

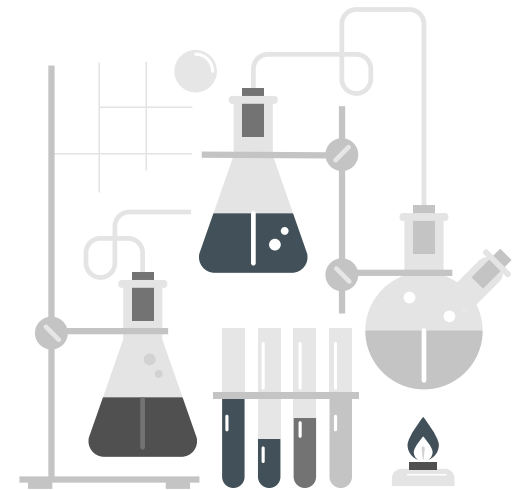
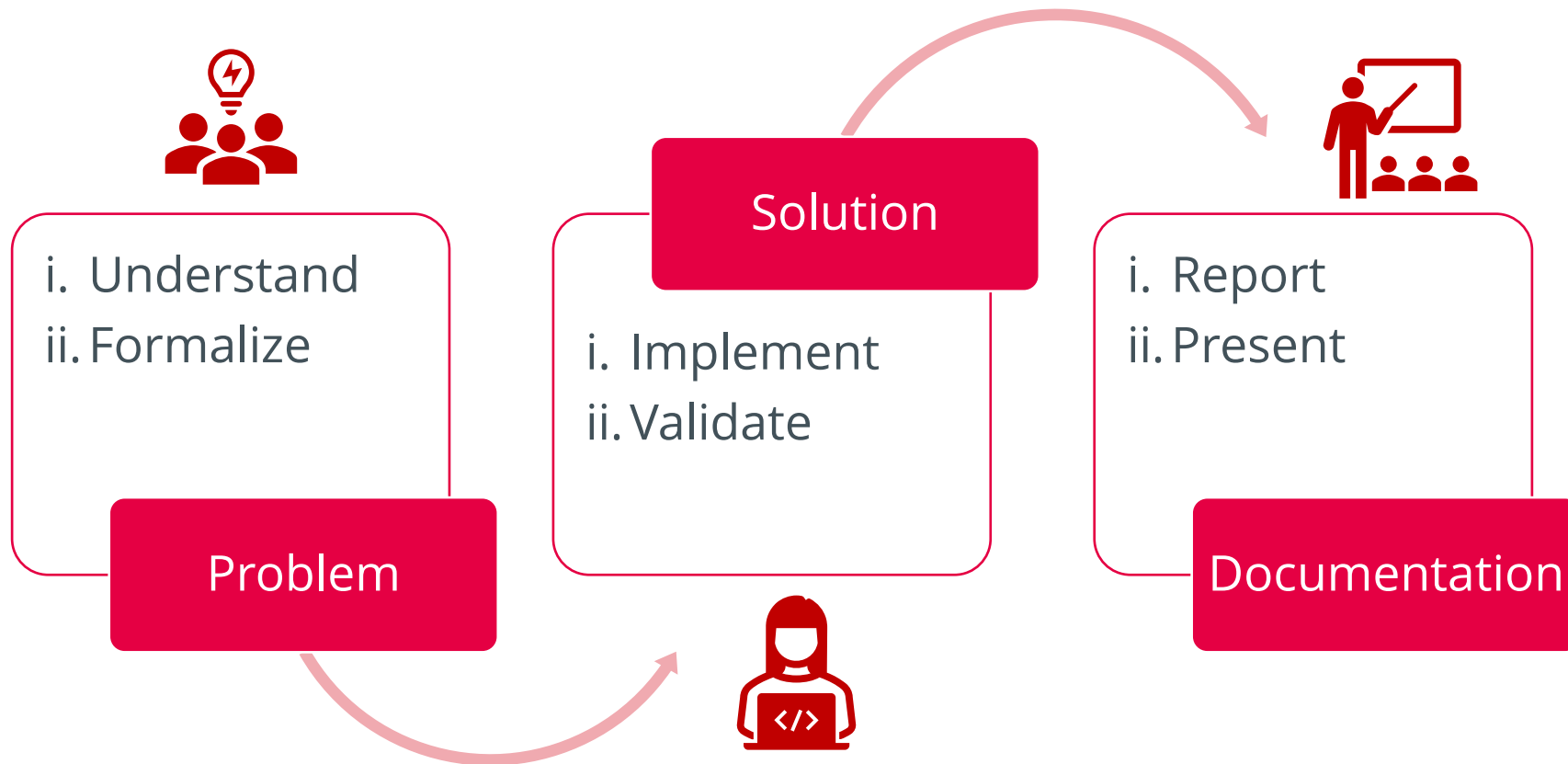


TECHNISCHE HOCHSCHULE  
OSTWESTFALEN-LIPPE  
UNIVERSITY OF  
APPLIED SCIENCES  
AND ARTS

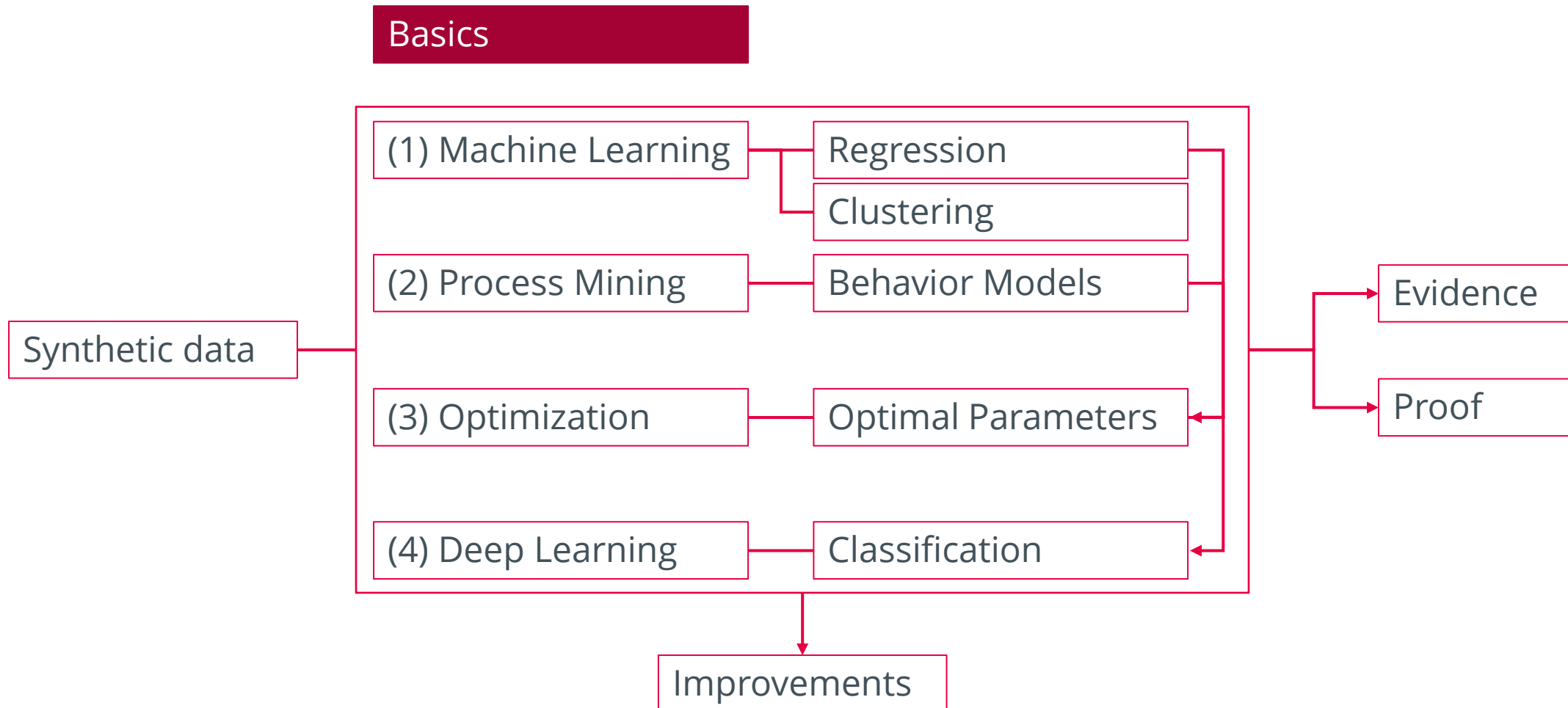
# Welcome

**to Advanced Topics in Algorithms**

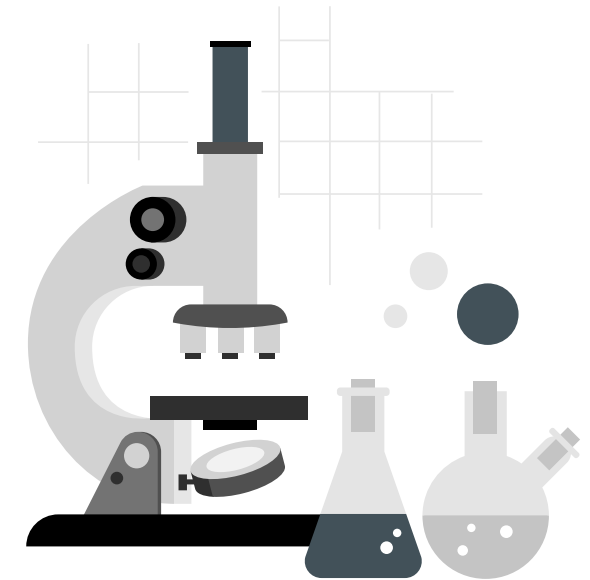
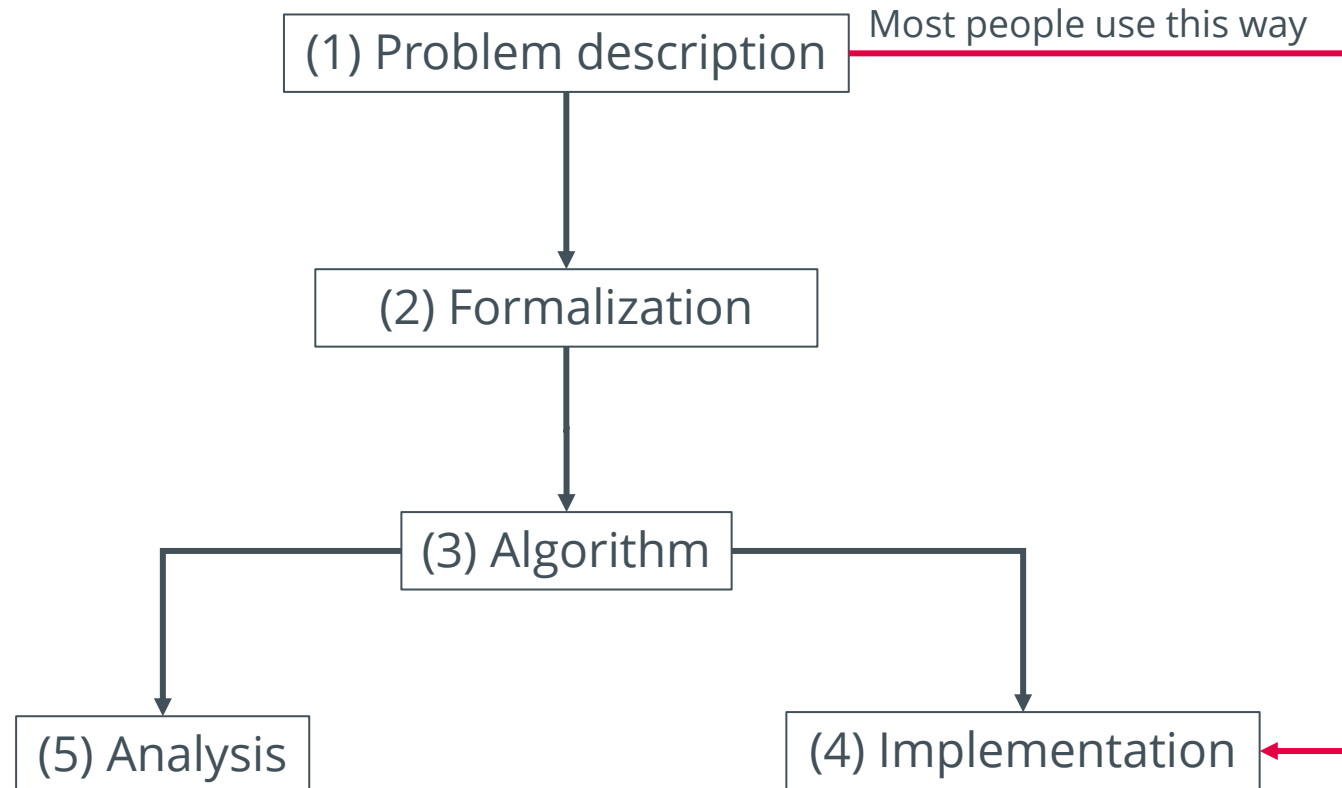
# Learning Objective: Become a Scientist



