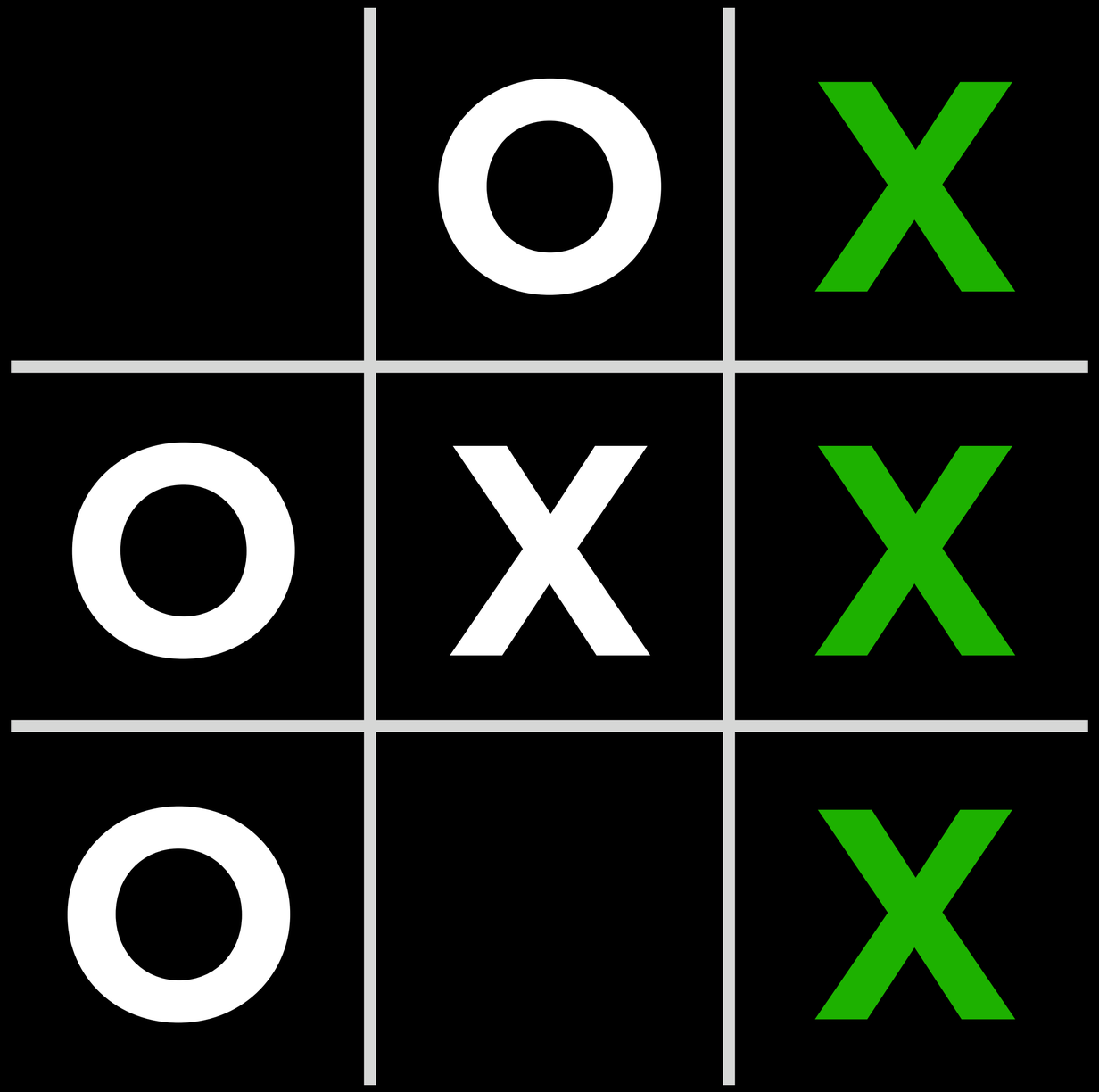
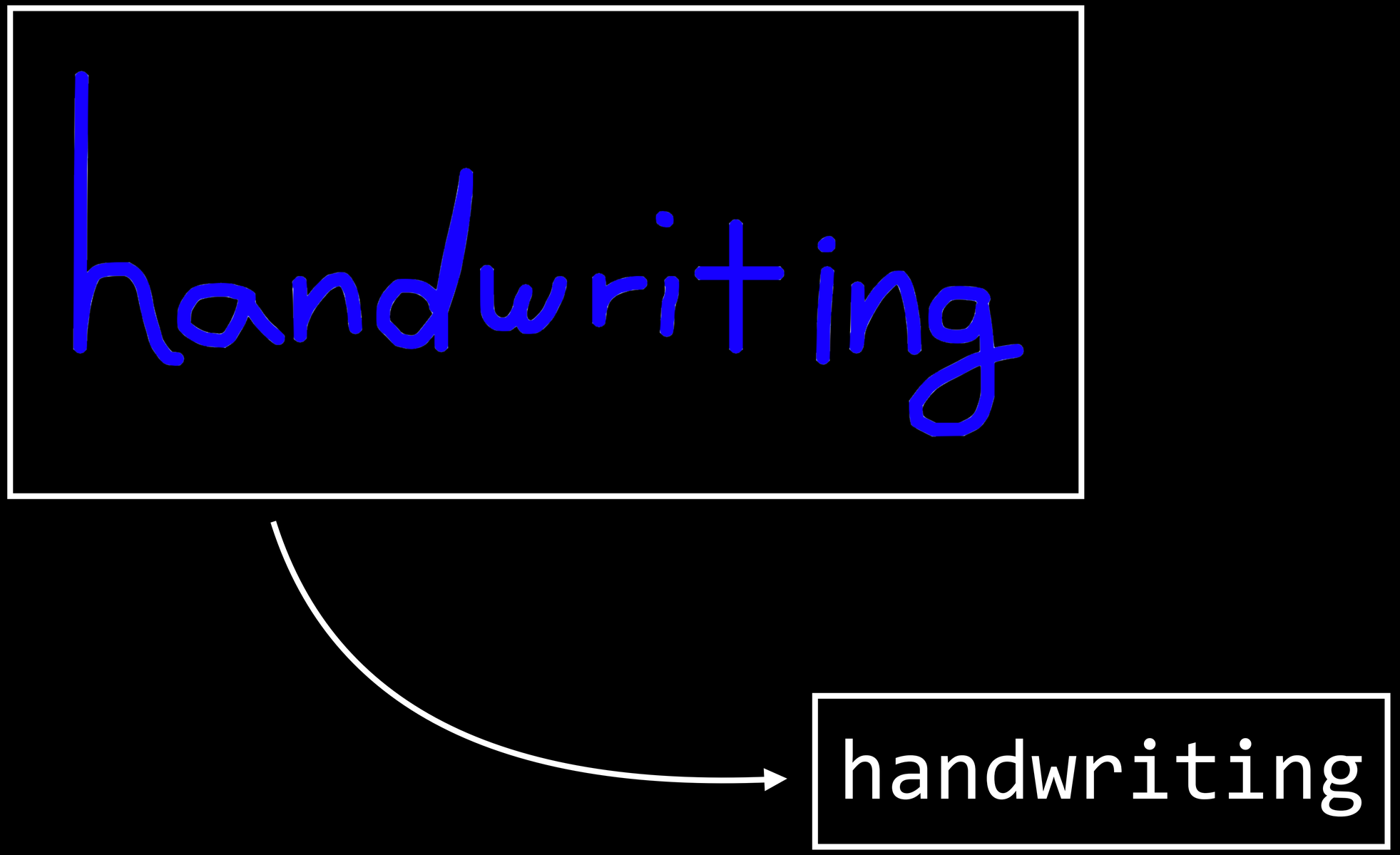
