfere optimization process

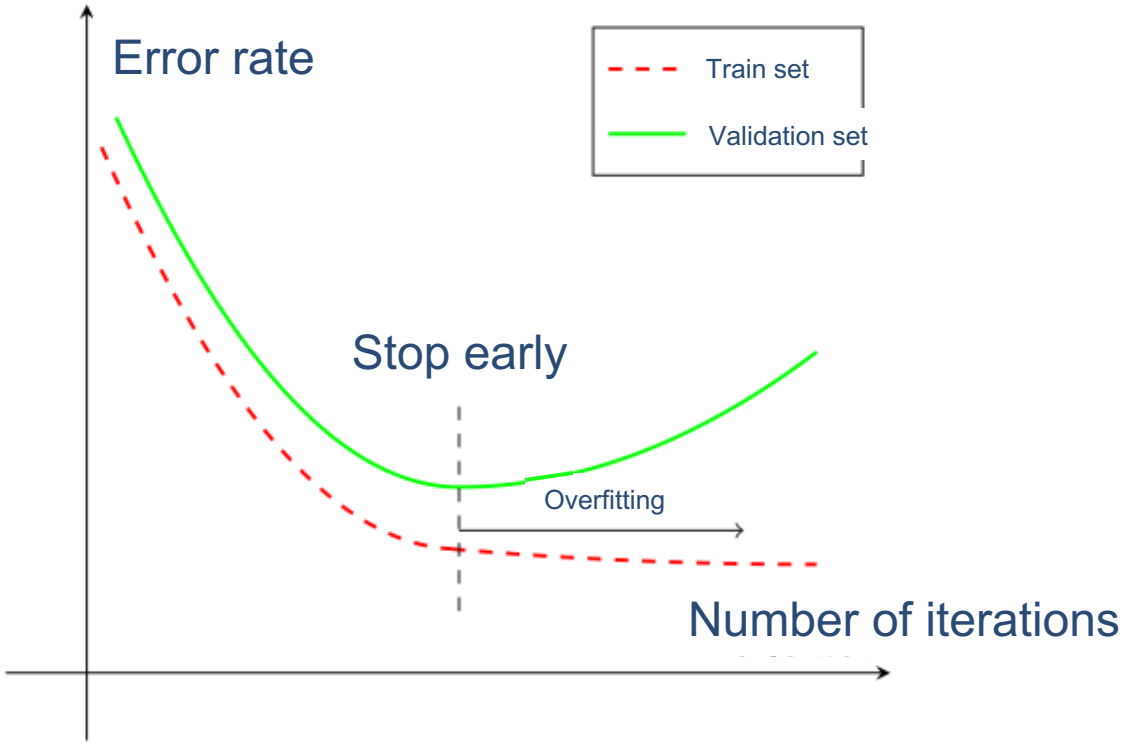
Weight decay, stochastic gradient descent, early stop





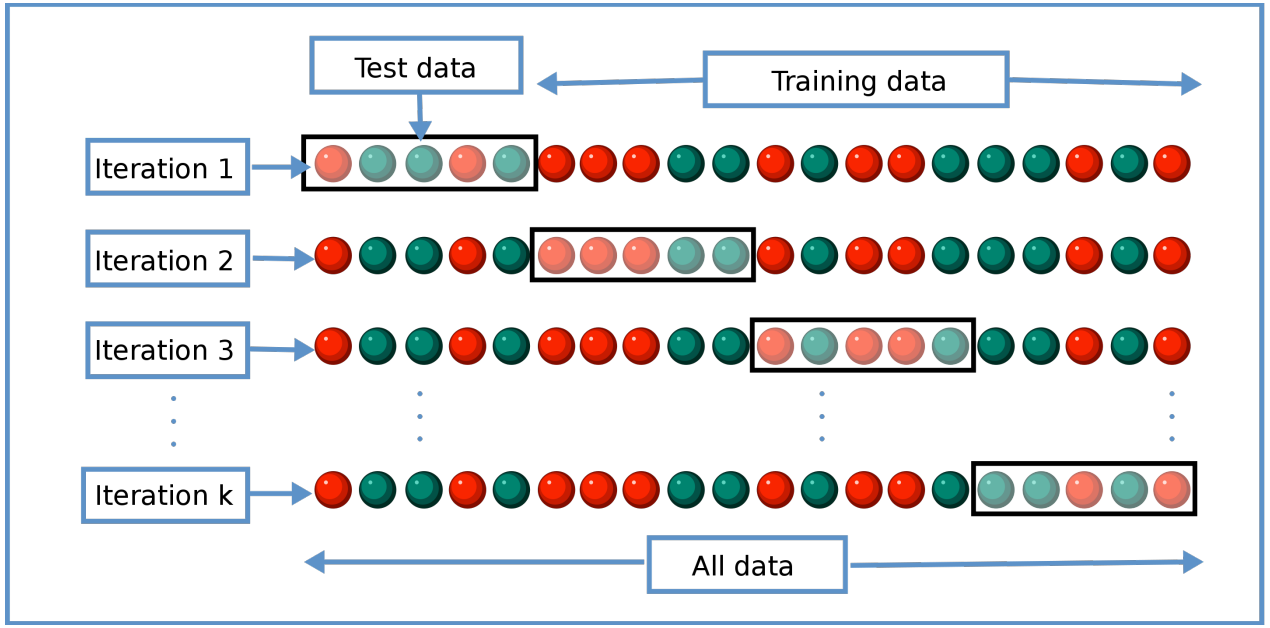
Stop early

- 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

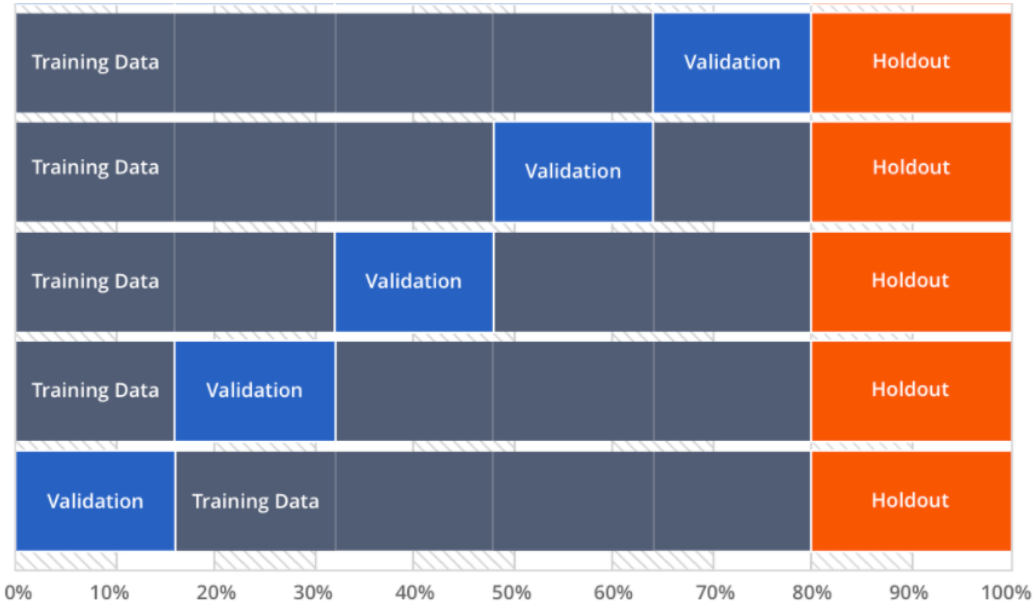
- Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample.
- Repeated use of data, split the sample data obtained, and combine them into different training sets and test sets



Why we need cross validation?

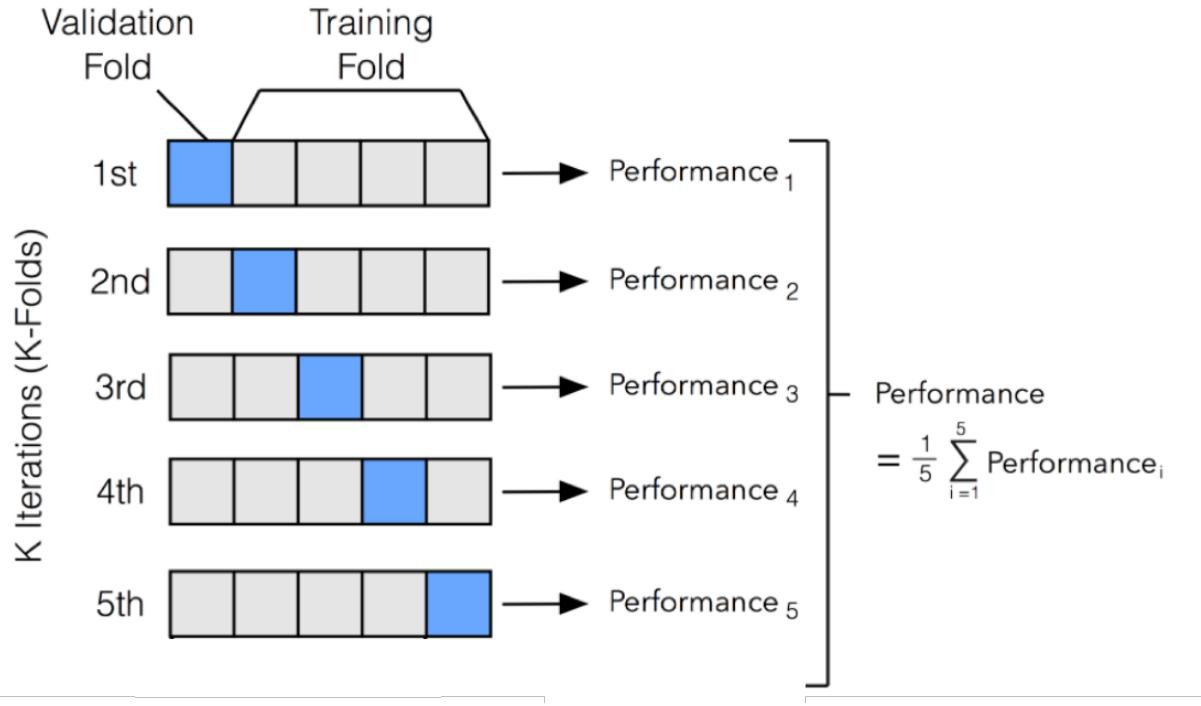
- Cross-validation is used when the amount of data is not very sufficient (for example, the amount of data is less than 10,000), and can obtain as much effective information as possible from the limited data.

- 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.