# Summary: Advanced Topics in Algorithms

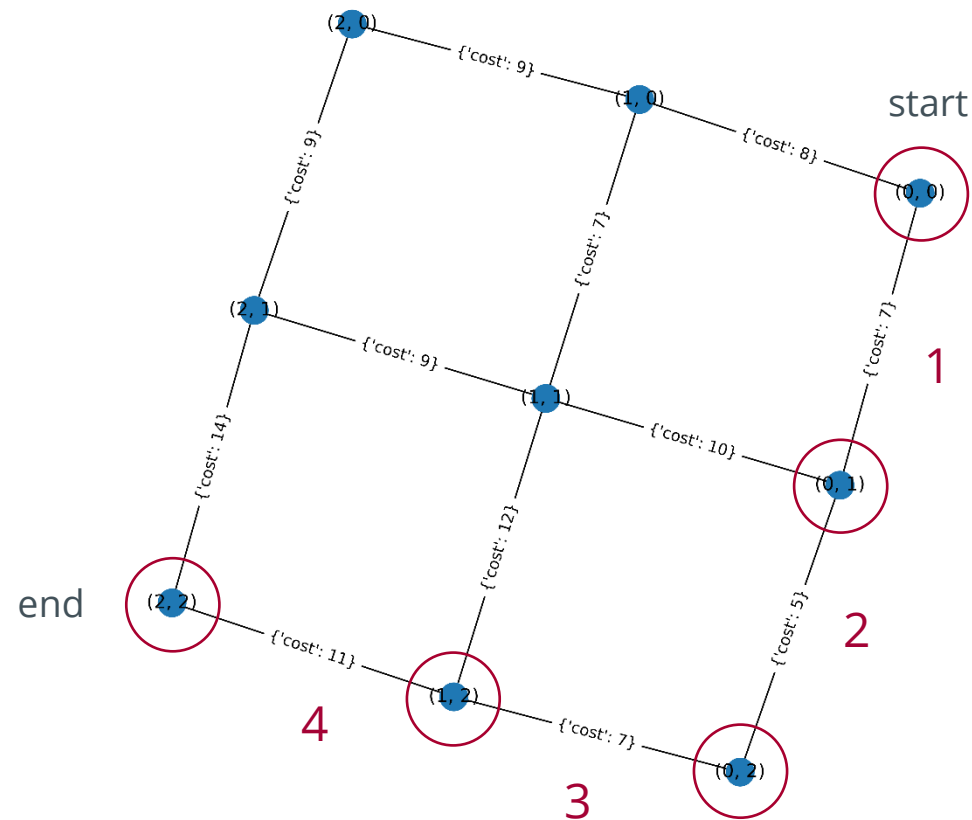


# Concept: Algorithm Design



It's not rocket science

# Algorithm Design: A\*-Algorithm



cost of the path from the start node

$$f(n) = g(n) + h(n)$$

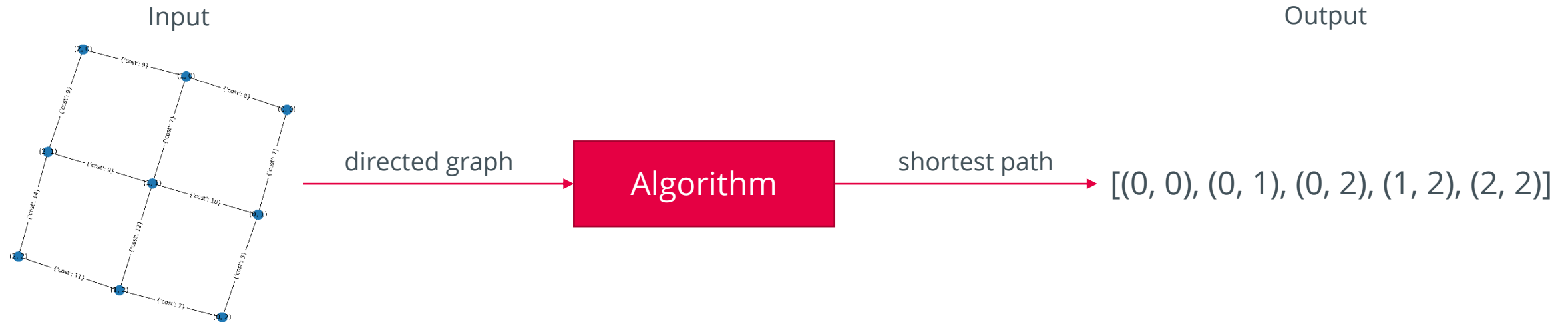
next node

next	end	Euclidean distance
(1, 0)	(2, 2)	2.2360
(0, 1)	(2, 2)	2.2360
(0, 0)	(2, 2)	2.8284
(1, 1)	(2, 2)	1.4142
(0, 2)	(2, 2)	2.0
(2, 0)	(2, 2)	2.0
(1, 2)	(2, 2)	1.0
(2, 1)	(2, 2)	1.0
(2, 2)	(2, 2)	0.0

# Algorithm Design: (1) Problem Description

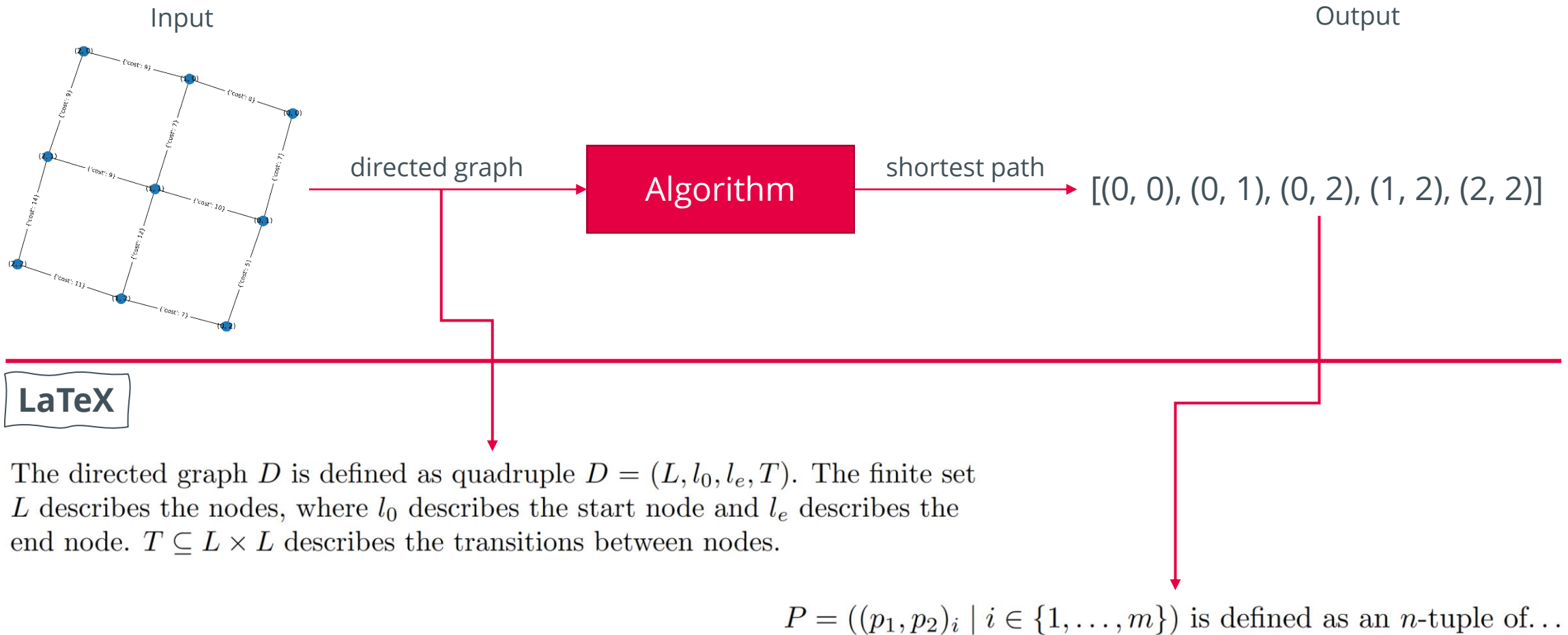
The shortest path of a directed graph should be calculated.

Textual description of the problem

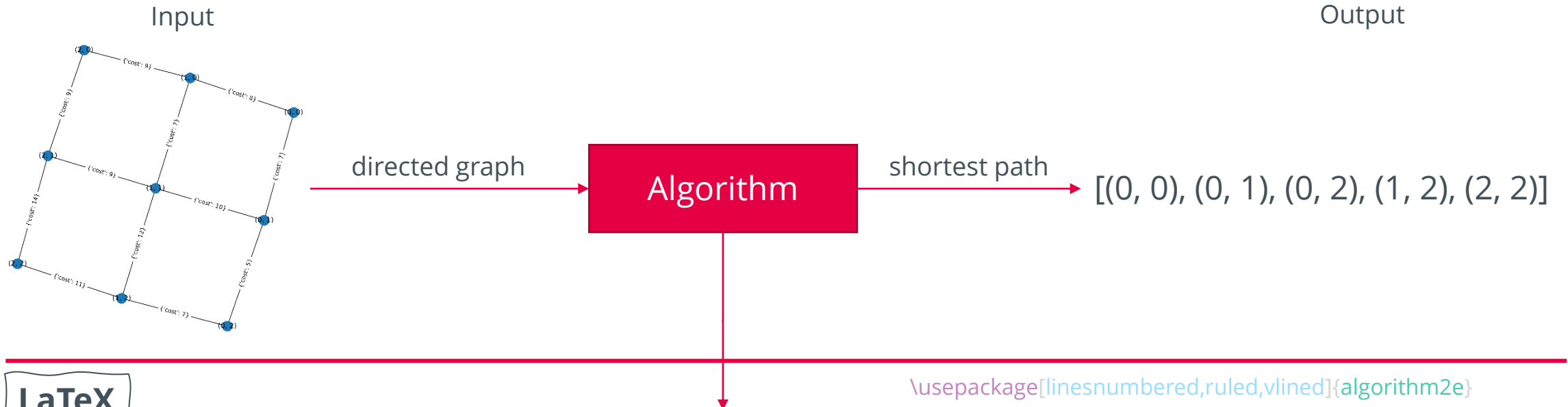


<https://networkx.org>

# Algorithm Design: (2) Formalization



# Algorithm Design: (3) Algorithm




---

**Algorithm 1:** The  $A^*$  ...

---

**Input:** Directed graph  $D$

**Output:**  $n$ -tuple of point tuple

// Step 1: (Calculate shortest path):