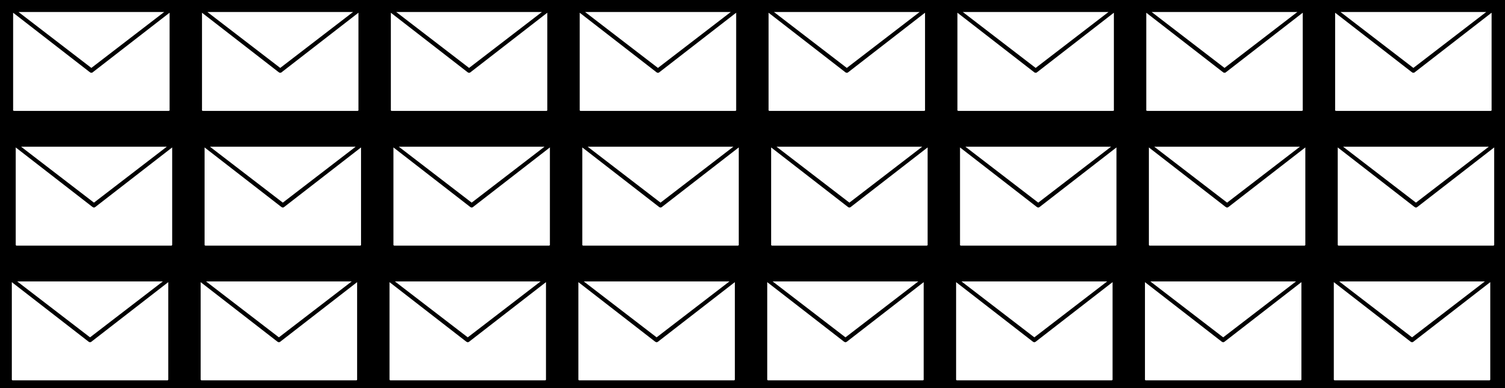
CS50 for MBAs