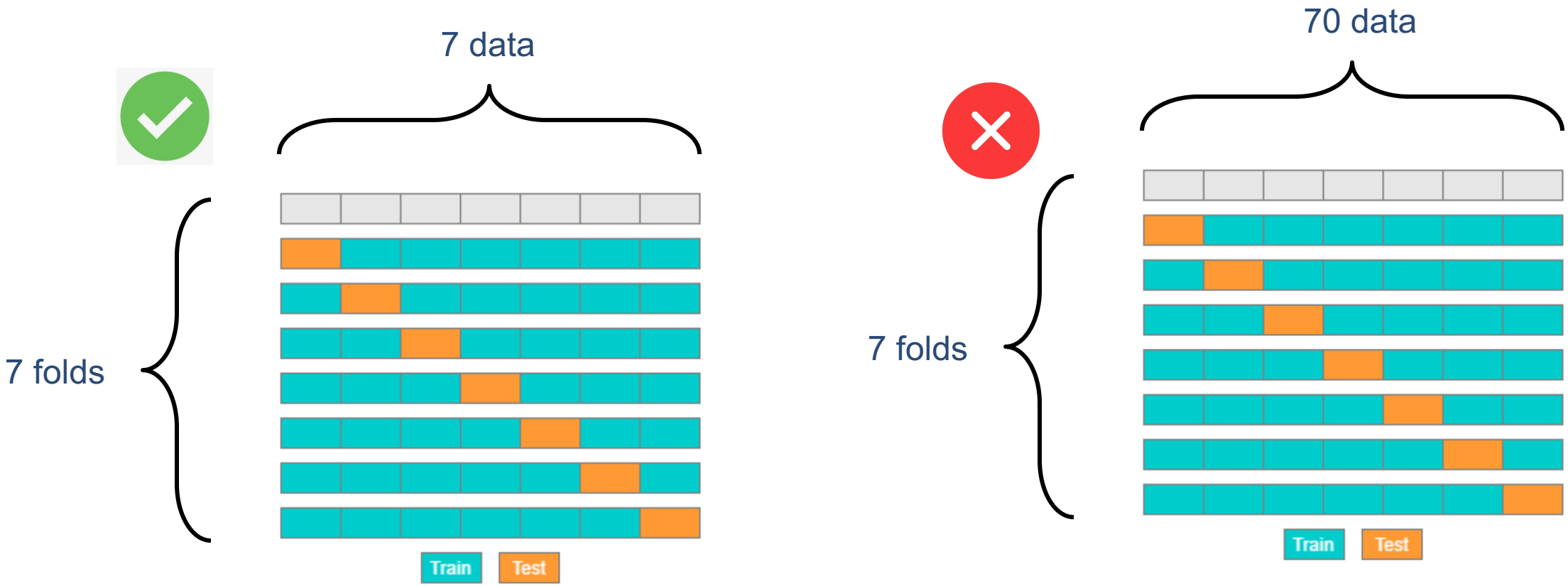
k-fold Cross Validation

- 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



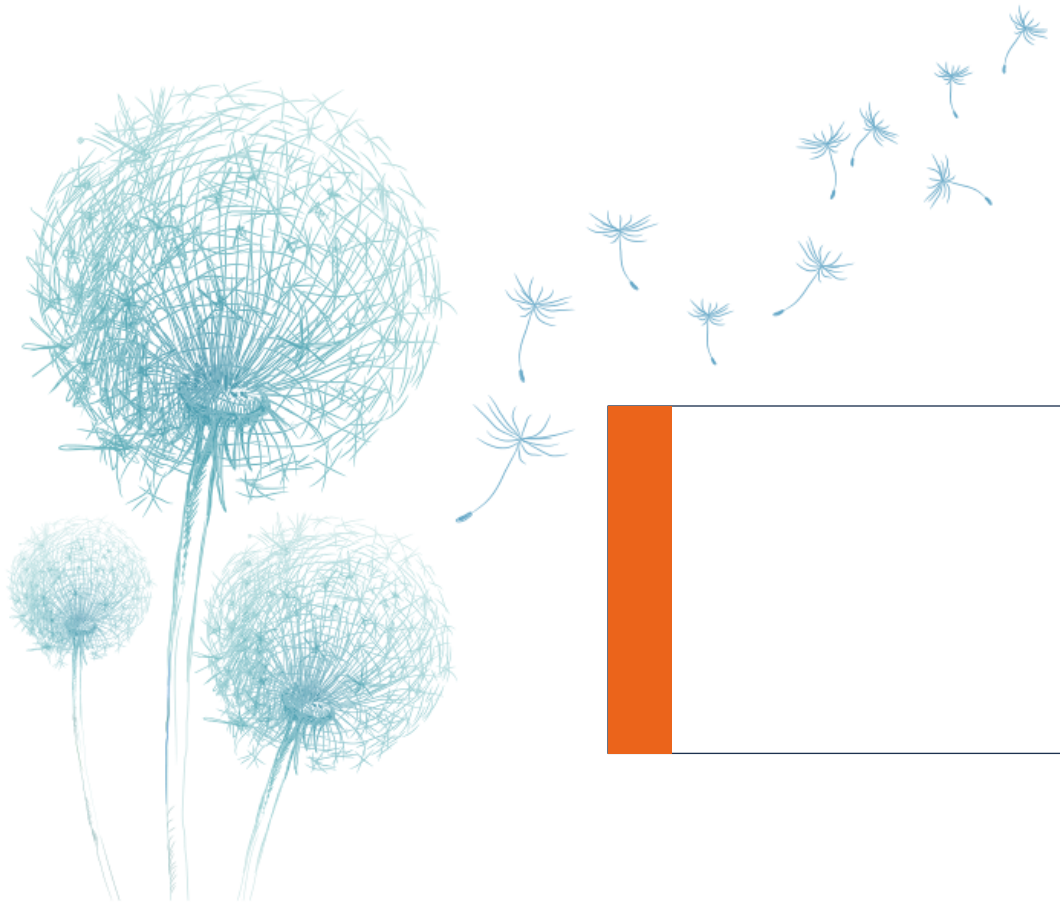
Leave-one-out cross-validation

- 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

Linear Regression

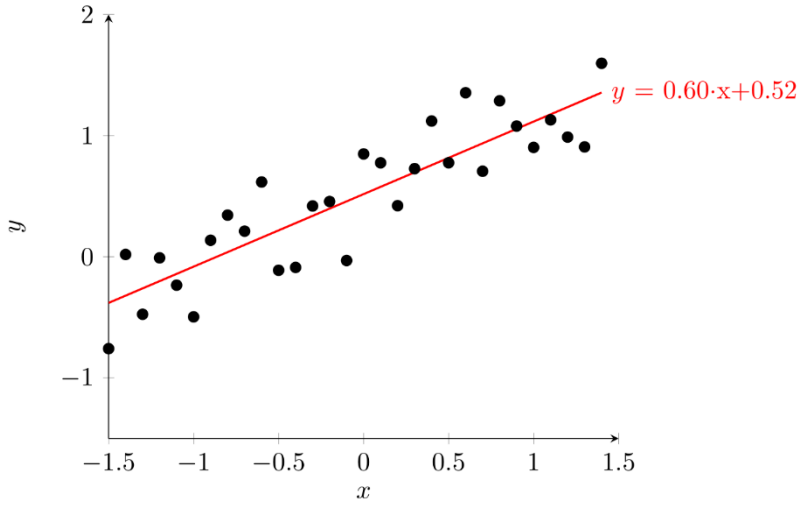
- **Model:**

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

  - Augmented weight vector and augmented feature vector

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

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



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

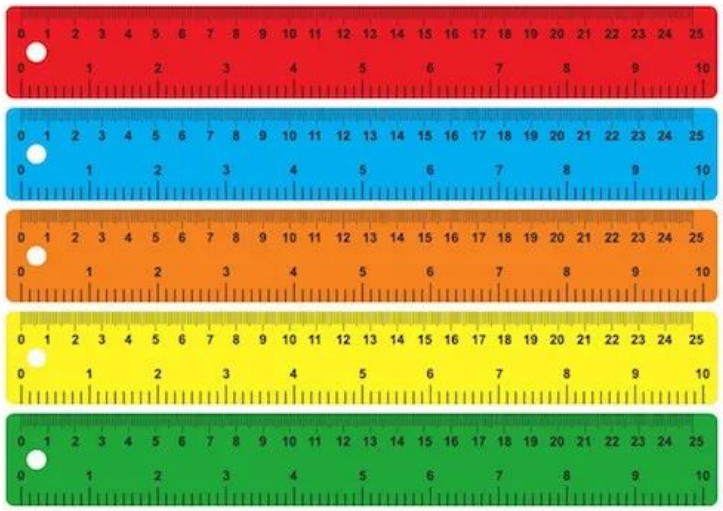
Optimization

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

Empirical risk minimization

Empirical risk minimization (least squares method)

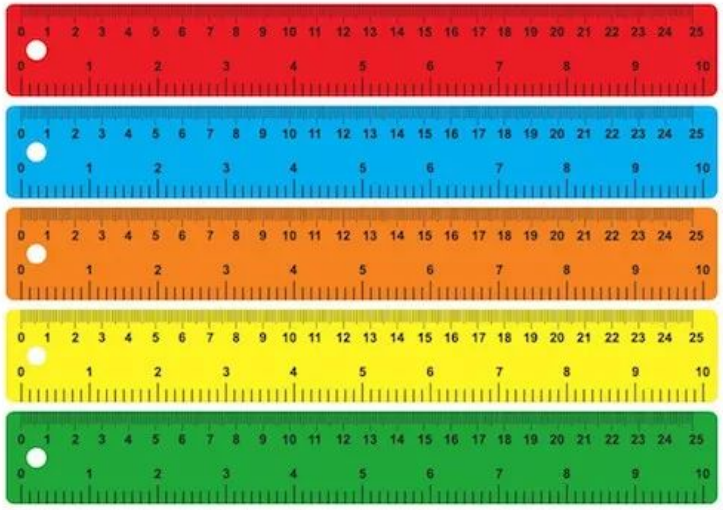
- Five rulers from five brands



	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

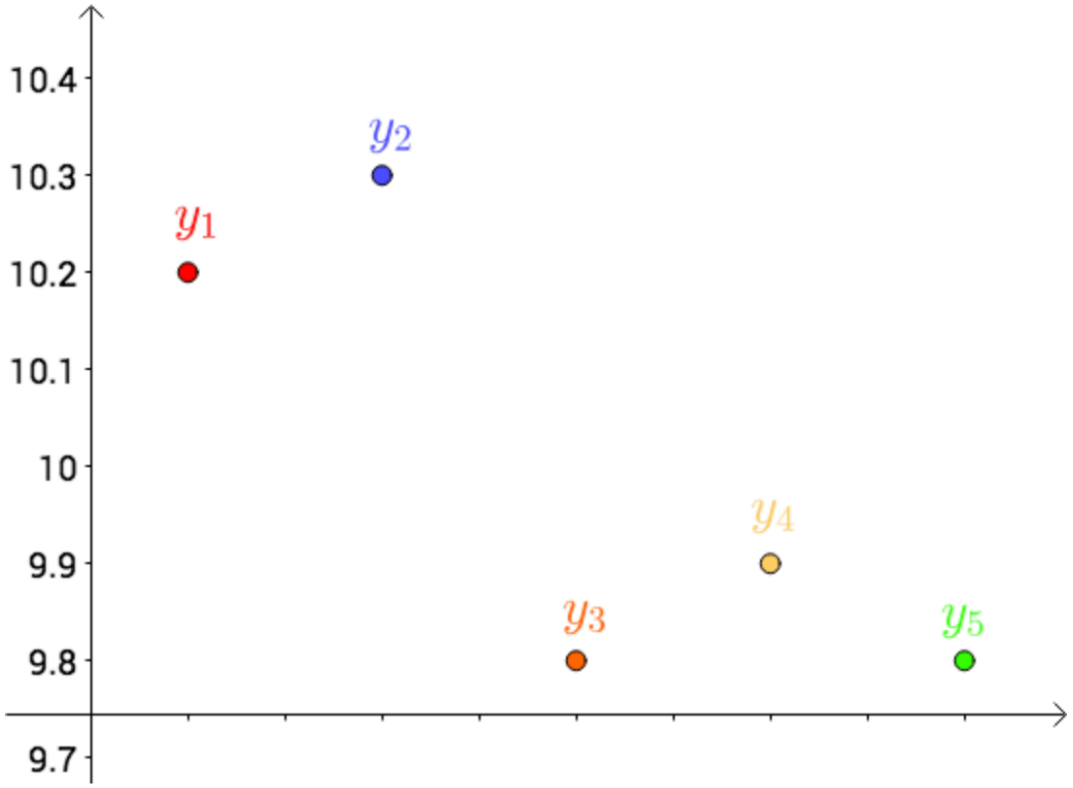
	Length
Red	10.2
Blue	10.3
Orange	9.8
Yellow	9.9
Green	9.8

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

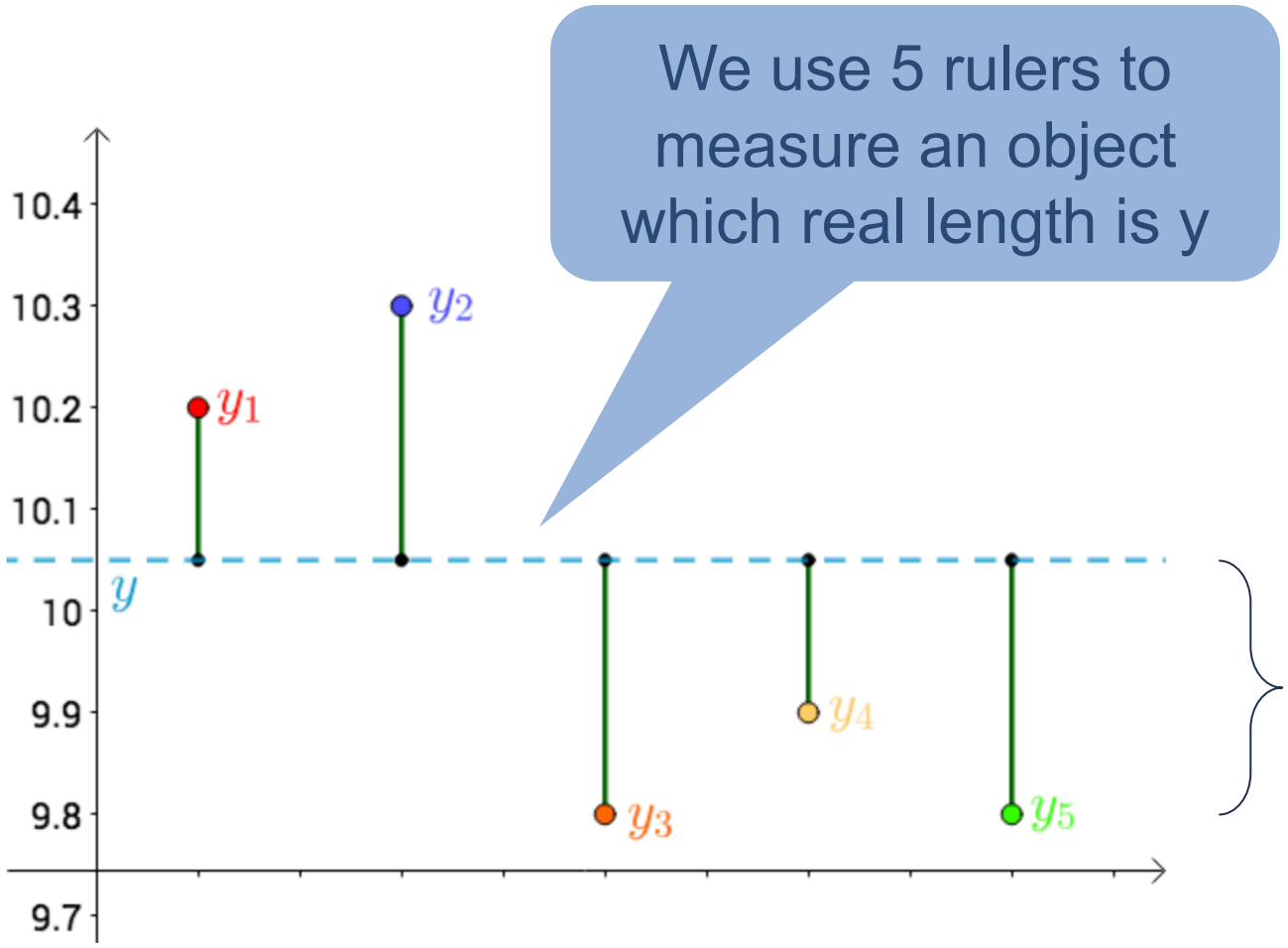
Empirical risk minimization (least squares method)