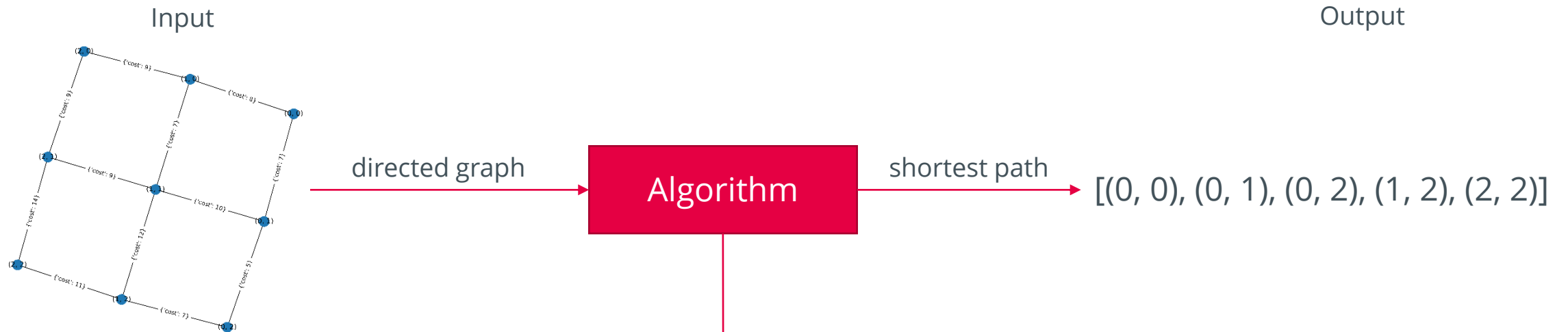
1  $P \leftarrow A^*(D)$

2 return  $P$

---



# Algorithm Design: (4) Implementation



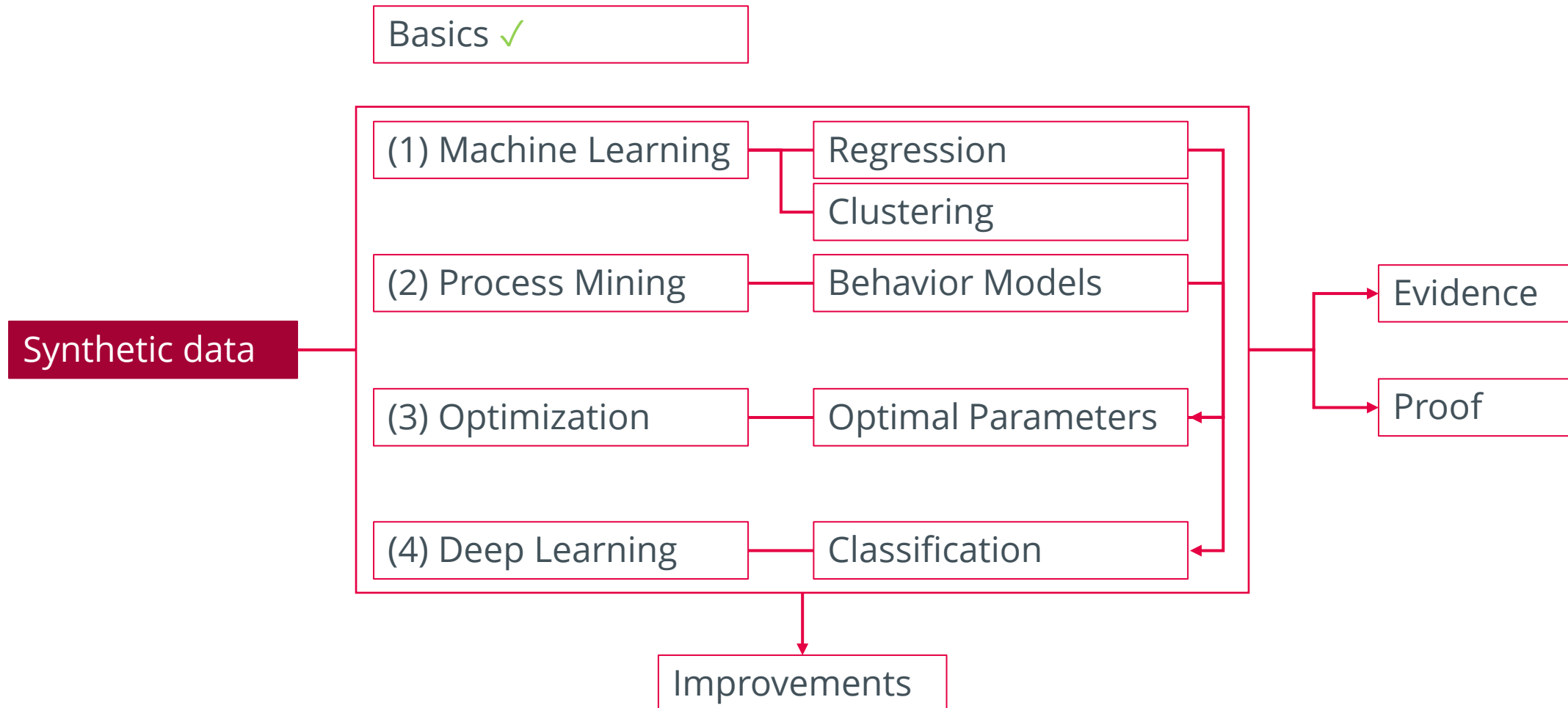
Python

```

Run Cell | Run Below | Debug Cell
2  # %%
3  import random
4  import math
5  import networkx as nx
6  import matplotlib.pyplot as plt
7  import matplotlib
8
9  from networkx.classes.graph import Graph
10 from networkx.algorithms.shortest_paths.weighted import _weight_function
11 from heapq import heappush, heappop
12 from itertools import count
13
14 matplotlib.rcParams.update({'font.size': 22, 'font.family': 'Arial'})
15
16 def star(G, source, target, heuristic=None, weight="weight"):
17
18     push = heappush
19     pop = heappop
20     weight = _weight_function(G, weight)
21

```

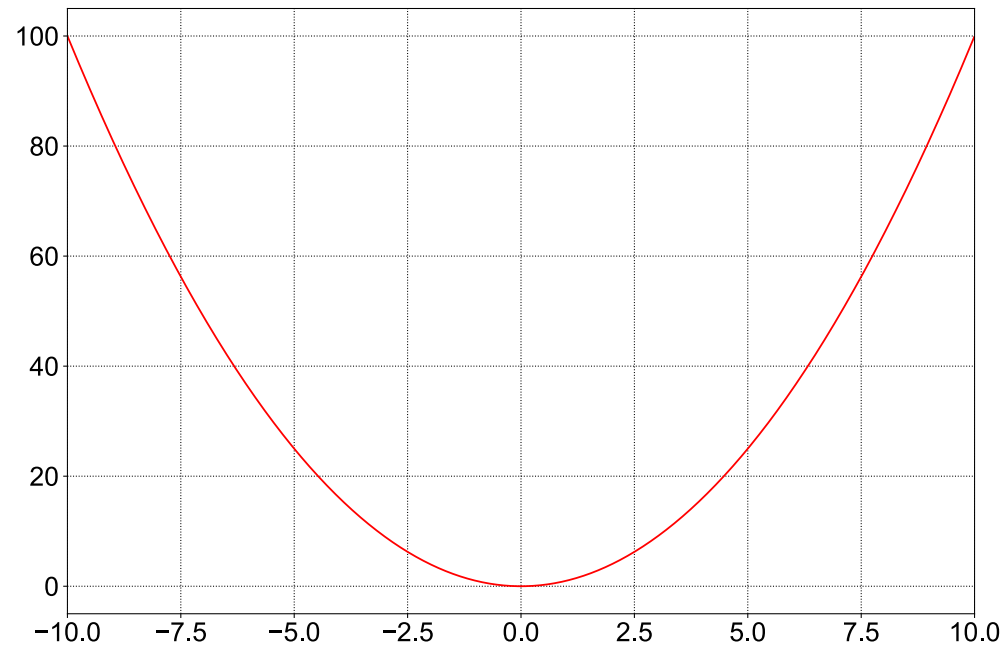
# Summary: Advanced Topics in Algorithms



# Synthetic Data 1/2

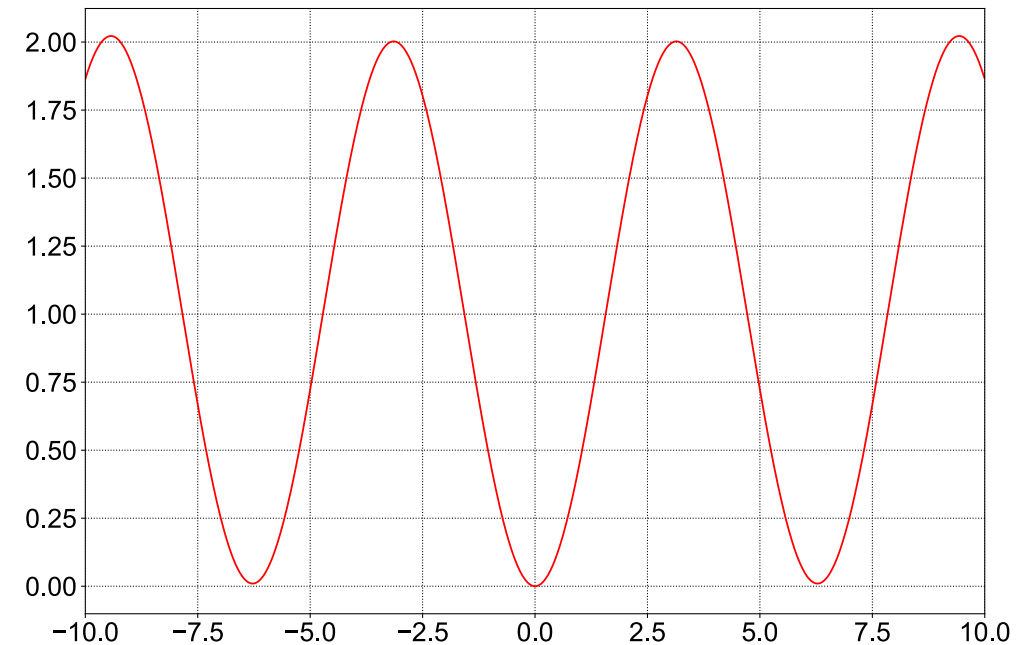
## Sphere

$$f_s(x) = \sum_{i=1}^n x_i^2$$



## Griewank

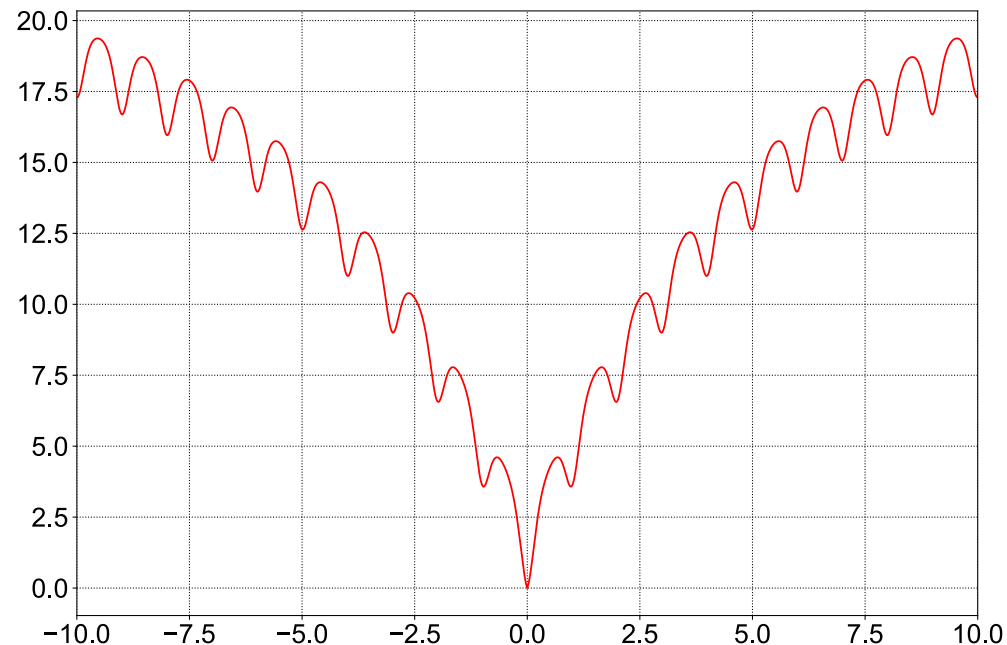
$$f_g(x) = 1 - \prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right) + \sum_{i=1}^n \frac{x_i^2}{4000}$$



# Synthetic Data 2/2

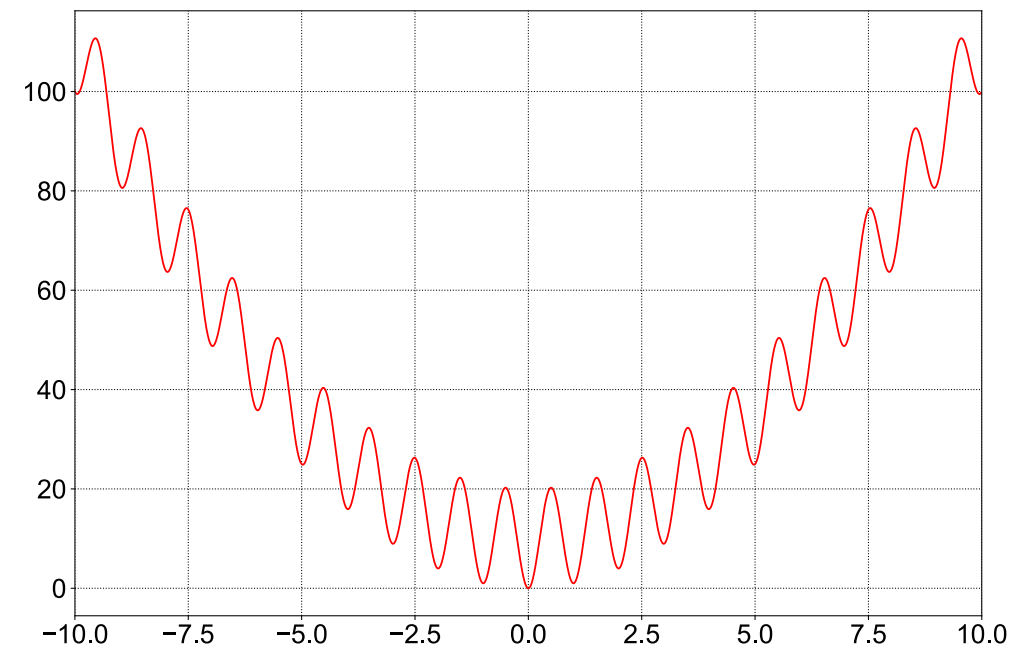
## Ackley

$$f_a(x) = 20 - 20 * \exp \left[ -0.2 \sqrt{\frac{\sum_{i=1}^n x_i^2}{n}} \right] + \exp(1) - \exp \left[ \frac{\sum_{i=1}^n \cos(2\pi x_i)}{n} \right]$$