Artificial Intelligence

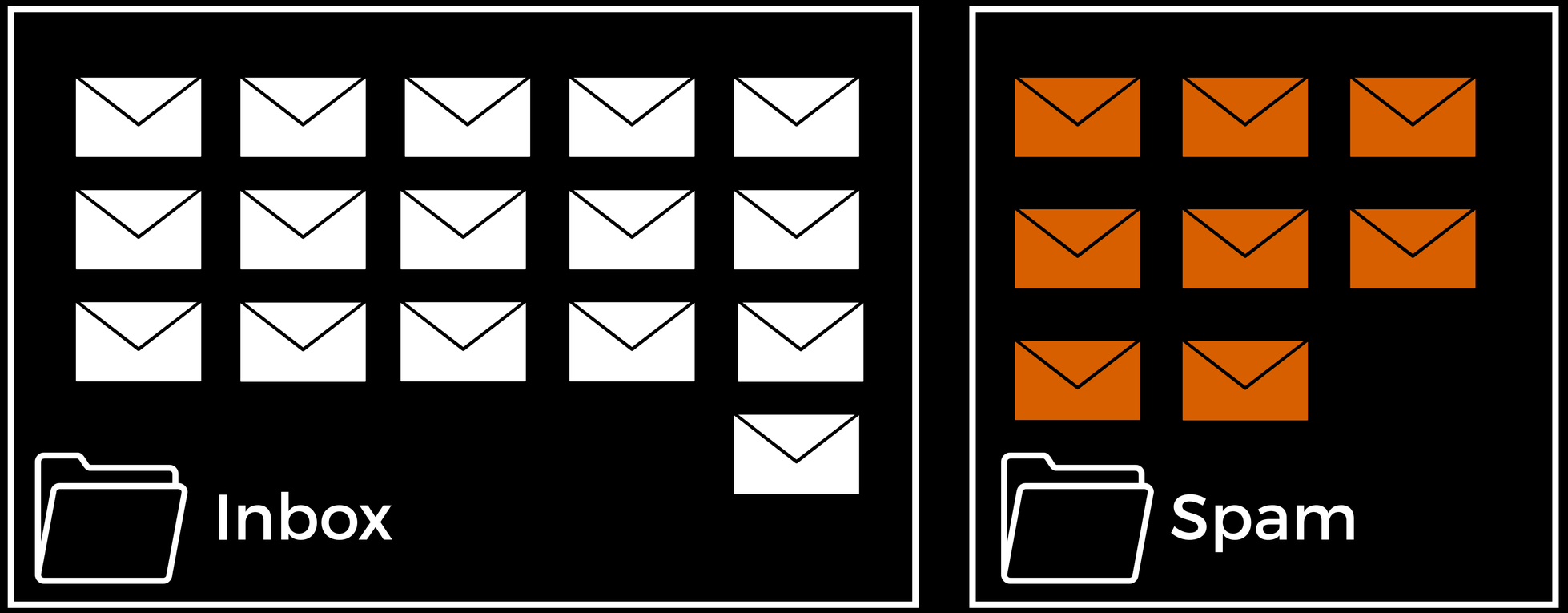

O X
O X X
O X

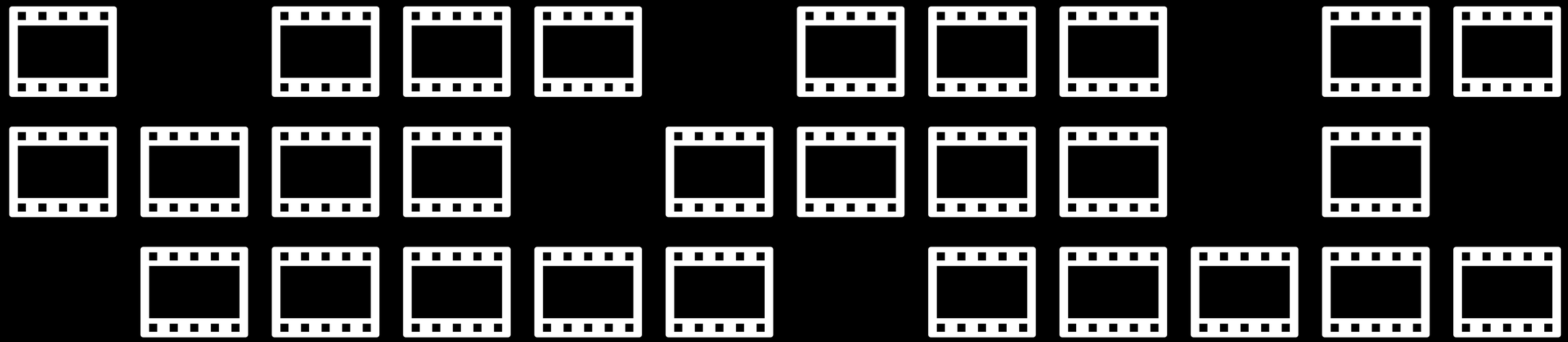