- Five rulers from five brands

	Length
Red	10.2
Blue	10.3
Orange	9.8
Yellow	9.9
Green	9.8



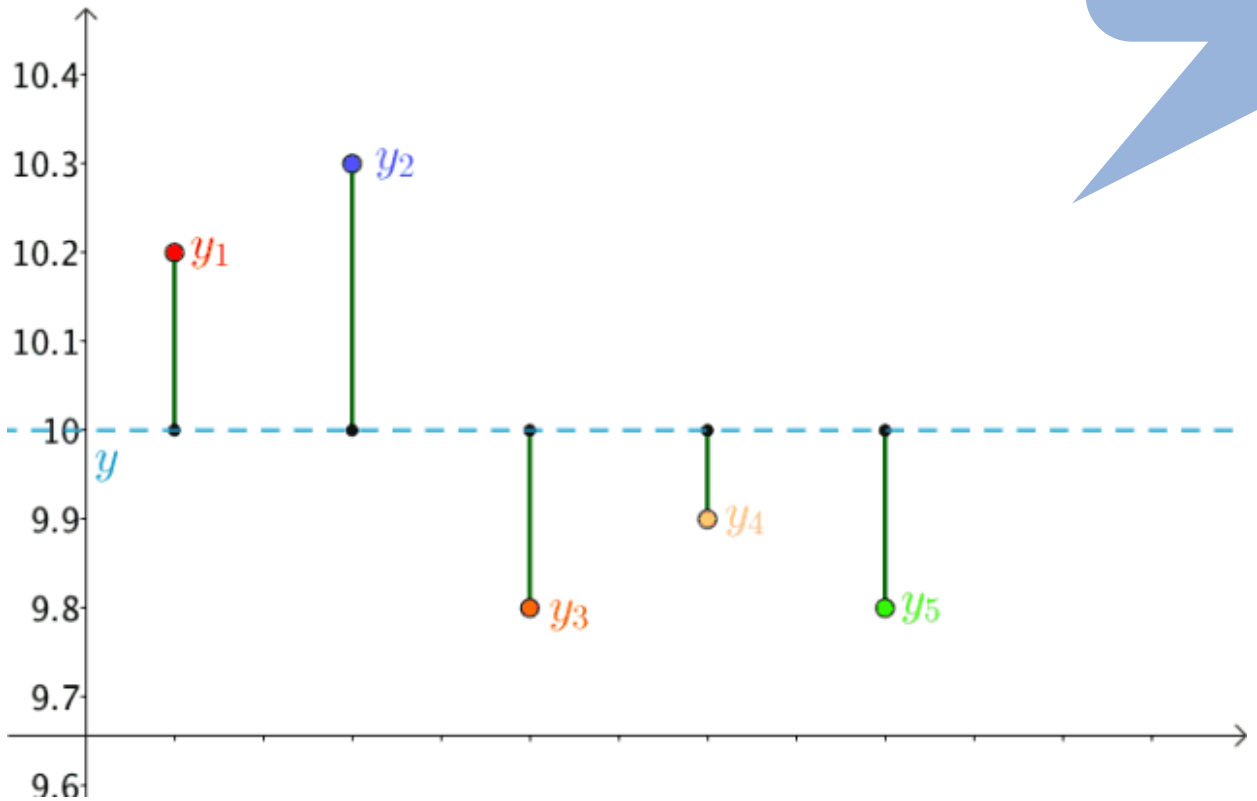
Empirical risk minimization (least squares method)



Error: $|y - y_i|$

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

Empirical risk minimization (least squares method)



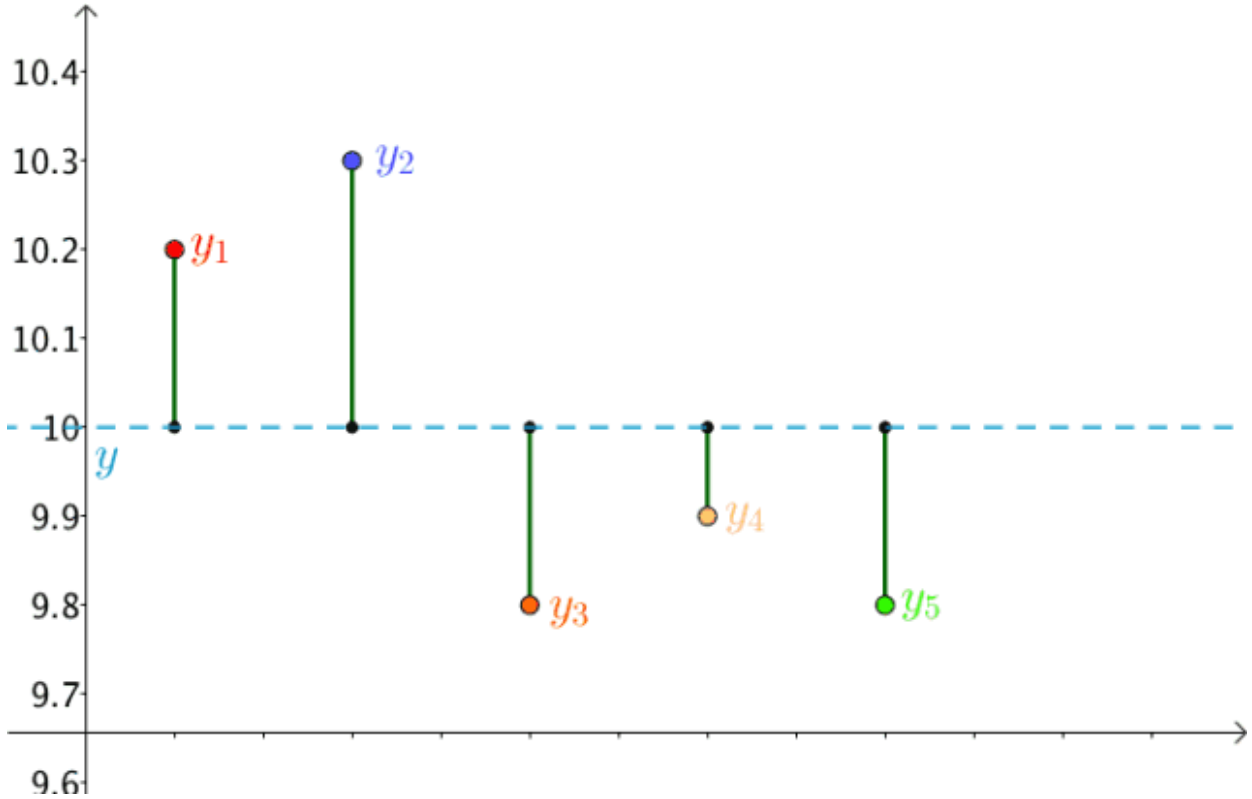
Different objects have various errors for the five rulers.

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

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

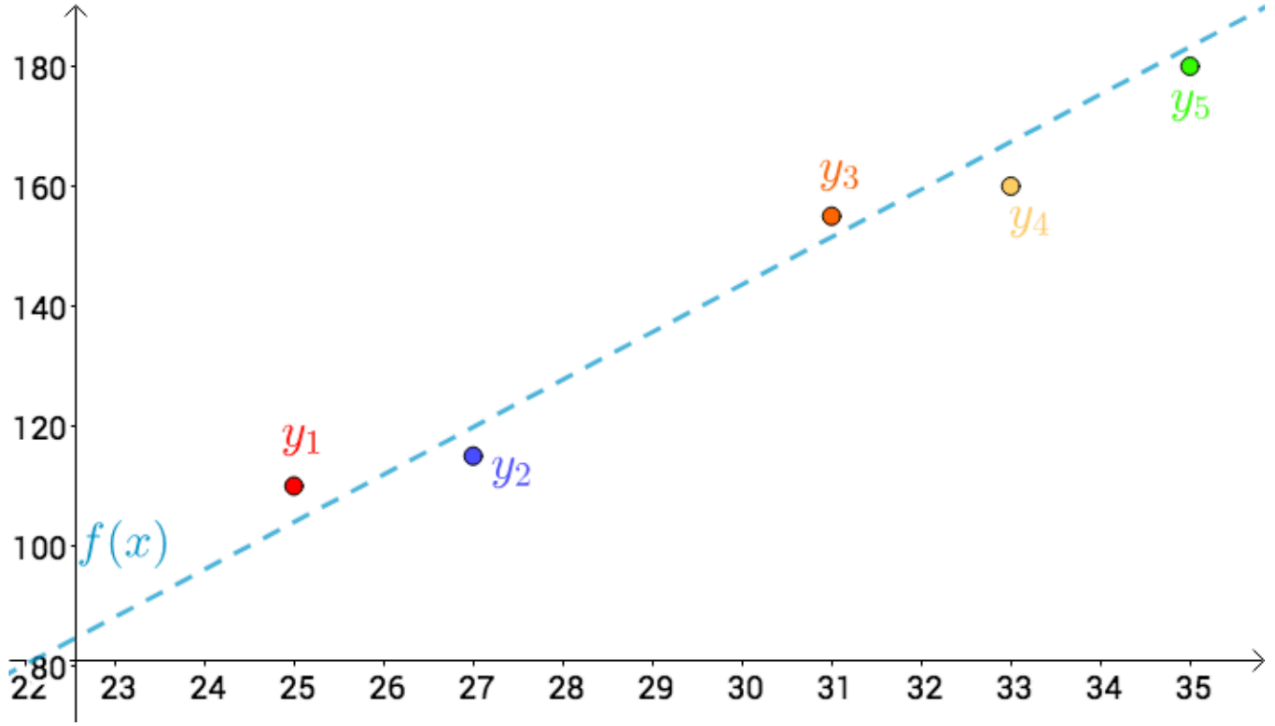
can make minimize S_{ϵ^2}

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

Empirical risk minimization (least squares method)

Temperature and ice cream sales:

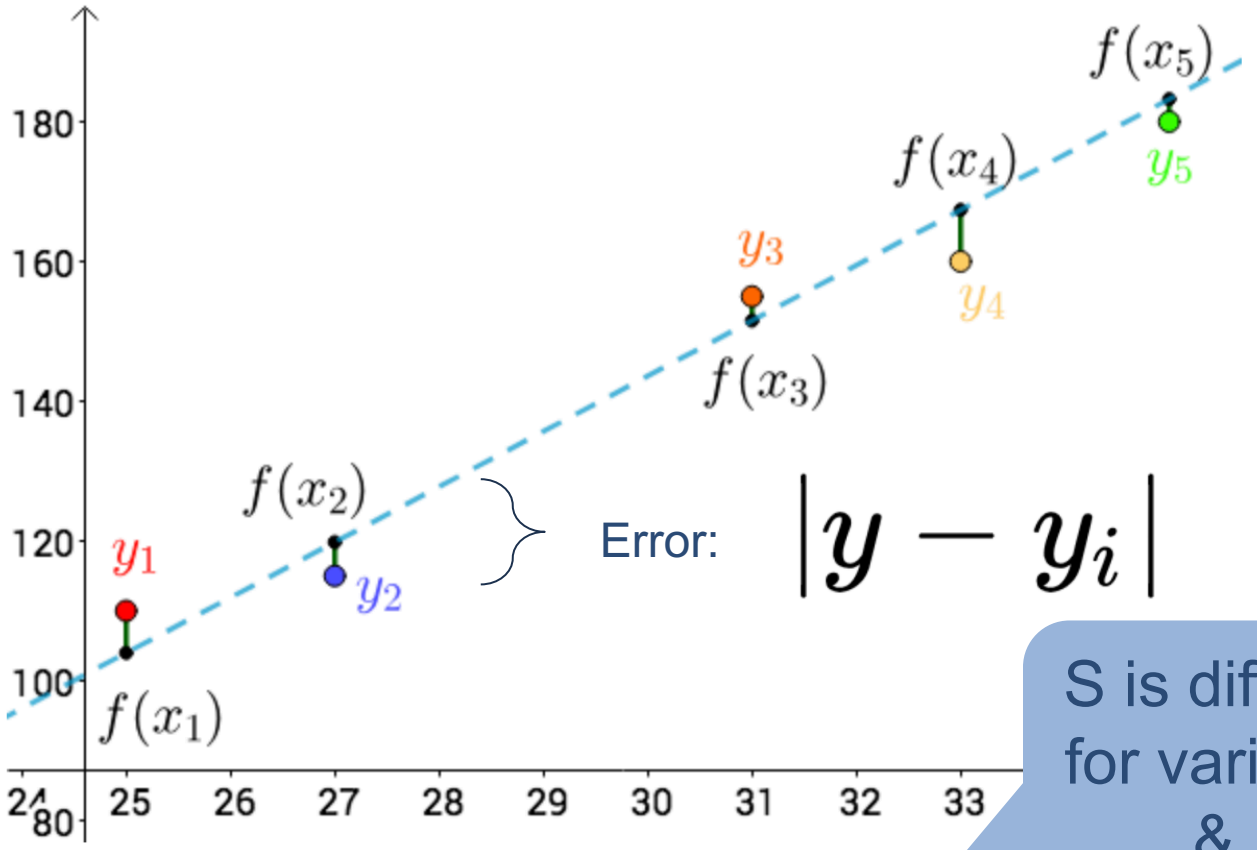
Temp (°C)	sales
25	110
27	115
31	155
33	160
35	180



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

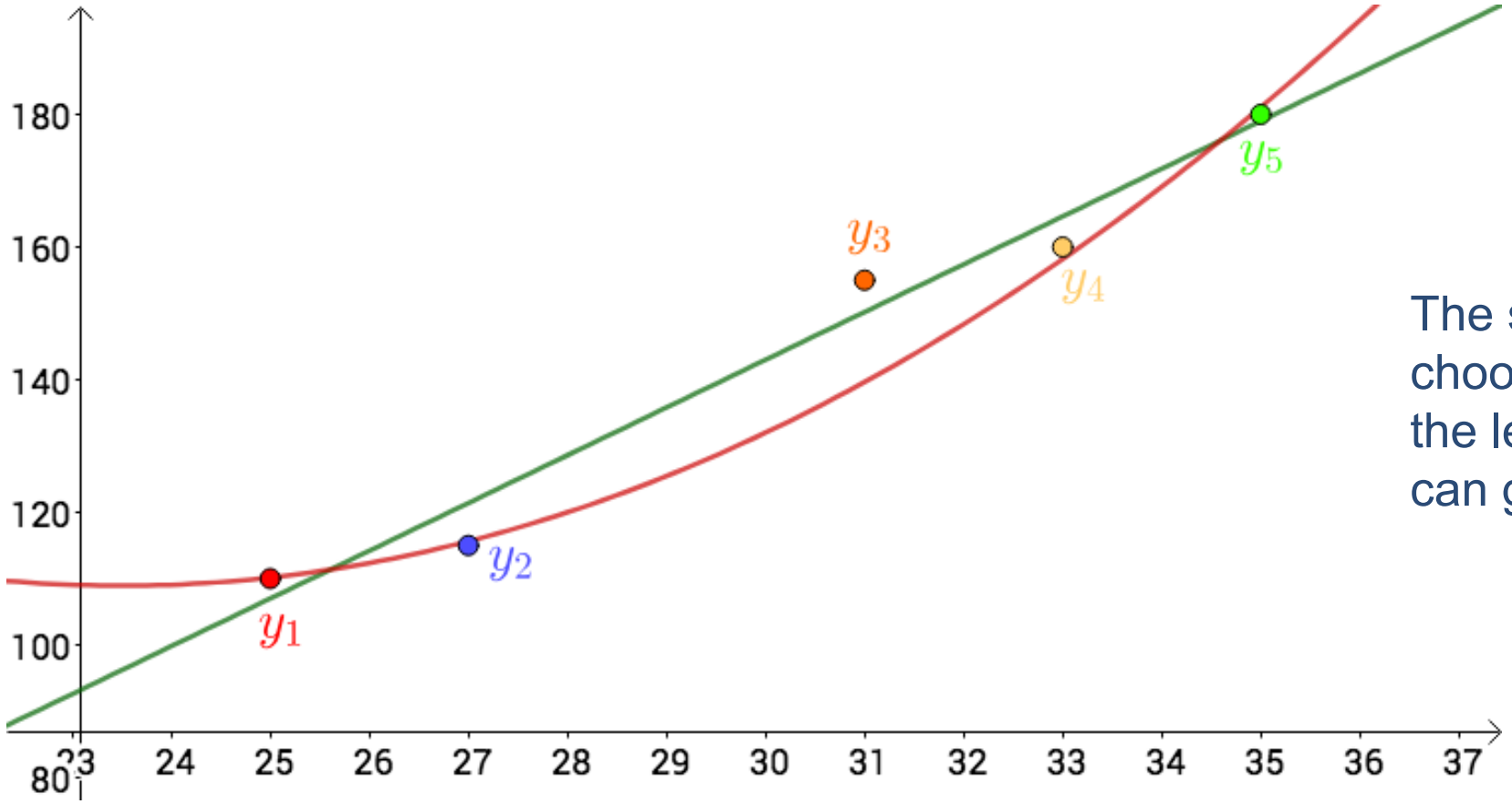
$$S_{\epsilon^2} = \sum (f(x_i) - y_i)^2 = \sum (ax_i + b - y_i)^2$$

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

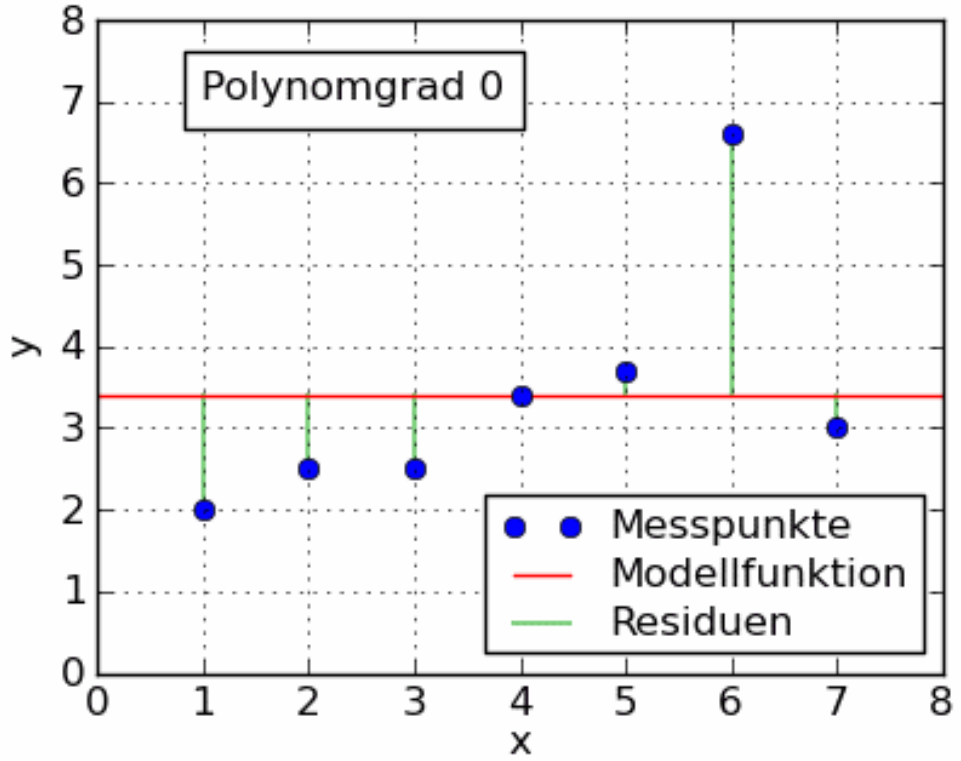
Empirical risk minimization (least squares method)

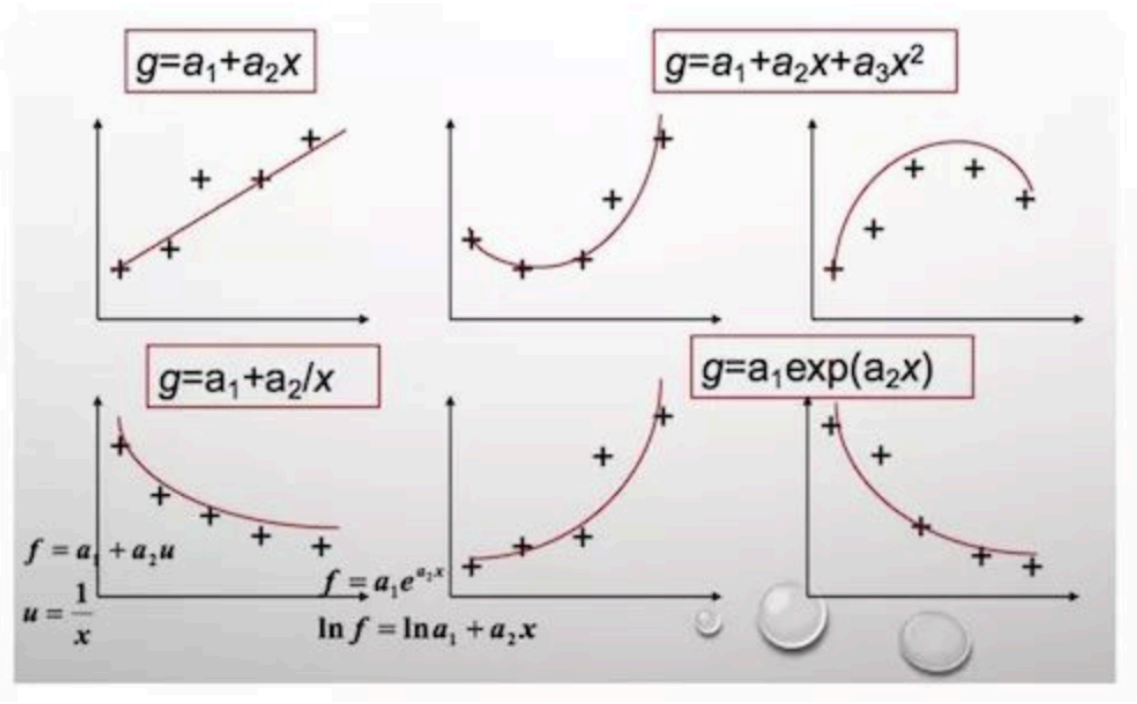


The same set of data, choose different f(x), through the least square method we can get different fitting curves

f(x) is called base function

Empirical risk minimization (least squares method)





Matrix calculus

- Partial derivative of a scalar with respect to a vector

$$\frac{\partial y}{\partial \boldsymbol{x}} = [\frac{\partial y}{\partial x_1}, \cdots, \frac{\partial y}{\partial x_M}]^{\mathrm{T}}$$

- Partial derivative of vector with respect to vector

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

- Vector functions and their derivatives

$$\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{x}} = \boldsymbol{I},$$
$$\frac{\partial \boldsymbol{A}\boldsymbol{x}}{\partial \boldsymbol{x}} = \boldsymbol{A}^{\mathrm{T}},$$
$$\frac{\partial \boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}}{\partial \boldsymbol{x}} = \boldsymbol{A}$$

Empirical risk minimization (least squares method)

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

- Study criteria

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

- Model

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

Maximum likelihood estimation

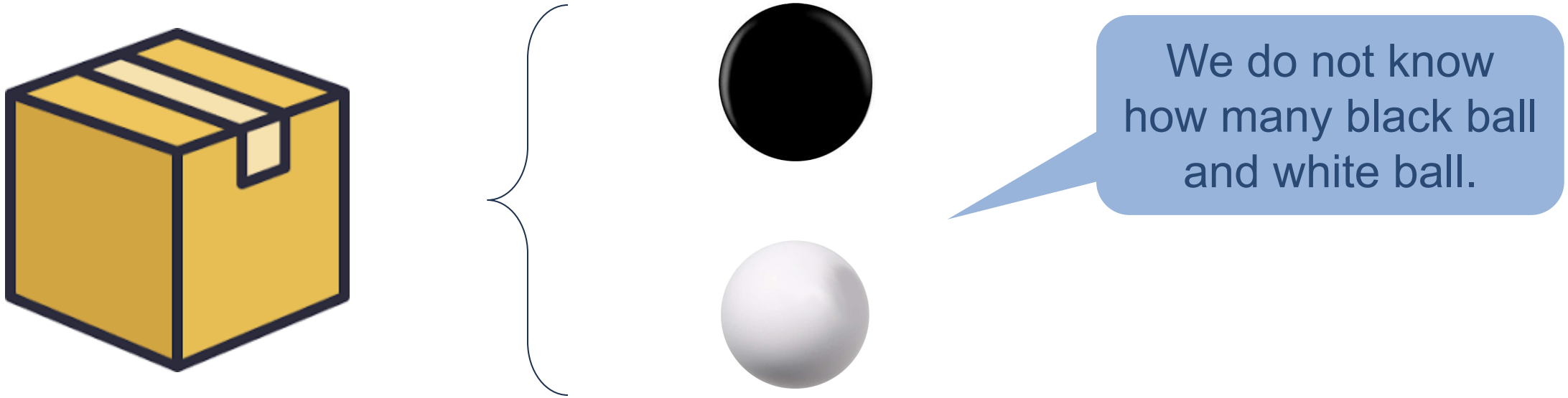
Probability vs. Likelihood

- **Likelihood** is the **probability** that an event that has already occurred would yield a specific outcome.