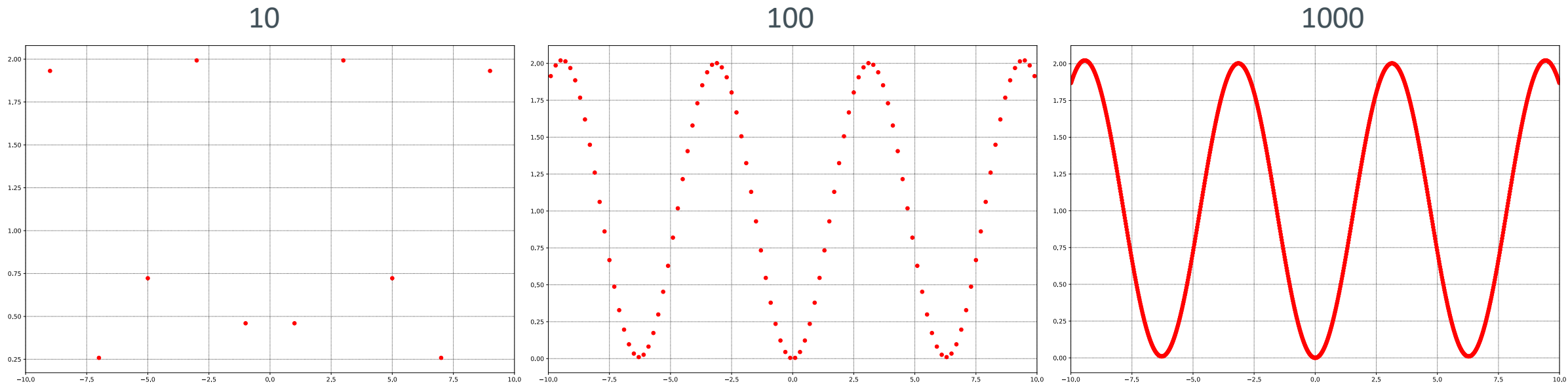


## Rastrigin

$$f_r(x) = 10 \cdot \left[ n - \sum_{i=1}^n \cos(2\pi x_i) \right] + \sum_{i=1}^n x_i^2$$



# Synthetic Data: Latin Hypercube Sampling

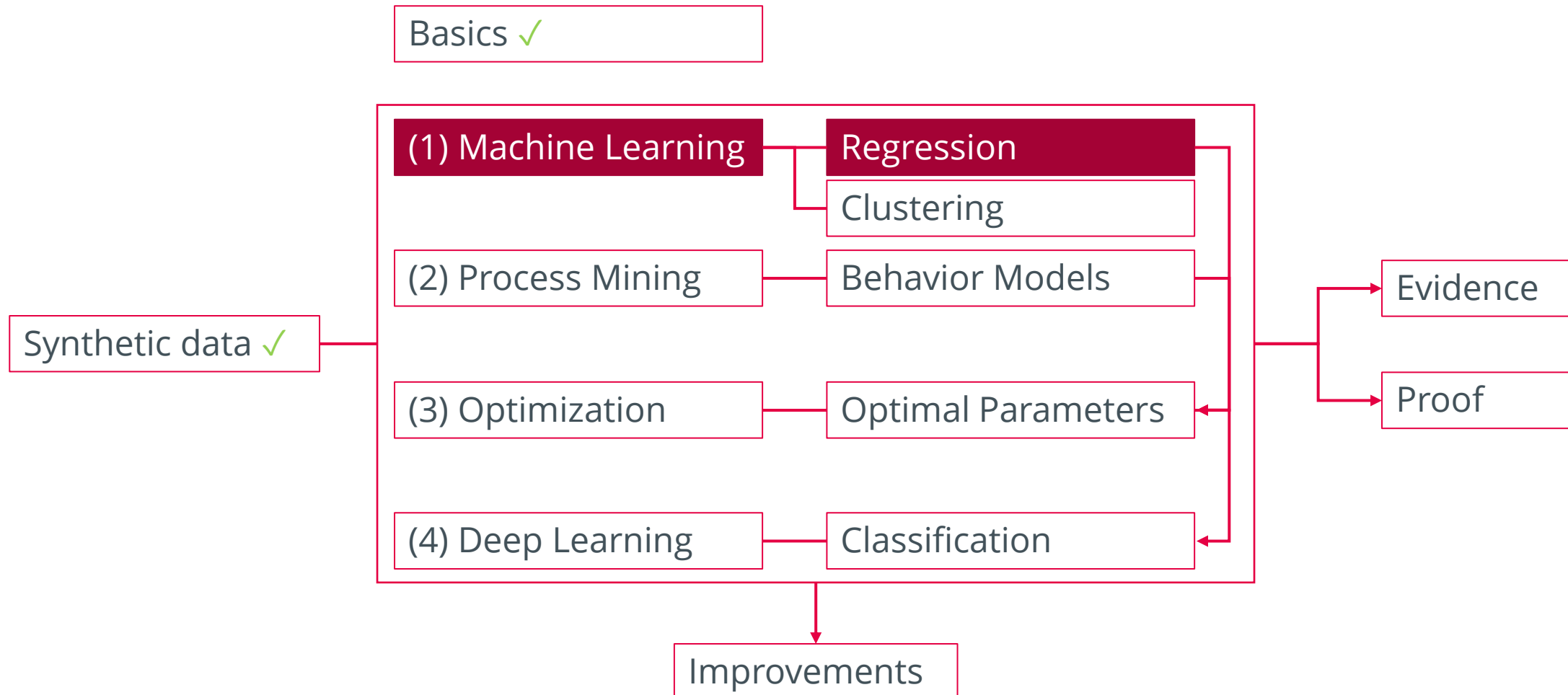


It is among the most popular sampling techniques in computer experiments thanks to its simplicity and projection properties with high-dimensional problems.

Jin, R. and Chen, W. and Sudjianto, A. (2005), "An efficient algorithm for constructing optimal design of computer experiments." *Journal of Statistical Planning and Inference*, 134:268-287.

[https://smt.readthedocs.io/en/latest/\\_src\\_docs/sampling\\_methods/lhs.html](https://smt.readthedocs.io/en/latest/_src_docs/sampling_methods/lhs.html)

# Summary: Advanced Topics in Algorithms



# Symbolic Regression 1/3

## Input

Time	Energy	
09:59	0	
10:00	21.16	
...		⋮
10:05	126.97	
10:06	0	
...		

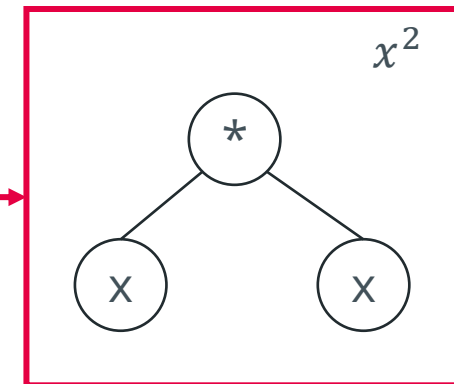
## Primitives

- add
- sub
- mult
- ...

- mean squared error
- root mean squared error

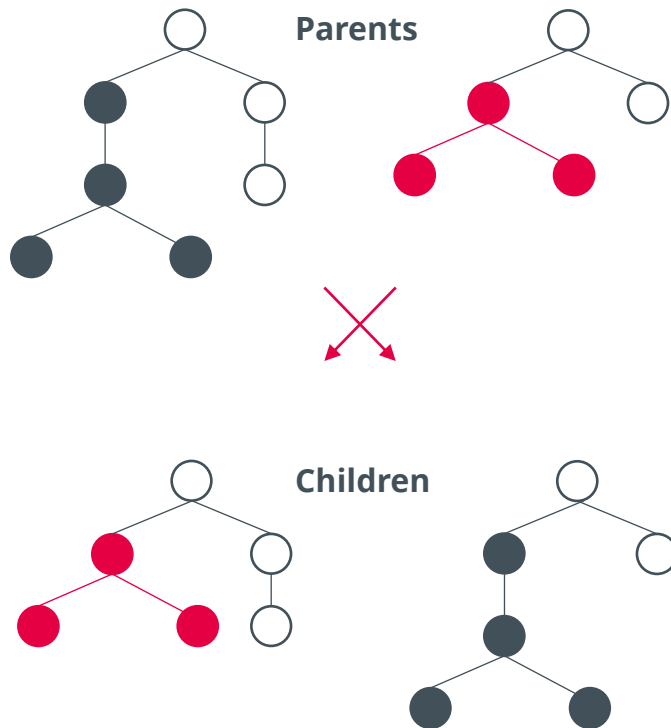
## Symbolic Regression

## Output



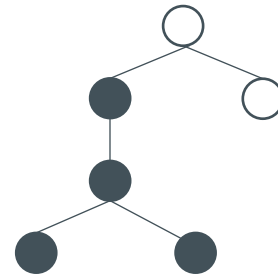
# Symbolic Regression 2/3

## Crossover

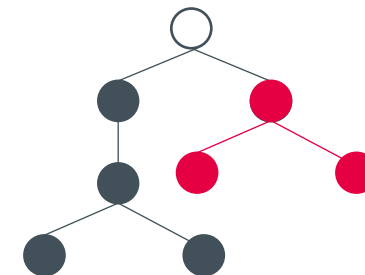
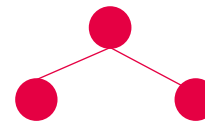


## Mutation

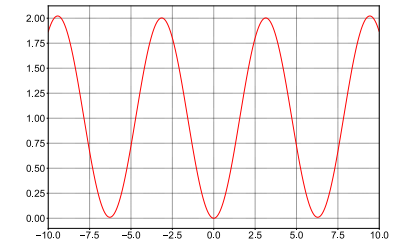
### Selected Individual



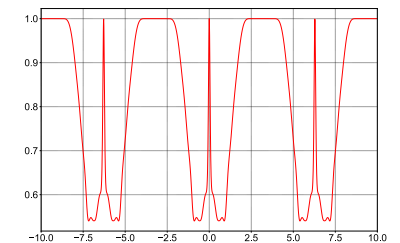
### Generate Mutation



## Evaluate

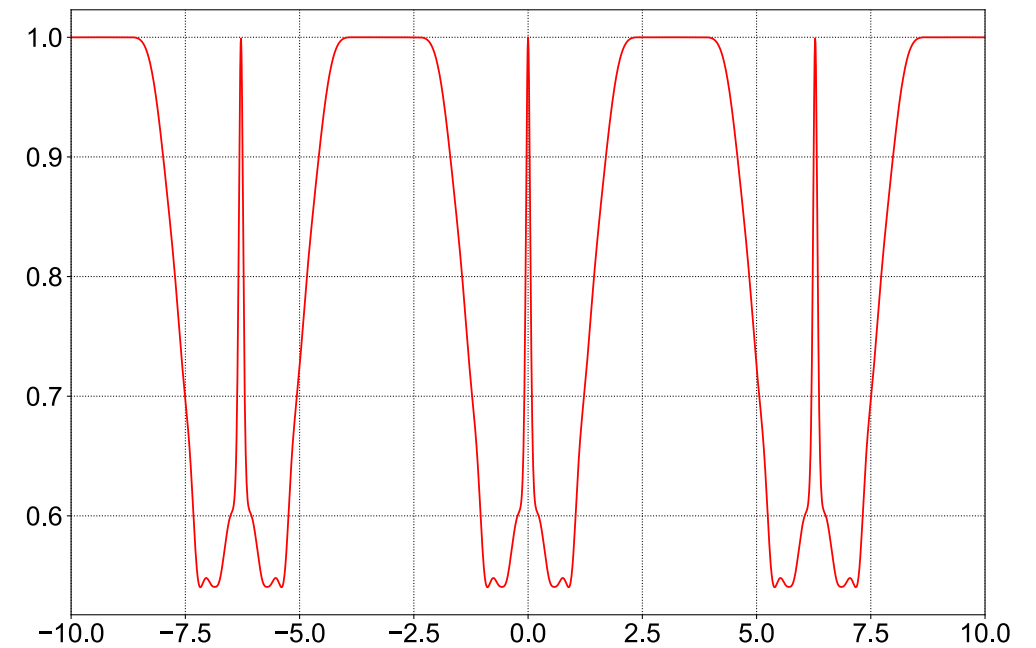
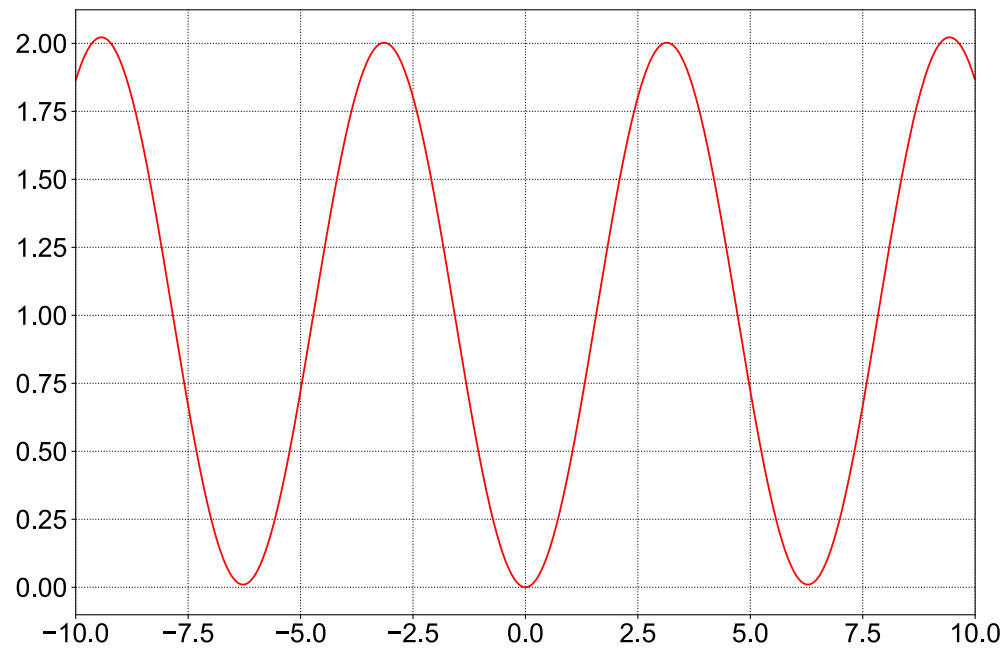


e.g. mean squared error

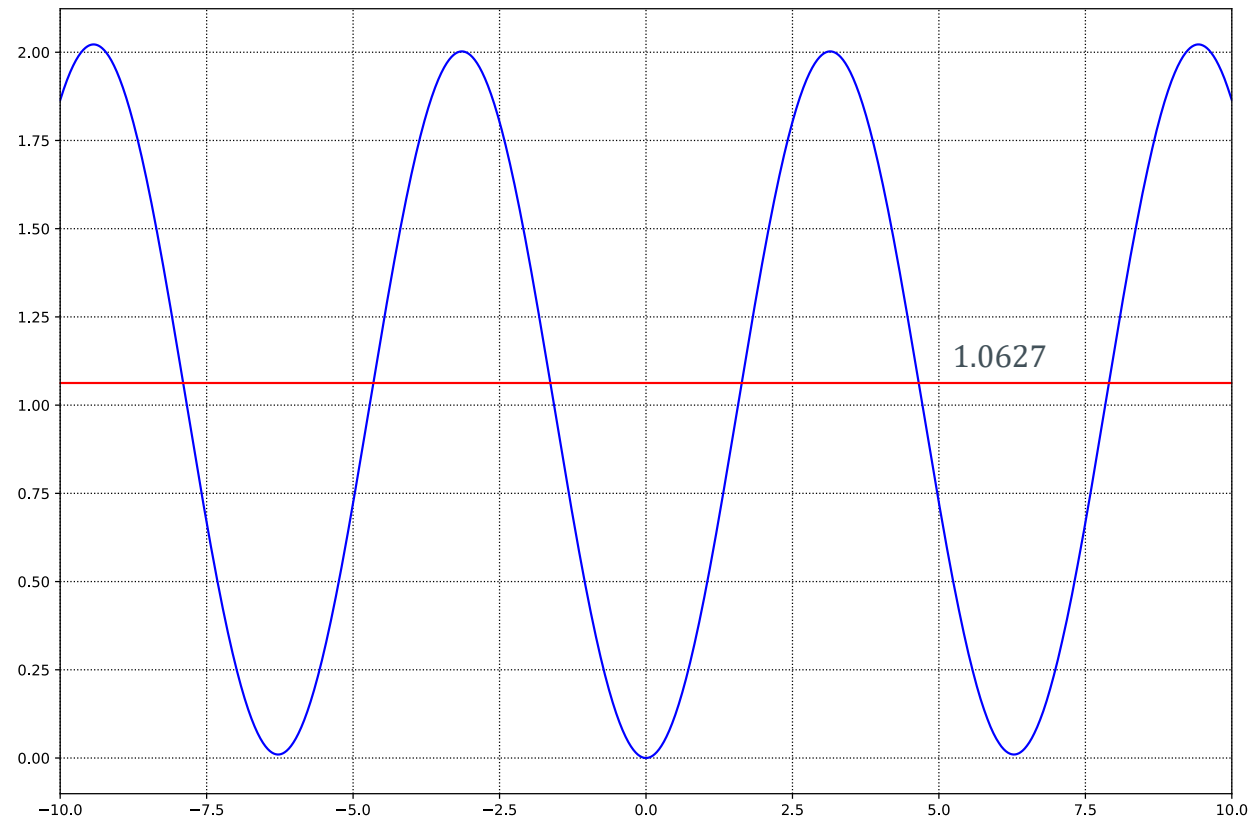




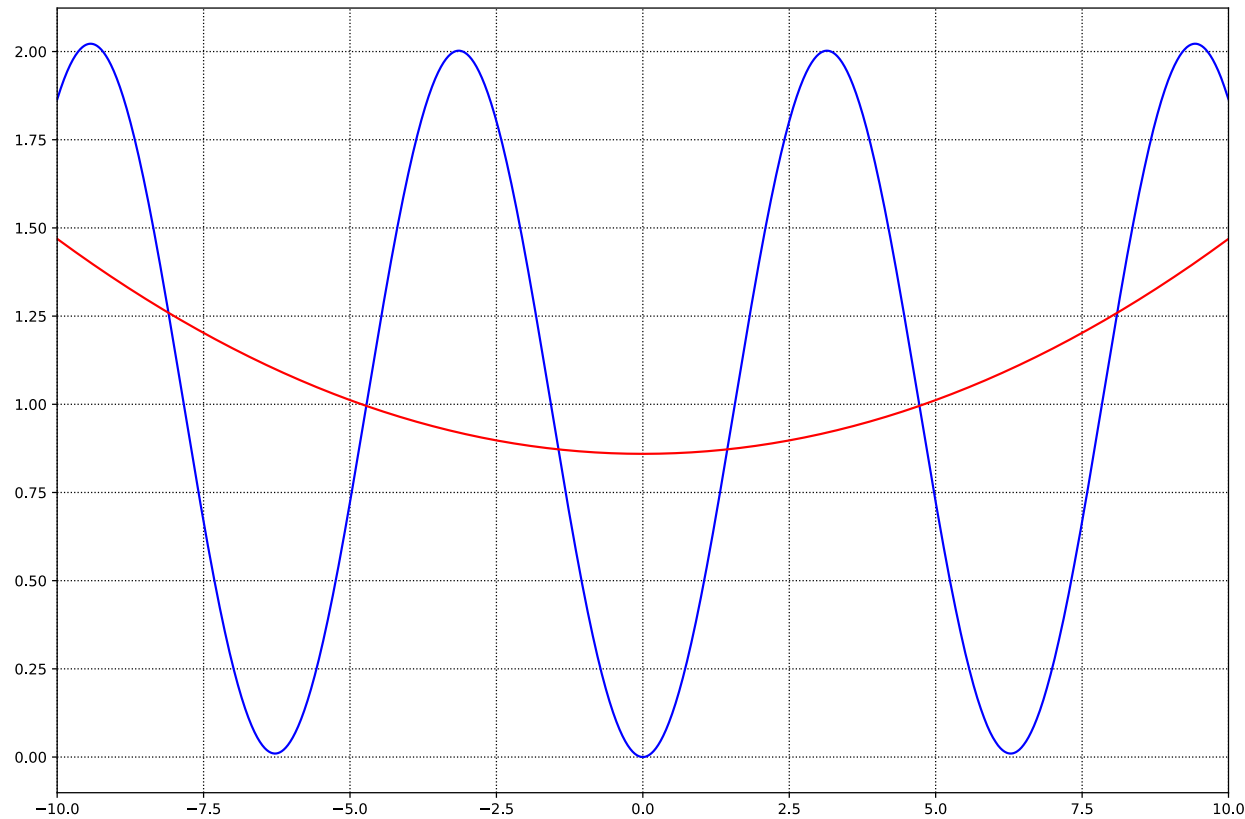
# Symbolic Regression 3/3



# Regression with Polynomial Features: Step 1

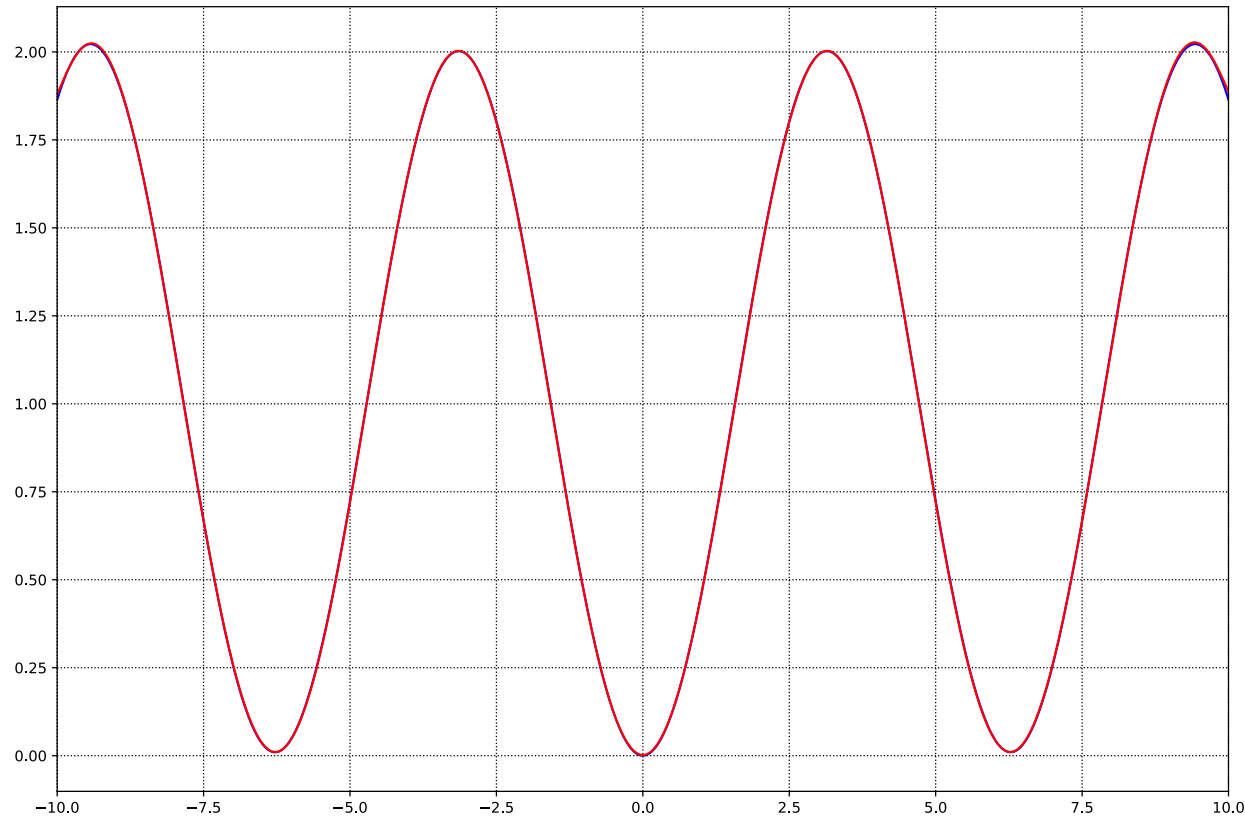


# Regression with Polynomial Features: Step 2



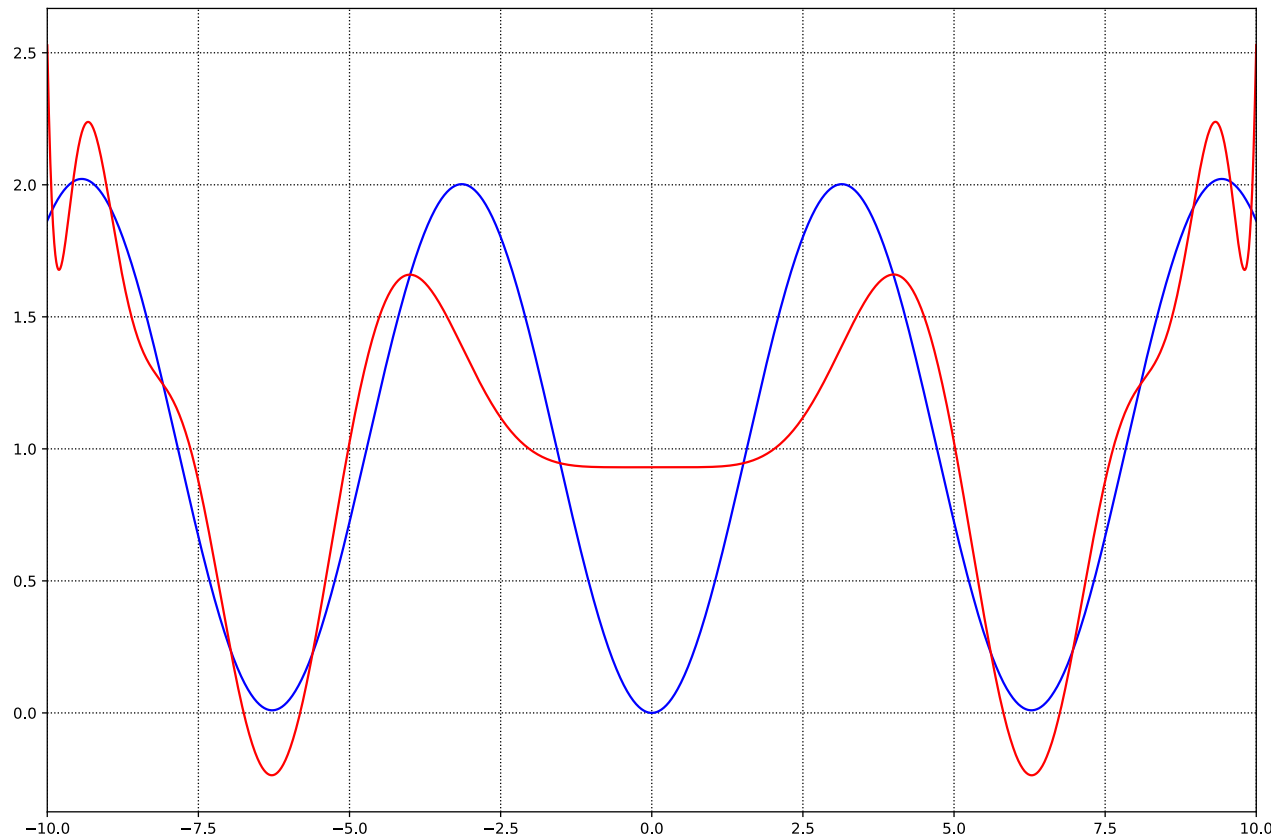
$$0.00609567x_0^2 + 0.8595$$

# Regression with Polynomial Features: Step 14



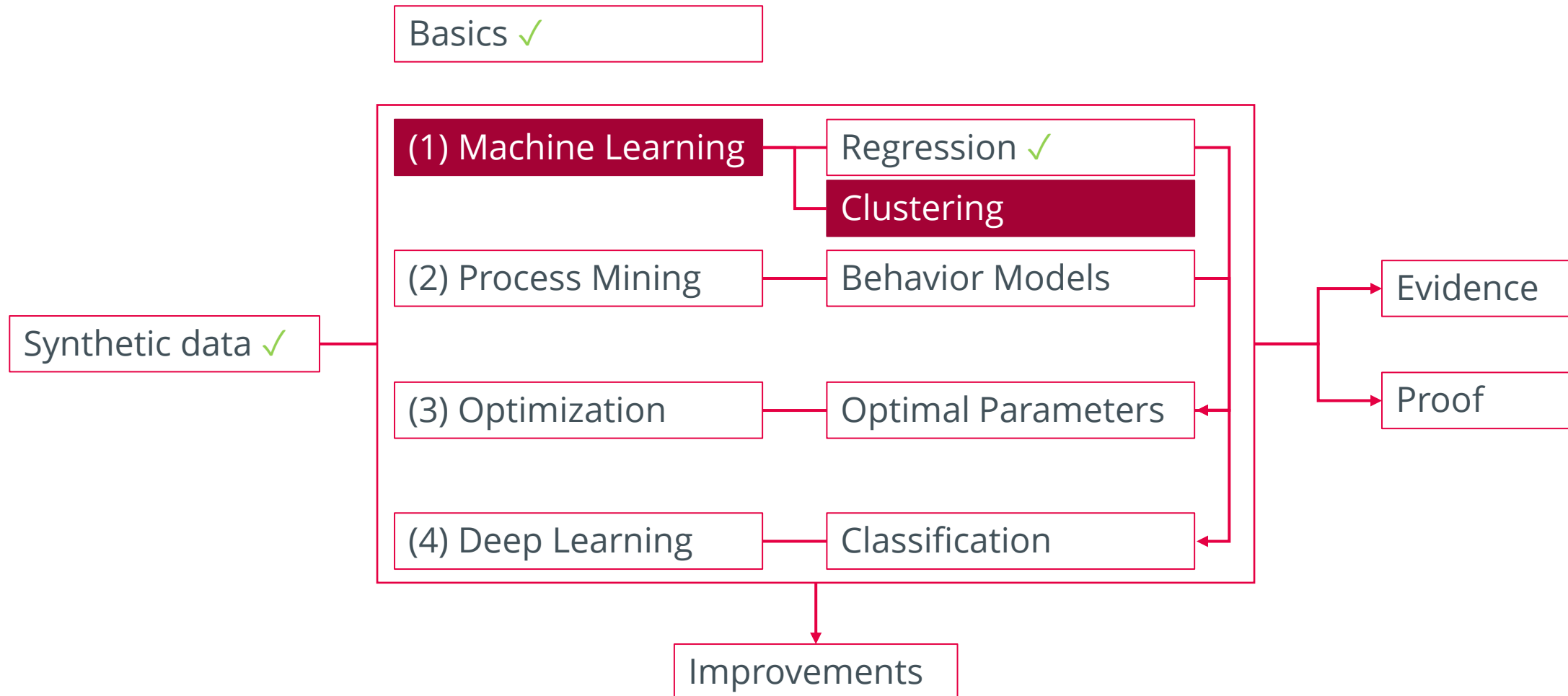
$$\begin{aligned} &2.0 \cdot 10^{-7} x_0^{10} - 2.167 \cdot 10^{-5} x_0^8 \\ &+ 0.00131727 x_0^6 - 4.0 \cdot 10^{-8} x_0^5 \\ &- 0.04085166 x_0^4 + 1.78 \cdot 10^{-6} x_0^3 \\ &+ 0.49669338 x_0^2 - 3.354 \cdot 10^{-5} x_0 + 0.0025 \end{aligned}$$

# Regression with Polynomial Features: Step 18

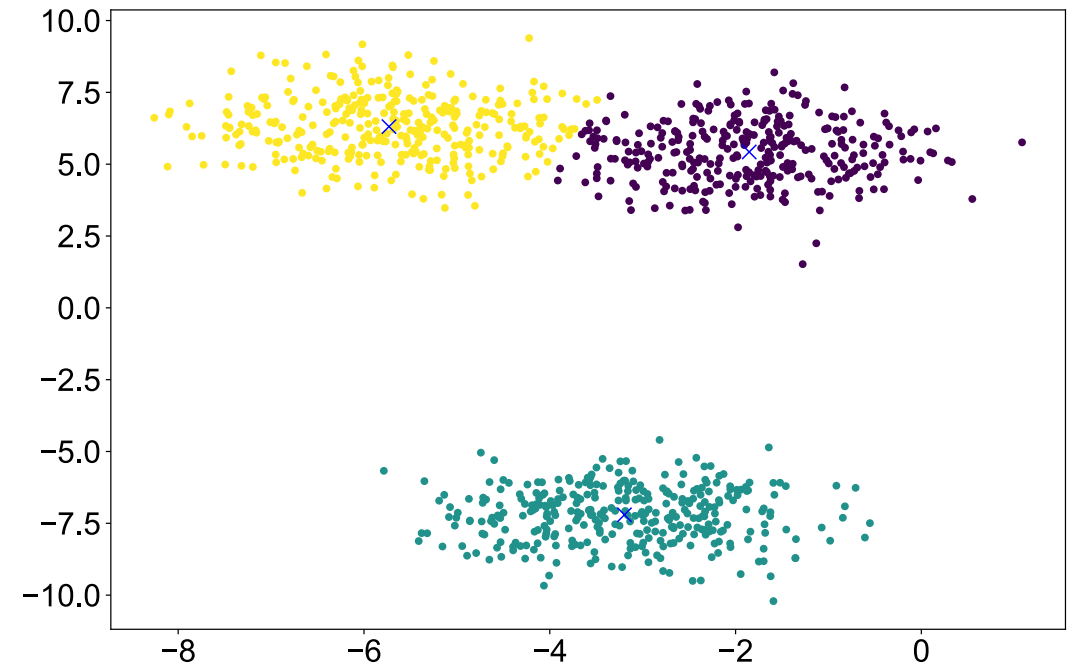
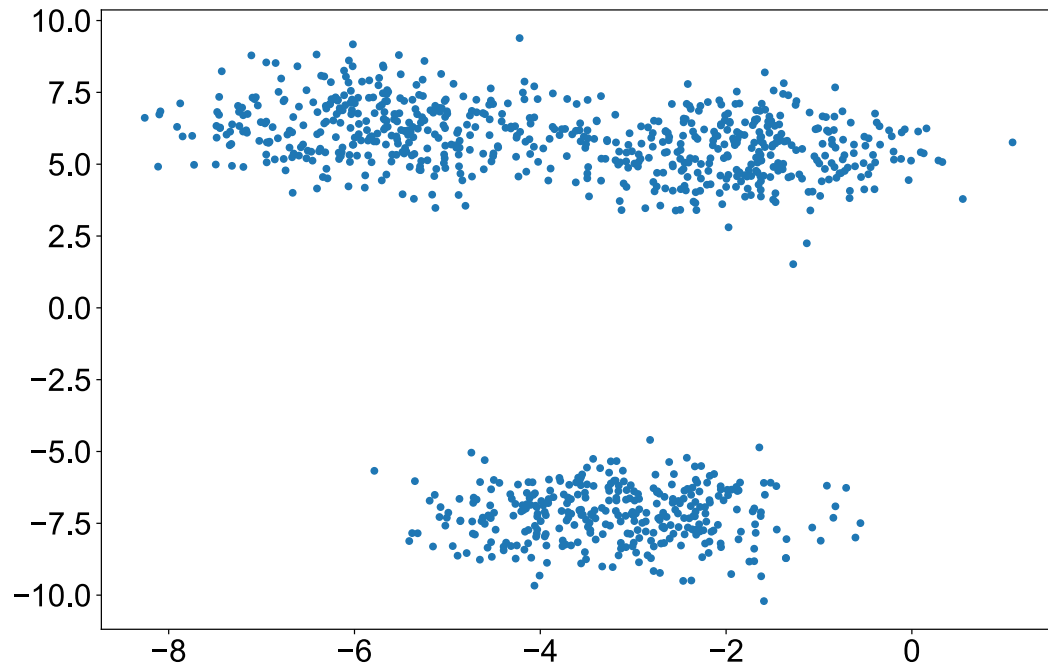