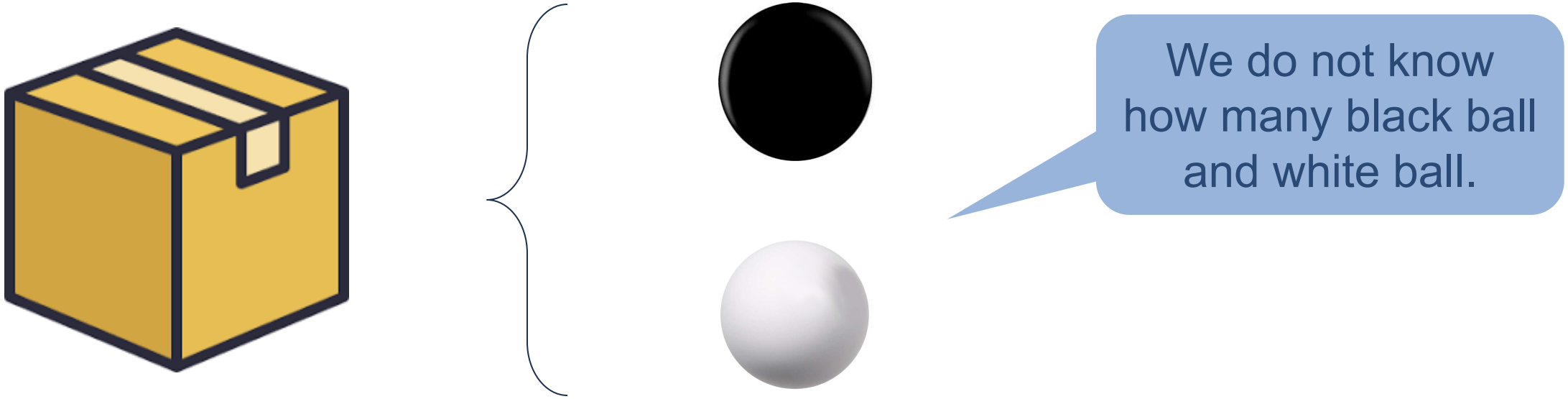
- **Probability** refers to the occurrence of future events, while a **likelihood** refers to past events with known outcomes.

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



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

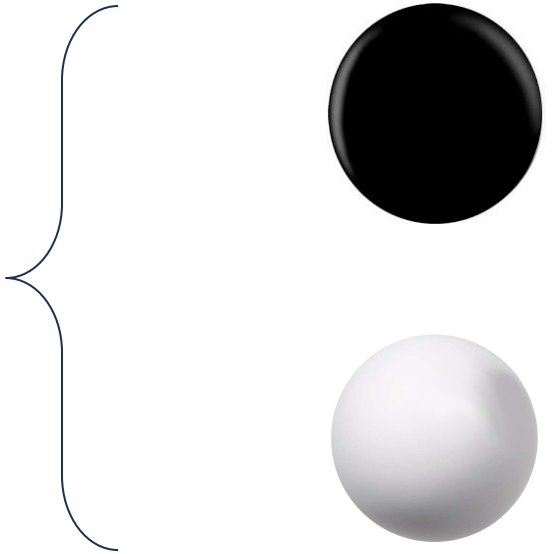
Maximum likelihood estimation

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

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

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

➔ 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

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

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

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

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

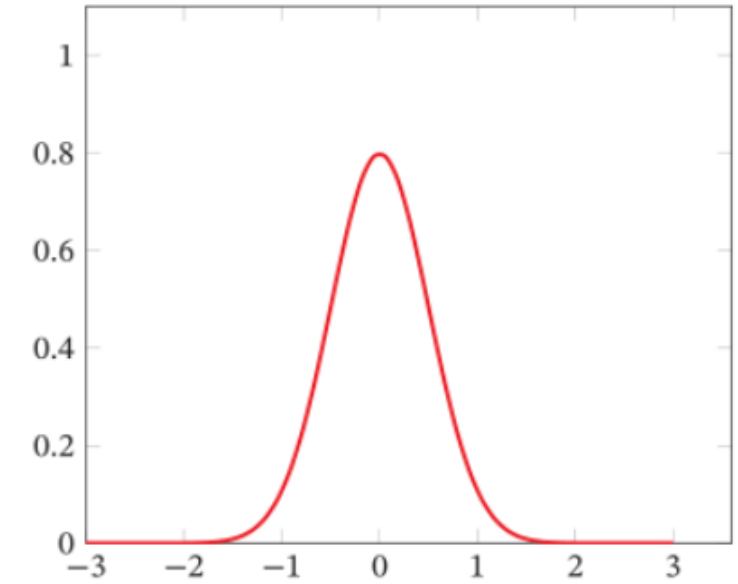
Some basic concepts about probability

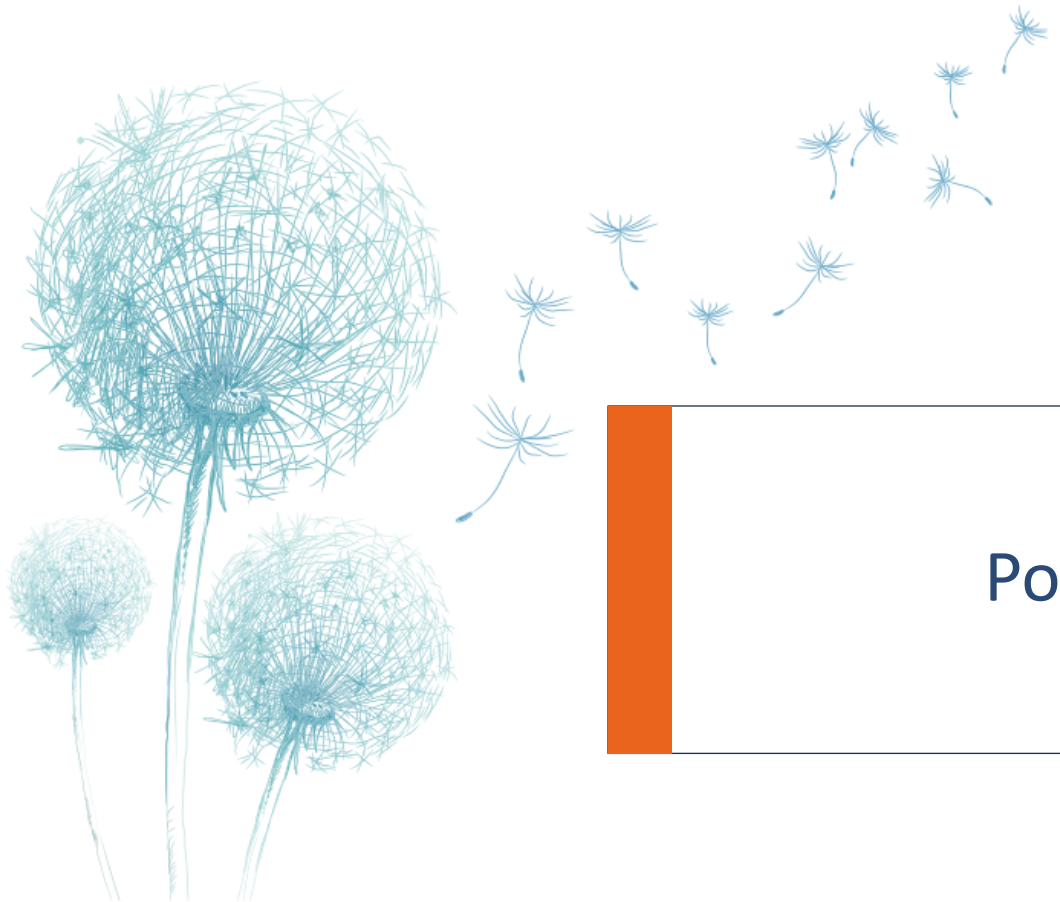
Gaussian Distribution (normal distribution)