$$\begin{aligned} & -1.4 \cdot 10^{-7} x_0^{12} + 6.5 \cdot 10^{-6} x_0^{10} + 2.0 \cdot 10^{-8} x_0^9 \\ & - 0.0001587 x_0^8 - 4.2 \cdot 10^{-7} x_0^7 + 0.00150314 x_0^6 \\ & + 3.51 \cdot 10^{-6} x_0^5 + 0.00022478 x_0^4 + 6.6 \cdot 10^{-7} x_0^3 \\ & + 1.824 \cdot 10^{-5} x_0^2 + 1.589 \cdot 10^{-5} x_0 + 0.9302 \end{aligned}$$

# Summary: Advanced Topics in Algorithms

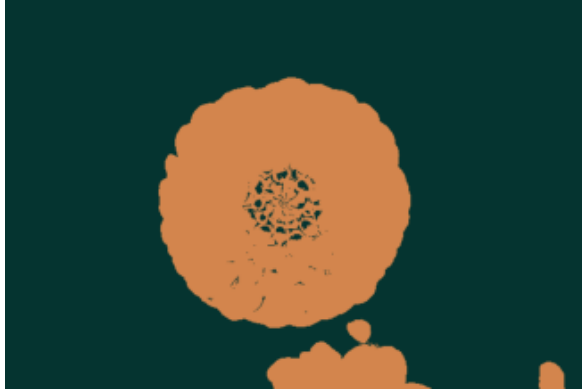


# K-Means



# K-Means

Quantized image (2 colors, K-Means)



Quantized image (16 colors, K-Means)



Quantized image (32 colors, K-Means)



Quantized image (64 colors, K-Means)



Quantized image (128 colors, K-Means)



Original image (96,615 colors)





# K-Means

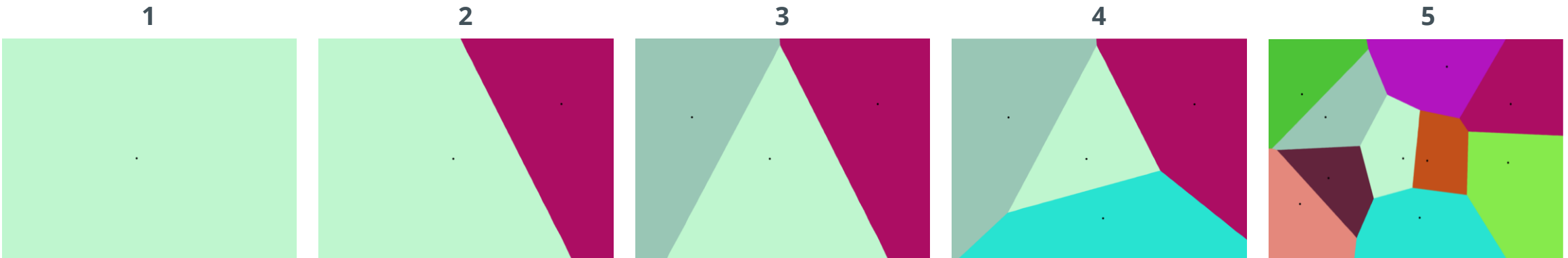
The algorithm can also be understood through the concept of Voronoi diagrams [1]:

1. First the Voronoi diagram of the points is calculated using the current centroids. Each segment in the Voronoi diagram becomes a separate cluster.
2. Secondly, the centroids are updated to the mean of each segment.

The algorithm then repeats this until a stopping criterion is fulfilled.

---

**Voronoi Diagram [2]**

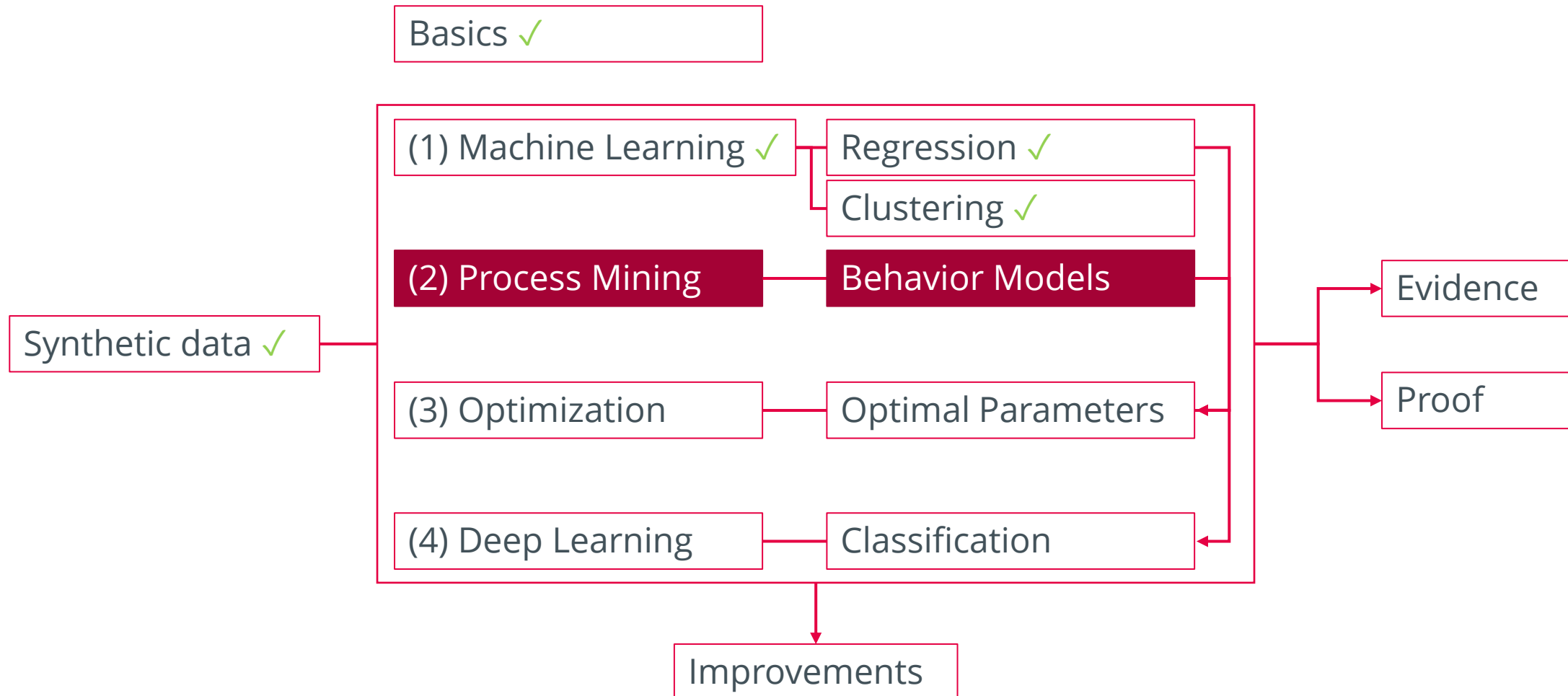


Fortune's algorithm:  $O(n \log n)$  time and  $O(n)$  space

[1] <https://scikit-learn.org/stable/modules/clustering.html#k-means>

[2] <http://alexbeutel.com/webgl/voronoi.html>

# Summary: Advanced Topics in Algorithms



# Produce Coffee

## (1) Power on



Activity	Timestamp
power on	12:00:00

--	--

## (2) Take Pad



Activity	Timestamp
power on	12:00:00
take pad	12:05:00

--	--

## (3) Add Pad



Activity	Timestamp
power on	12:00:00
take pad	12:05:00
add pad	12:05:30

--	--

## (4) Close

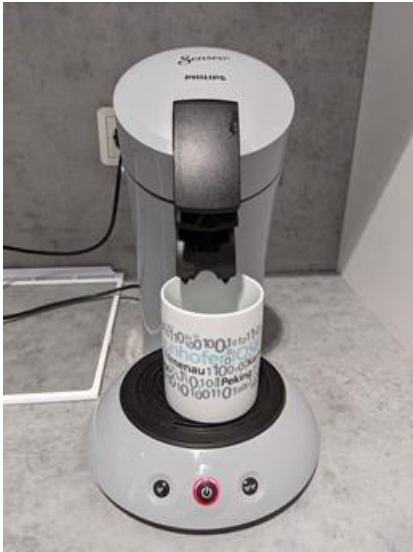


Activity	Timestamp
power on	12:00:00
take pad	12:05:00
add pad	12:05:30
close	12:05:40

--	--

# Produce Coffee

## (5) Start



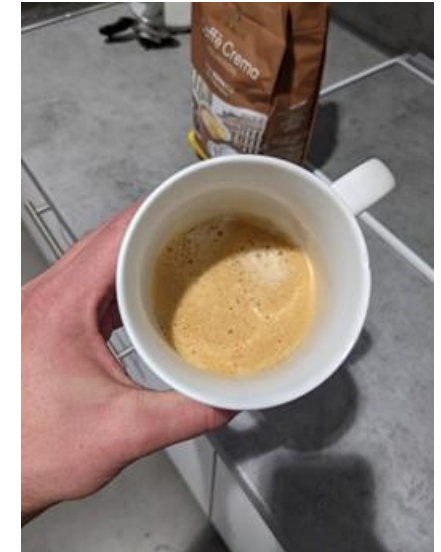
## (6) Produce



## (7) Remove Pad



## (8) Enjoy



Activity	Timestamp
start	12:05:45

Activity	Timestamp
start	12:05:45
produce	12:07:00

Activity	Timestamp
start	12:05:45
produce	12:07:00
remove pad	12:07:30

Activity	Timestamp
start	12:05:45
produce	12:07:00
remove pad	12:07:30
enjoy	12:20:00

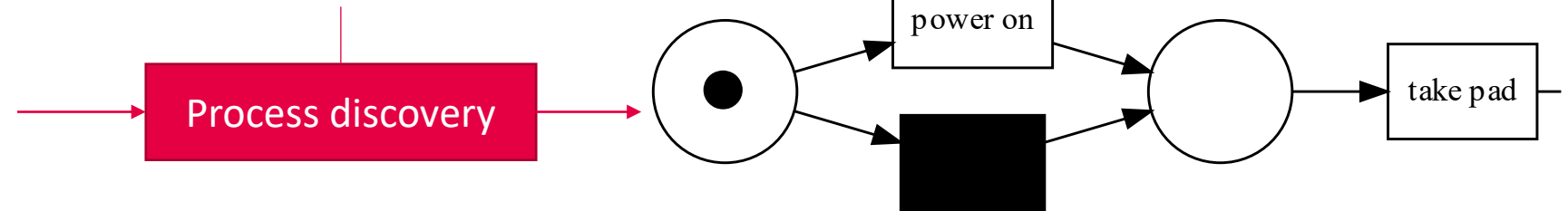
# Process Mining

## Event log\*

Case	Activity	Timestamp
1	power on	12:00:00
1	take pad	12:05:00
1	add pad	12:05:30
1	close	12:05:40
1	start	12:05:45
1	produce	12:07:00
1	remove pad	12:07:30
1	enjoy	12:20:00
2	power on	12:45:00
...		

## Petri Net

- Alpha miner
- Heurist miner
- Inductive miner



\*IEEE XES is a standard format describing how event logs are stored  
<http://www.xes-standard.org/>

# Excuse: Deep Packet Inspection

0\_normal\_run\_1.pcap

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Operation	Info
1	0.000000	Hirschma_5c:46:4a	Spanning-tree-(for-...	STP	60		RST, Root = 32768/0/ec:74:ba:5c:46:40 Cost = 0 Port = 0x8006
2	0.000417	Hirschma_5c:46:4b	Spanning-tree-(for-...	STP	60		RST, Root = 32768/0/ec:74:ba:5c:46:40 Cost = 0 Port = 0x8007
3	0.199438	Hirschma_5c:46:48	LLDP_Multicast	LLDP	273		MA/ec:74:ba:5c:46:40 MA/ec:74:ba:5c:46:48 120 SysN=192.168.0.141 SysD=Hirschmann Rail Switch Power - SW: HIOS-2S-PRP-07
4	0.930132	Siemens_ad:78:fc	PN-MC_00:00:00	PN-DCP	60		Ident Req, Xid:0x1020002, NameOfStation:"bk9103-1"
5	0.930132	Siemens_ad:78:fc	PN-MC_00:00:00	PN-DCP	60		Ident Req, Xid:0x1020002, NameOfStation:"bk9103-1"
6	0.932825	Beckhoff_2b:ae:e8	Siemens_ad:78:fc	PN-DCP	132		Ident Ok, Xid:0x1020002, NameOfStation:"bk9103-1", Dev-Options(9), DeviceVendorValue, Dev-Role, Dev-ID, IP, MAC
7	1.116081	Hirschma_5c:46:48	LLDP_Multicast	LLDP	273		MA/ec:74:ba:5c:46:40 MA/ec:74:ba:5c:46:48 120 SysN=192.168.0.141 SysD=Hirschmann Rail Switch Power - SW: HIOS-2S-PRP-07
8	1.208735	::	ff02::16	ICMPv6	110		Multicast Listener Report Message v2
9	1.496733	::	ff02::16	ICMPv6	110		Multicast Listener Report Message v2
10	1.517053	Hirschma_5c:46:40	Broadcast	ARP	60		Who has 192.168.0.141? (ARP Probe)
11	1.517323	Hirschma_5c:46:40	Broadcast	ARP	60		Who has 192.168.0.141? (ARP Probe)
12	1.528741	::	ff02::1:ff98:e49	ICMPv6	86		Neighbor Solicitation for fe80::6841:e60a:ac98:e49
13	1.700483	Hirschma_5c:46:40	Broadcast	ARP	60		Who has 192.168.0.141? (ARP Probe)
14	1.700483	Hirschma_5c:46:40	Broadcast	ARP	60		Who has 192.168.0.141? (ARP Probe)
15	1.833188	Hirschma_5c:46:4a	Spanning-tree-(for-...	STP	60		RST, Root = 32768/0/ec:74:ba:5c:46:40 Cost = 0 Port = 0x8006
16	1.833798	Hirschma_5c:46:4b	Spanning-tree-(for-...	STP	60		RST, Root = 32768/0/ec:74:ba:5c:46:40 Cost = 0 Port = 0x8007
17	1.846887	Siemens_ad:78:fc	Broadcast	ARP	64		Who has 192.168.0.2? Tell 192.168.0.1
18	1.883742	Hirschma_5c:46:40	Broadcast	ARP	60		Who has 192.168.0.141? (ARP Probe)
19	1.883980	Hirschma_5c:46:40	Broadcast	ARP	60		Who has 192.168.0.141? (ARP Probe)
20	2.067031	Hirschma_5c:46:40	Broadcast	ARP	60		Who has 192.168.0.141? (ARP Probe)
21	2.067321	Hirschma_5c:46:40	Broadcast	ARP	60		Who has 192.168.0.141? (ARP Probe)
22	2.255461	Hirschma_5c:46:40	Broadcast	ARP	60		ARP Announcement for 192.168.0.141
23	2.255461	Hirschma_5c:46:40	Broadcast	ARP	60		ARP Announcement for 192.168.0.141
24	2.568233	fe80::6841:e60a:ac9...	ff02::16	ICMPv6	110		Multicast Listener Report Message v2
25	2.571766	fe80::6841:e60a:ac9...	ff02::16	ICMPv6	90		Multicast Listener Report Message v2
26	2.573863	fe80::6841:e60a:ac9...	ff02::2	ICMPv6	62		Router Solicitation
27	2.575726	fe80::6841:e60a:ac9...	ff02::16	ICMPv6	130		Multicast Listener Report Message v2
28	2.951745	fe80::6841:e60a:ac9...	ff02::16	ICMPv6	130		Multicast Listener Report Message v2
29	3.143738	fe80::6841:e60a:ac9...	ff02::16	ICMPv6	130		Multicast Listener Report Message v2
30	3.313524	fe80::6841:e60a:ac9...	ff02::fb	MDNS	208		Standard query response 0x0000 PTR, cache flush fedora-carsten.local AAAA, cache flush fe80::6841:e60a:ac98:e49
31	3.666339	Hirschma_5c:46:4a	Spanning-tree-(for-...	STP	60		RST, Root = 32768/0/ec:74:ba:5c:46:40 Cost = 0 Port = 0x8006
32	3.667201	Hirschma_5c:46:4b	Spanning-tree-(for-...	STP	60		RST, Root = 32768/0/ec:74:ba:5c:46:40 Cost = 0 Port = 0x8007