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

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

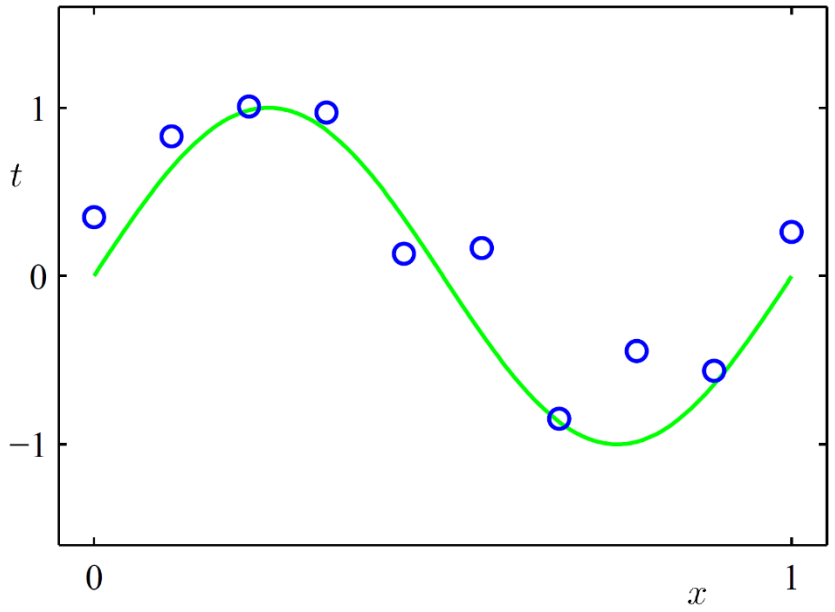
Expectation is μ
Variance is σ^2





Polynomial regression

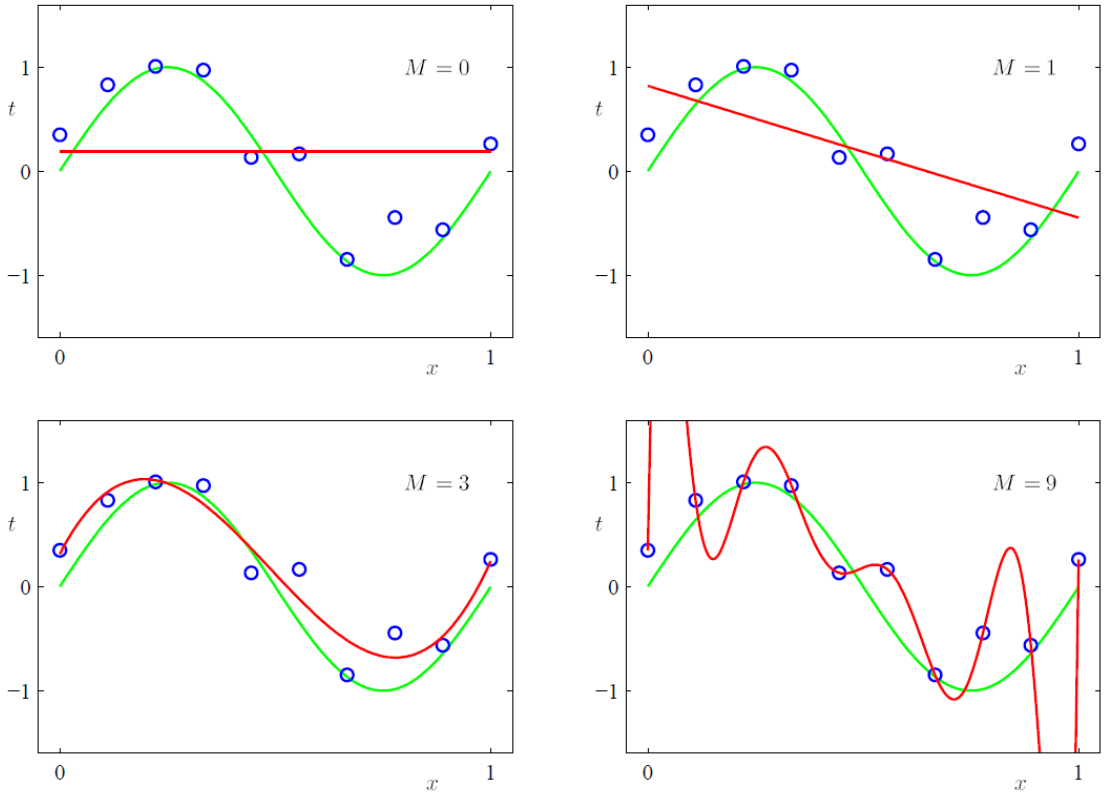
Polynomial Curve Fitting



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

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

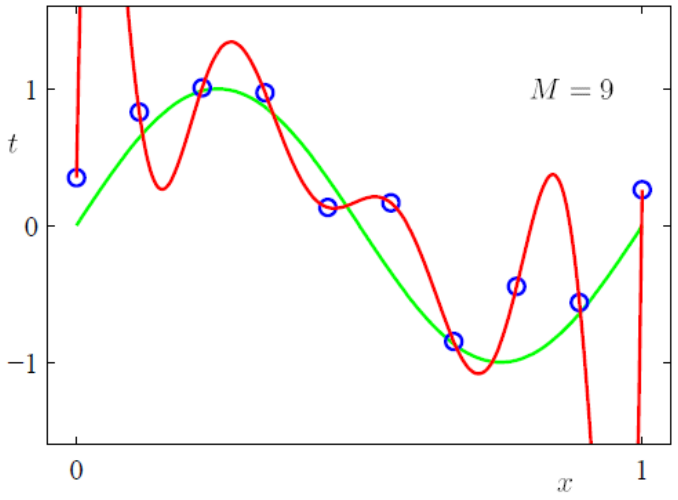
Which Degree of Polynomial?



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

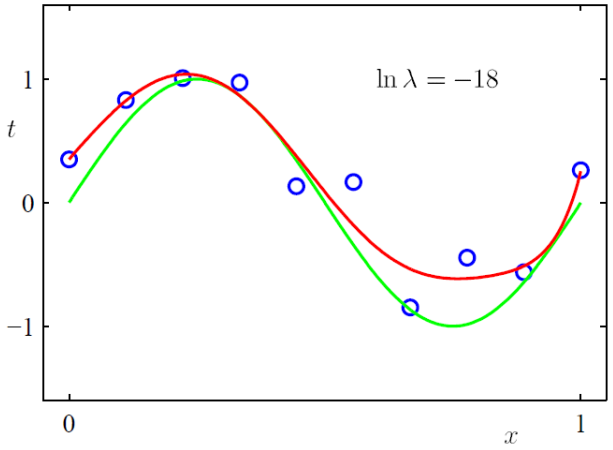
As order of polynomial M increases, so do coefficient magnitudes!

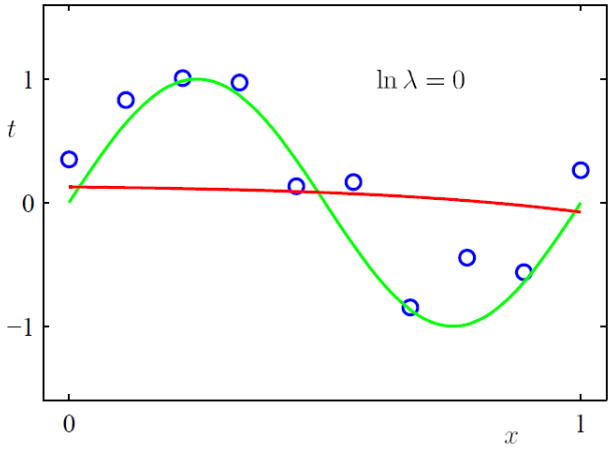
$$\tilde{E}(\boldsymbol{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \boldsymbol{w}) - t_n\}^2 + \frac{\lambda}{2}||\boldsymbol{w}||^2$$

Penalize large coefficients

Controlling Overfitting: Regularization

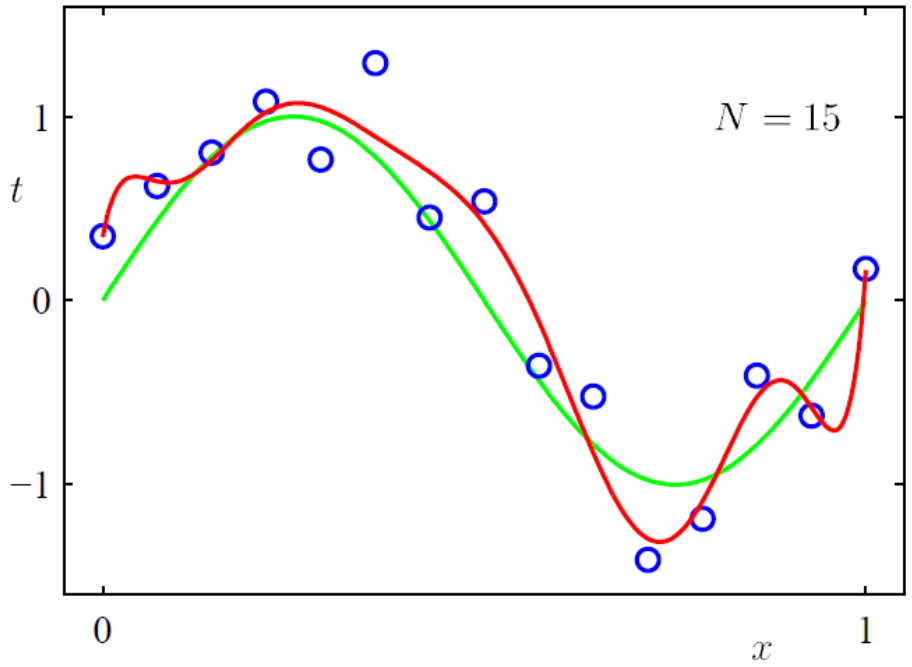
$$\tilde{E}(\boldsymbol{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \boldsymbol{w}) - t_n\}^2 + \frac{\lambda}{2}||\boldsymbol{w}||^2$$

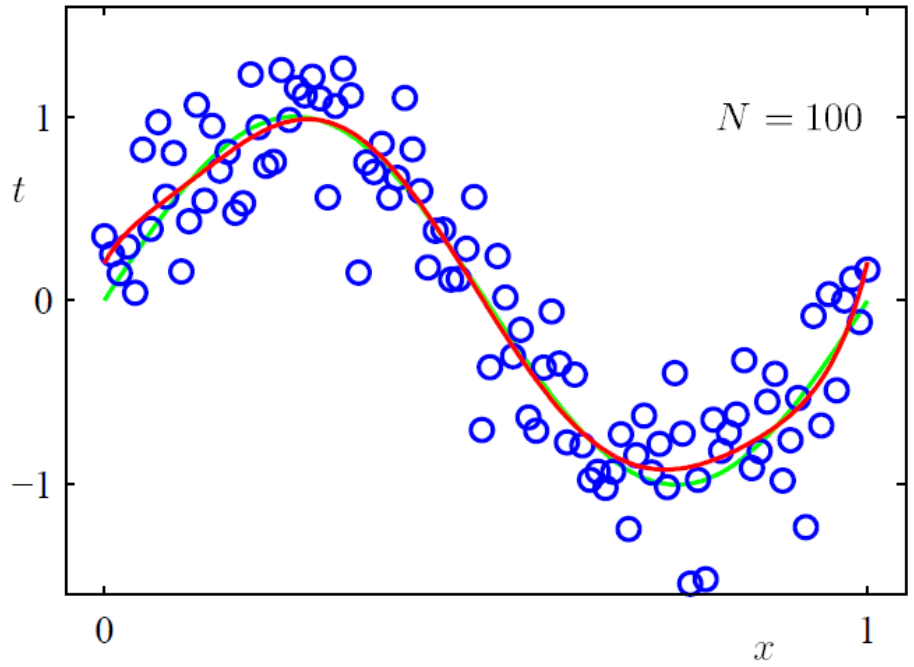




	$\ln\lambda = -\infty$	$\ln\lambda = -18$	$\ln\lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18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

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

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

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

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

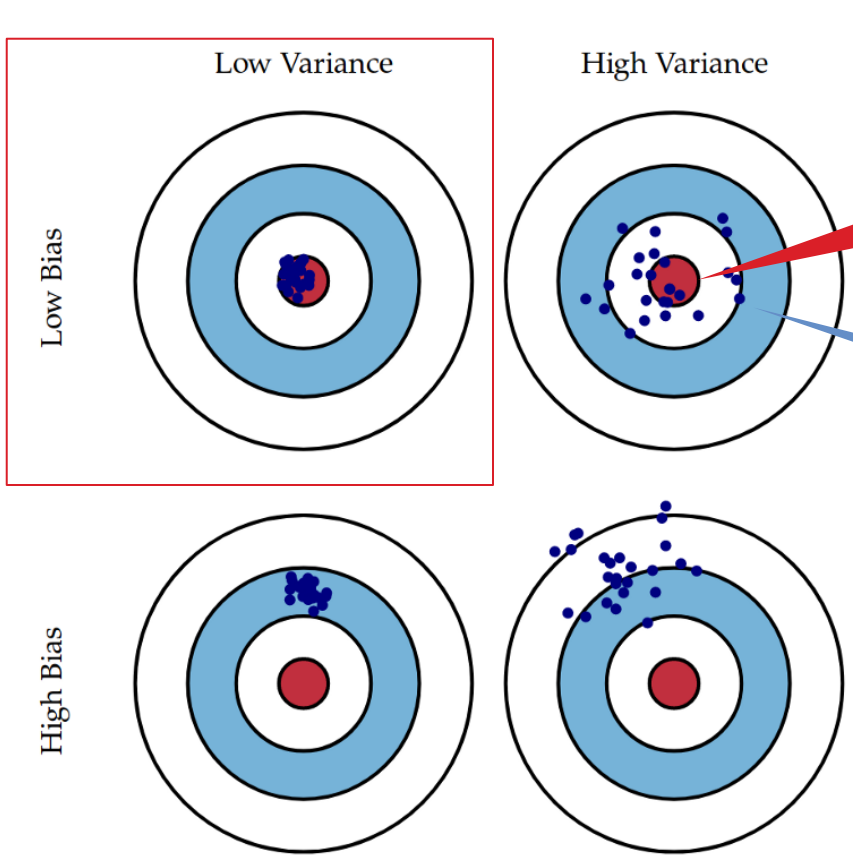
Model selection: bias and variance

- 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

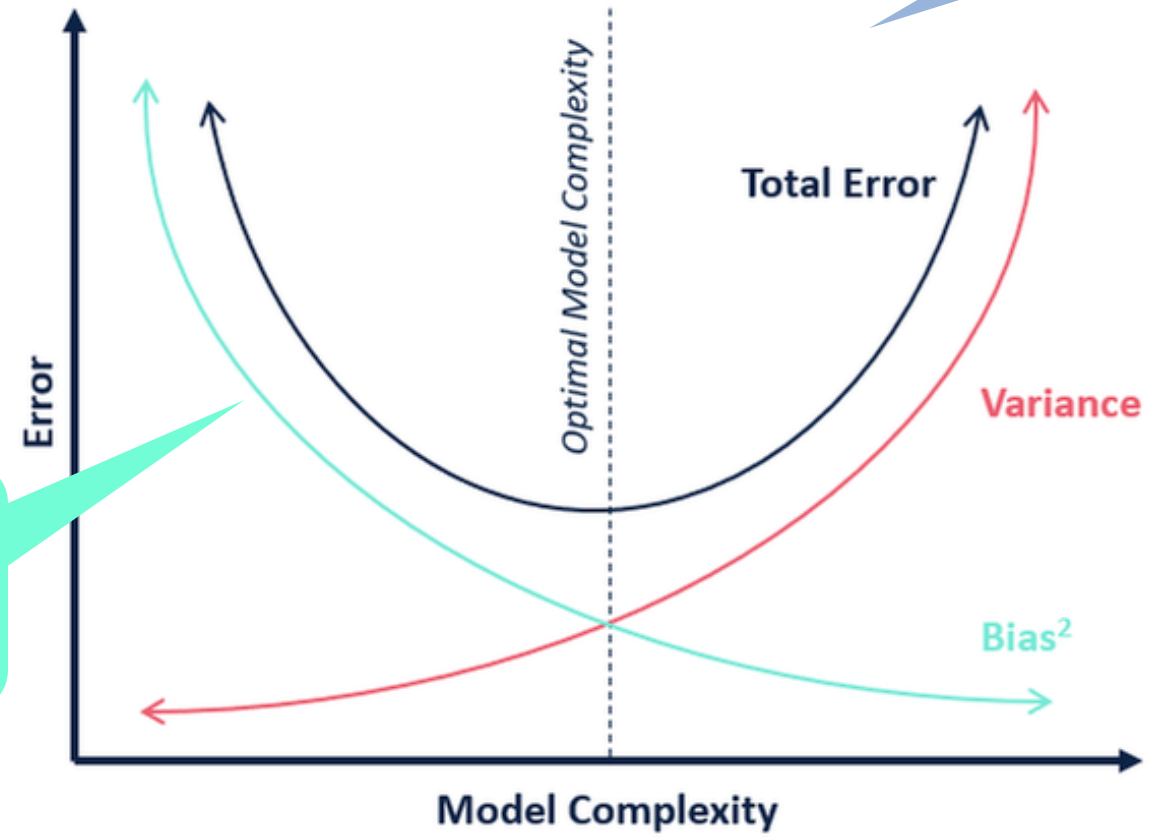


The perfect correct prediction value of the learning algorithm

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



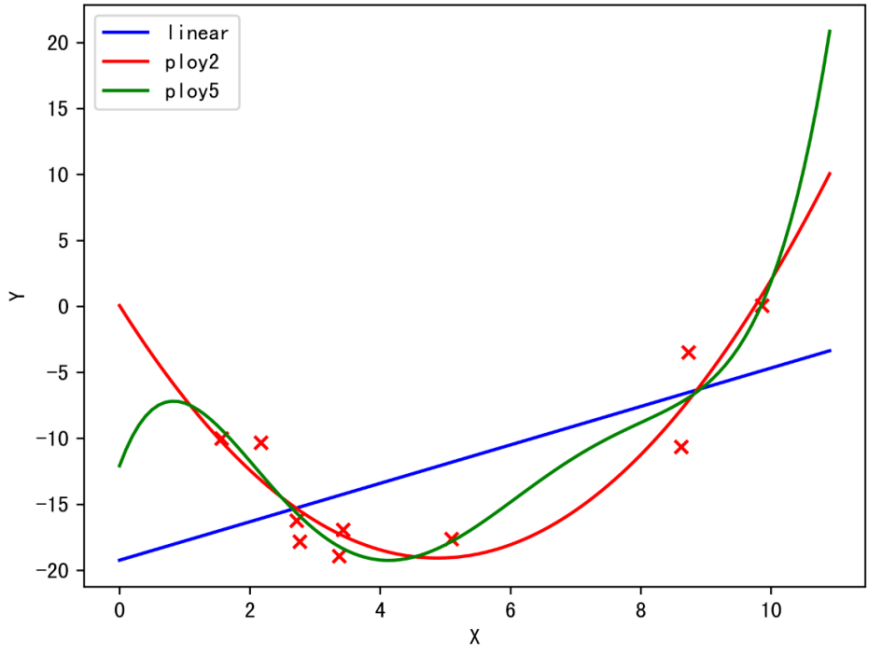
A model with too low complexity cannot fit the training data well, but it is easy to cause large bias

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.

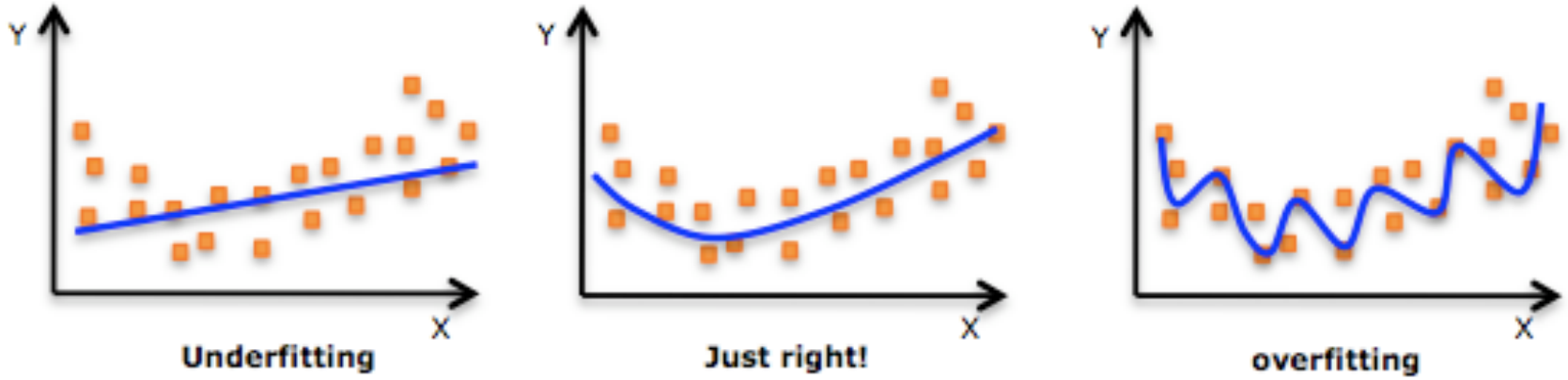
  - Too small data is always bad

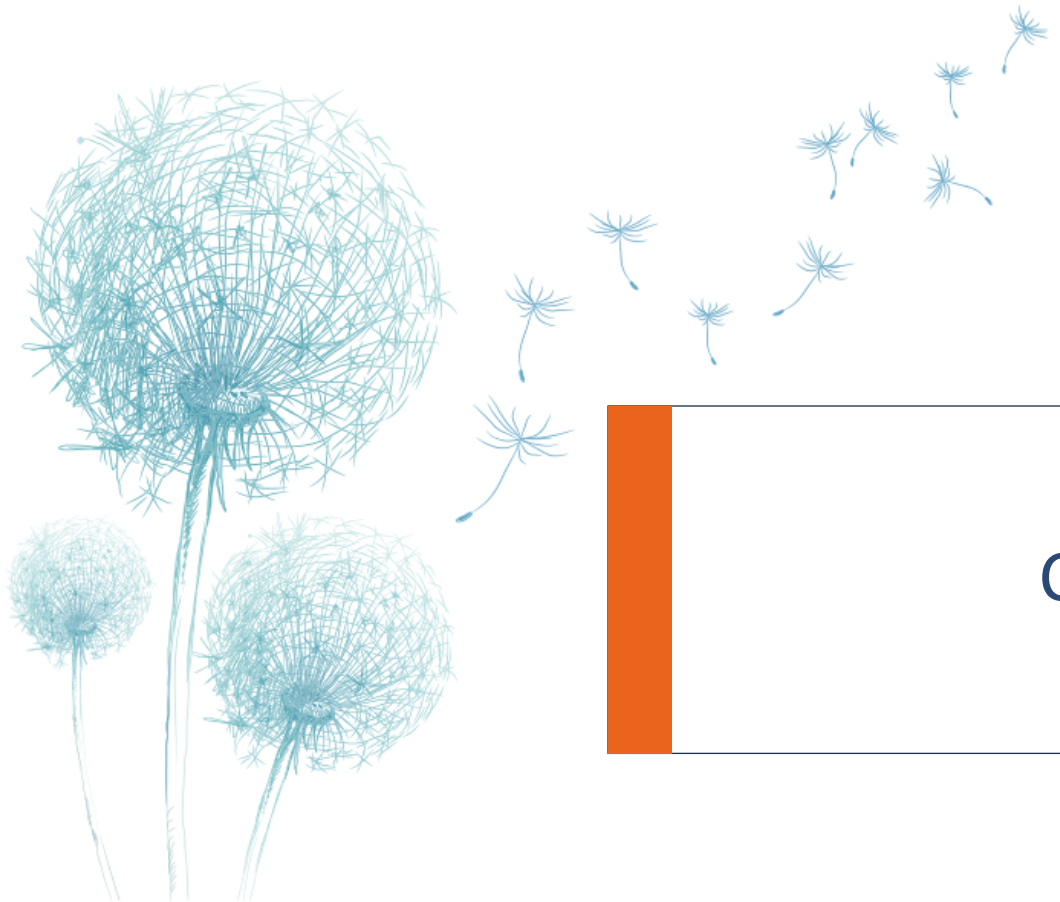
  - The model complexity is too high, the variance is large, and the models trained on different data sets vary greatly