> Frame 1: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)

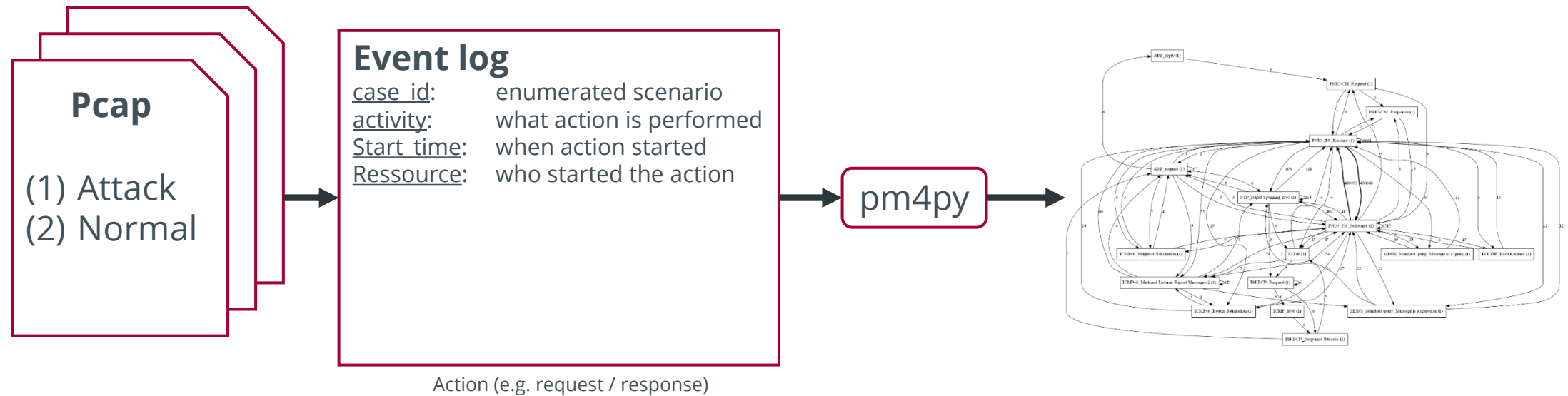
- > IEEE 802.3 Ethernet
- > Logical-Link Control
- > Spanning Tree Protocol

```

0000 01 80 c2 00 00 00 ec 74 ba 5c 46 4a 00 27 42 42 .....t..FJ.'BB
0010 03 00 00 02 02 0e 80 00 ec 74 ba 5c 46 40 00 00 .....t..F@..
0020 00 00 00 00 ec 74 ba 5c 46 40 80 06 00 00 14 00 .....t..F@....
0030 02 00 0f 00 00 00 00 00 00 00 00 00 .....
  
```

0\_normal\_run\_1.pcap | Packets: 117240 · Displayed: 117240 (100.0%) | Profile: Default

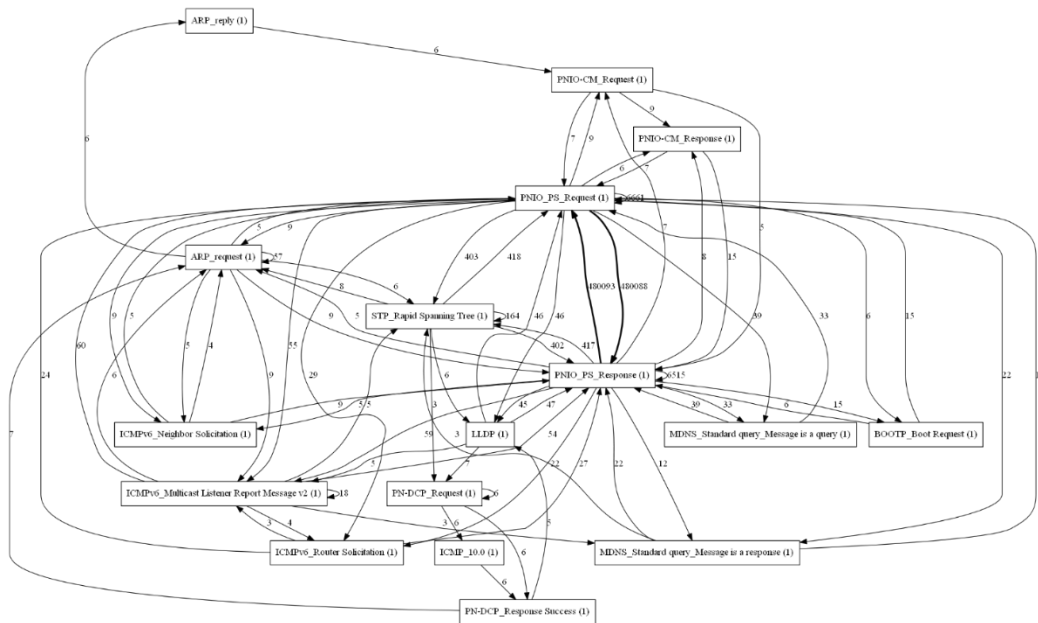
# Excuse: Deep Packet Inspection



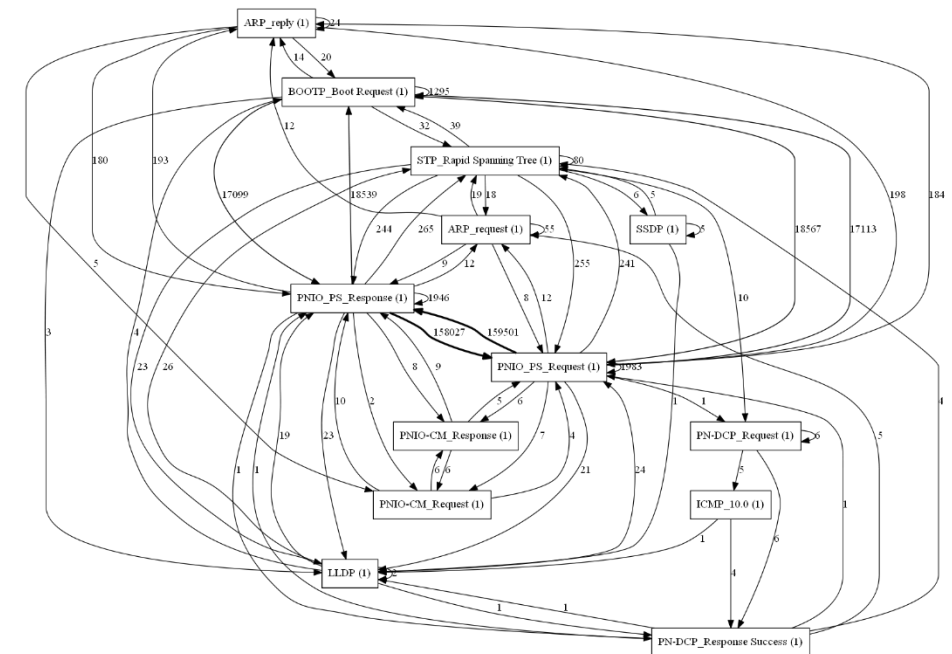


# Excuse: Deep Packet Inspection

Normal

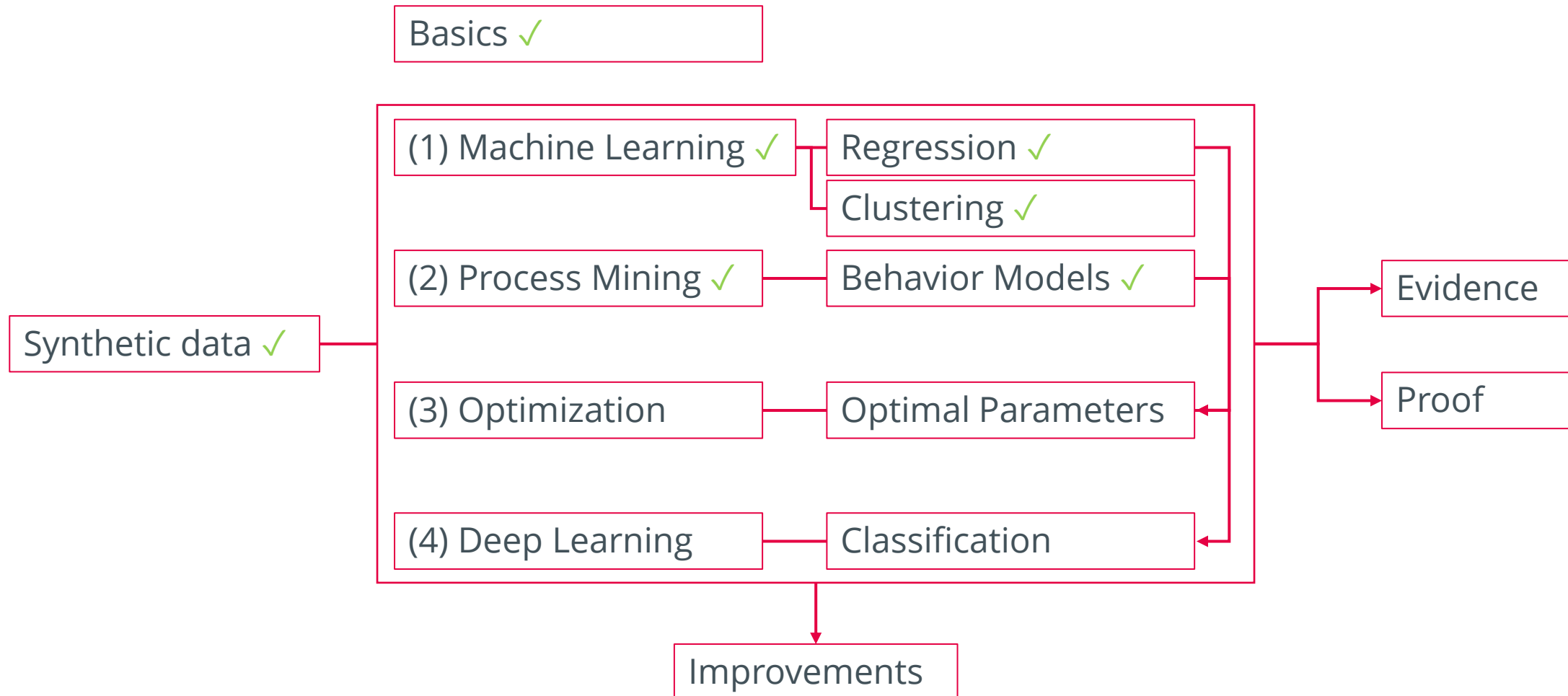


ARP CP





# Summary: Advanced Topics in Algorithms





TECHNISCHE HOCHSCHULE  
OSTWESTFALEN-LIPPE  
UNIVERSITY OF  
APPLIED SCIENCES  
AND ARTS

Thank you!