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





Common theorem

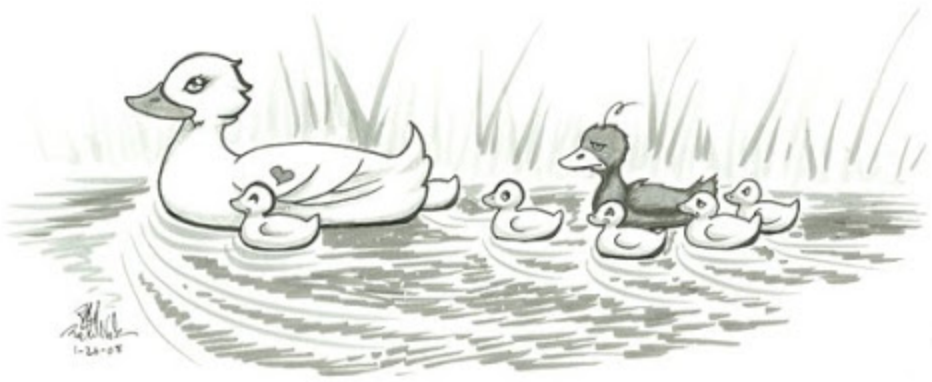
Common theorem

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

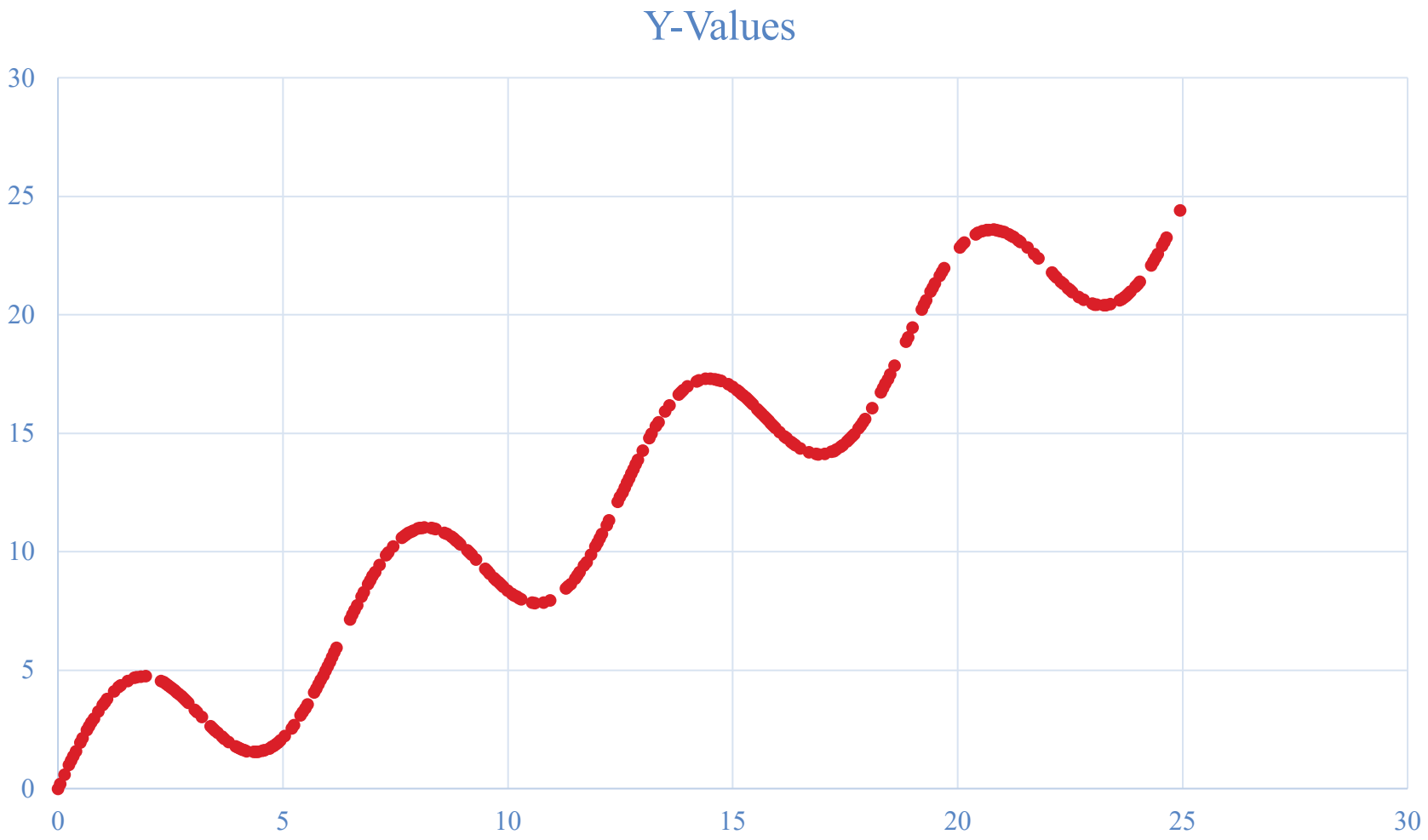
Inductive Bias

- 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

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



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)
```

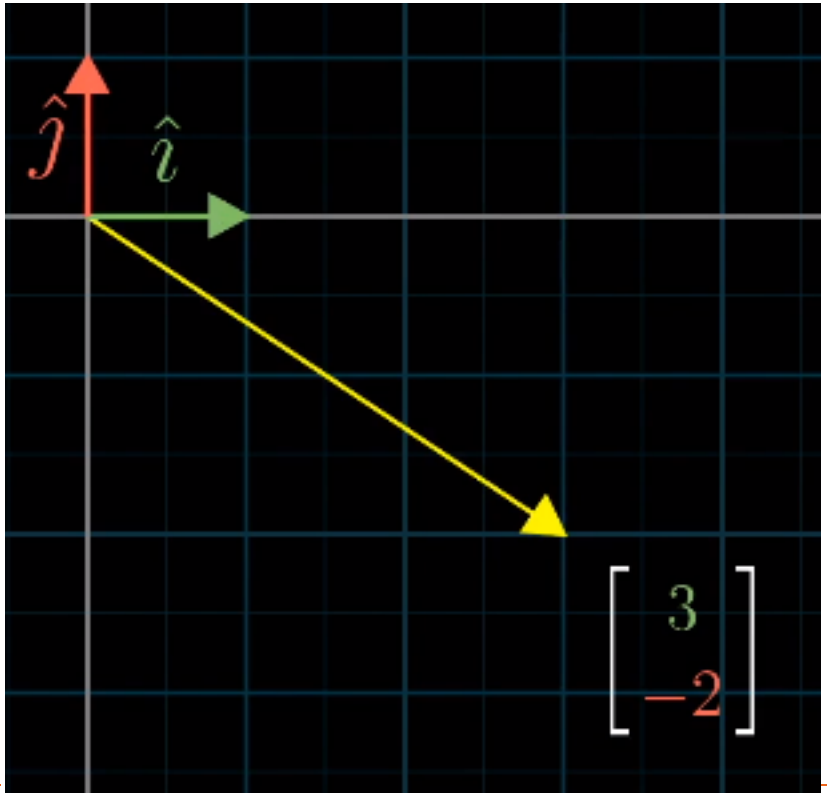
Basis function

```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
```

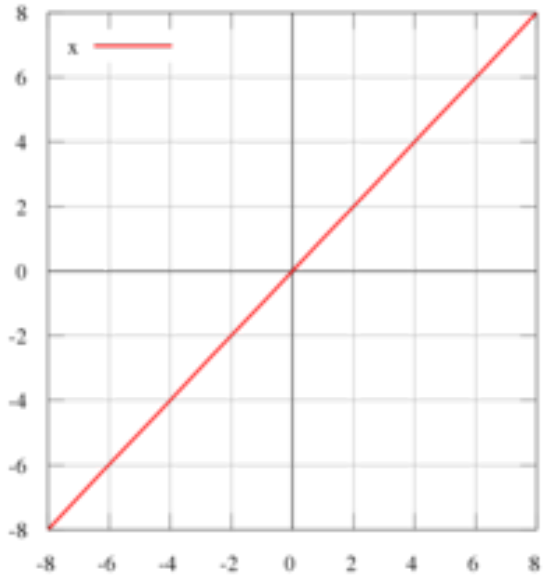
```python
"""-------------Train model--------------- """
def main(x_train, y_train):
    basis_func = gaussian_basis
    phi0 = np.expand_dims(np.ones_like(x_train), axis=1)
    phi1 = basis_func(x_train)
    phi = np.concatenate([phi0, phi1], axis=1)
    w = np.dot(np.linalg.pinv(phi), y_train)

    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

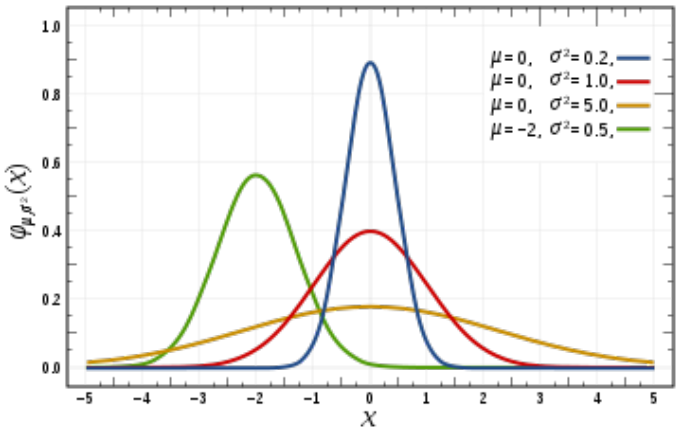


$f(x) = x$ for all elements x in M

Identity function

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

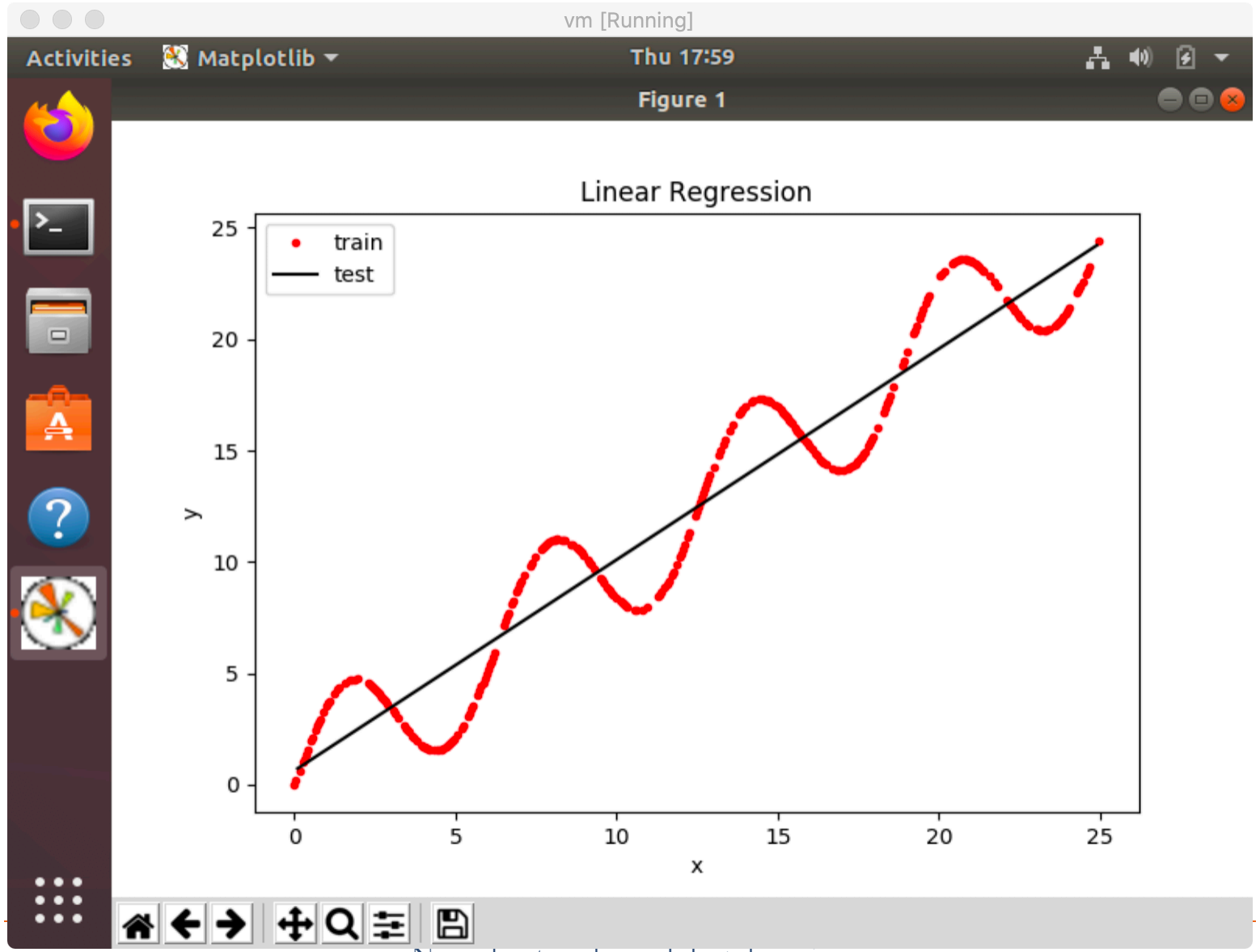
Polynomial function



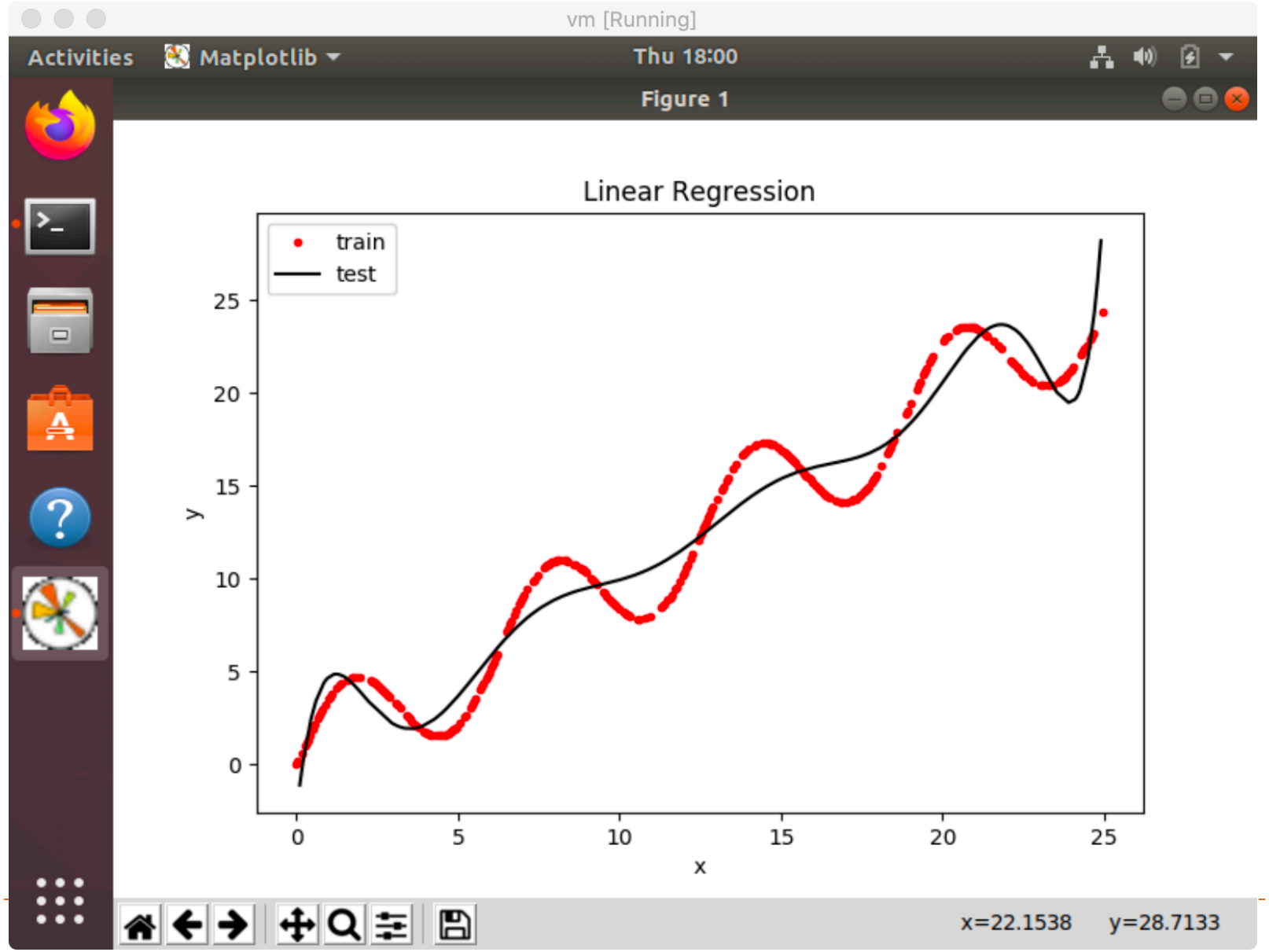
$$f(x) = a \cdot \exp\left(-\frac{(x-b)^2}{2c^2}\right)$$

Gaussian function

Execution: identity_basis



Execution: multinomial_basis



Execution: gaussian_basis