handwriting
handwriting

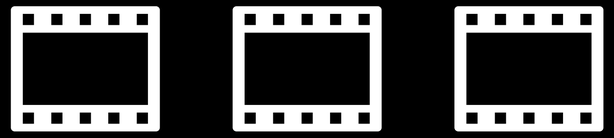


Inbox

Spam


Inbox
Spam

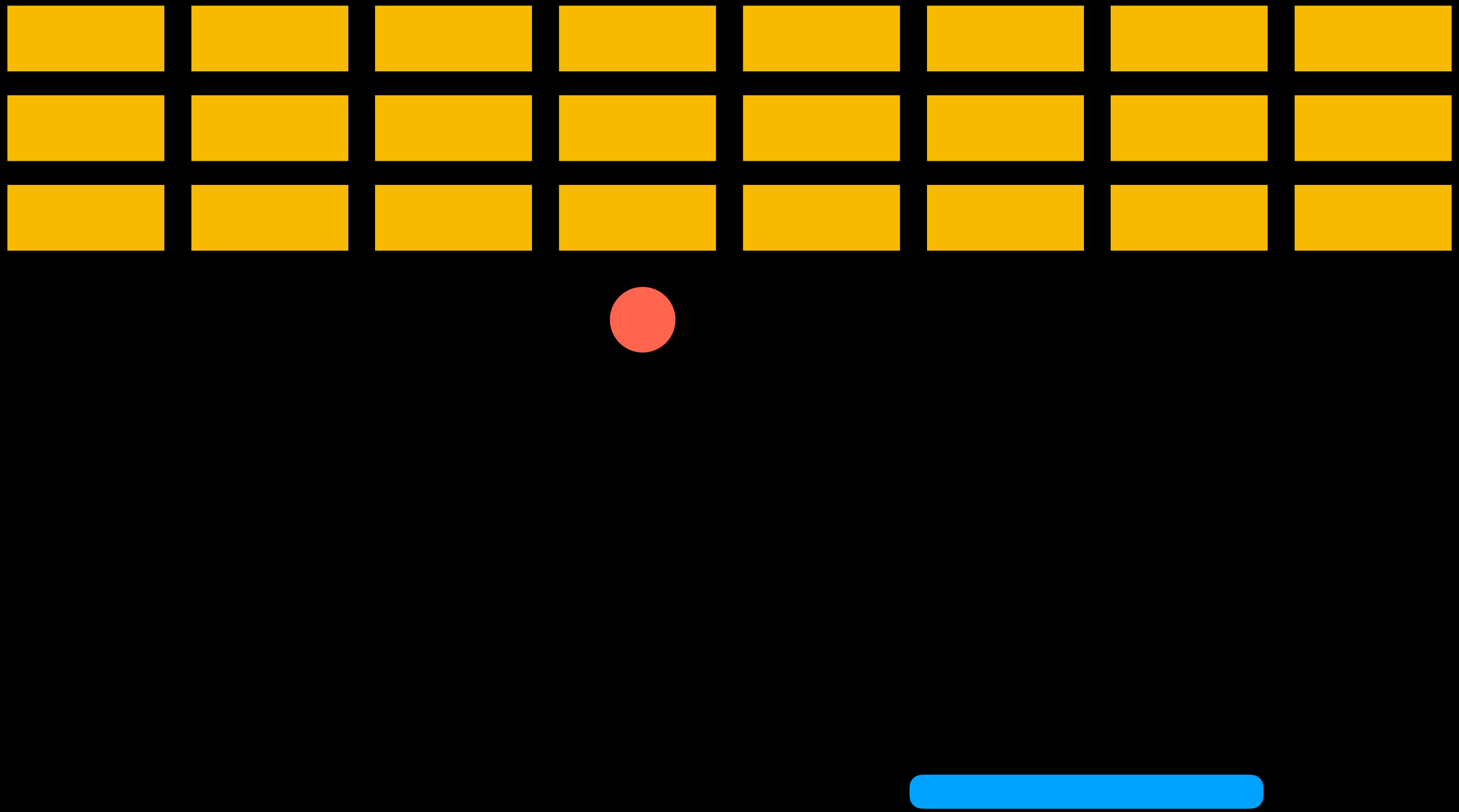


Watch History 

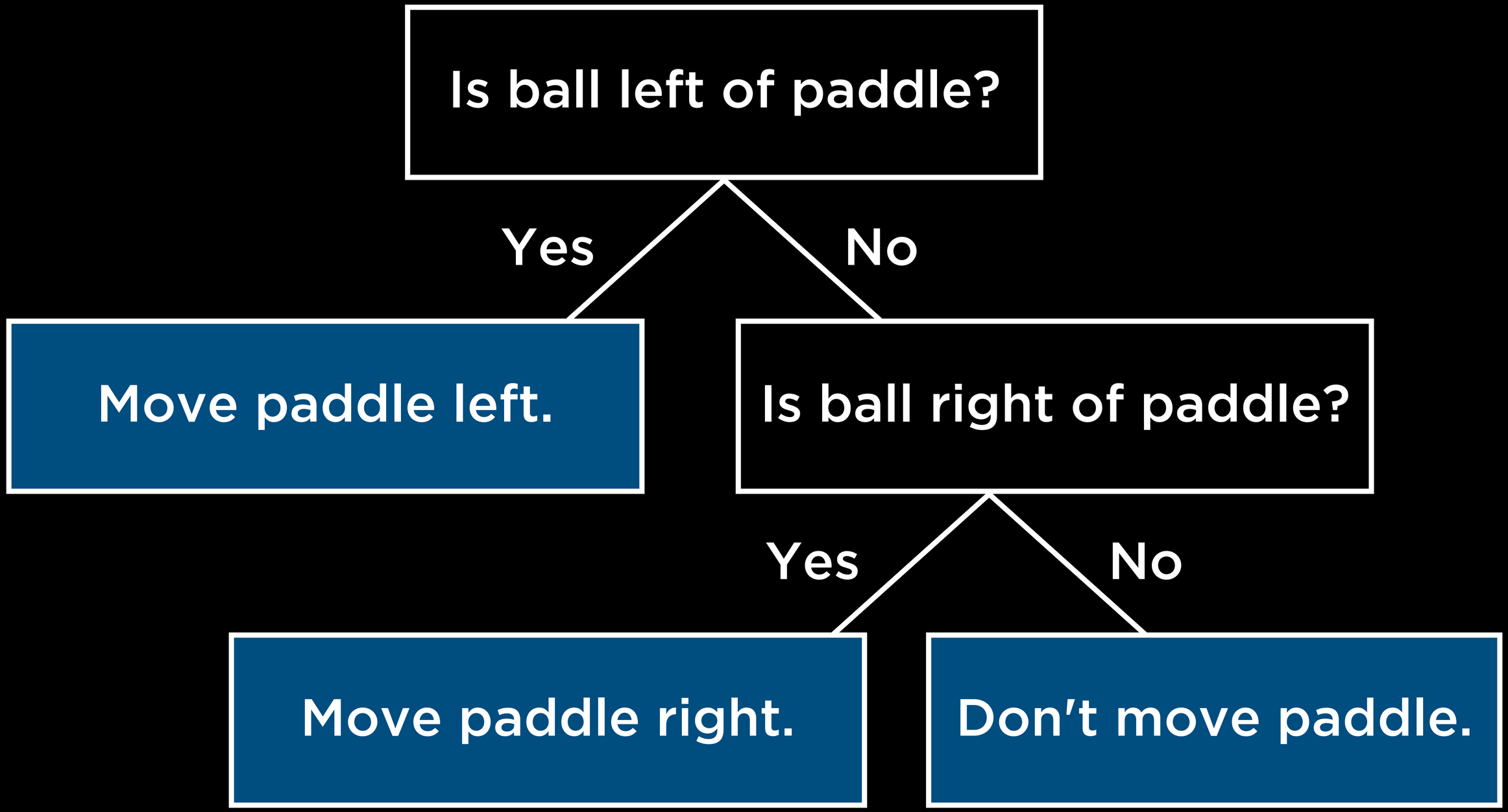
Recommended 

Artificial Intelligence

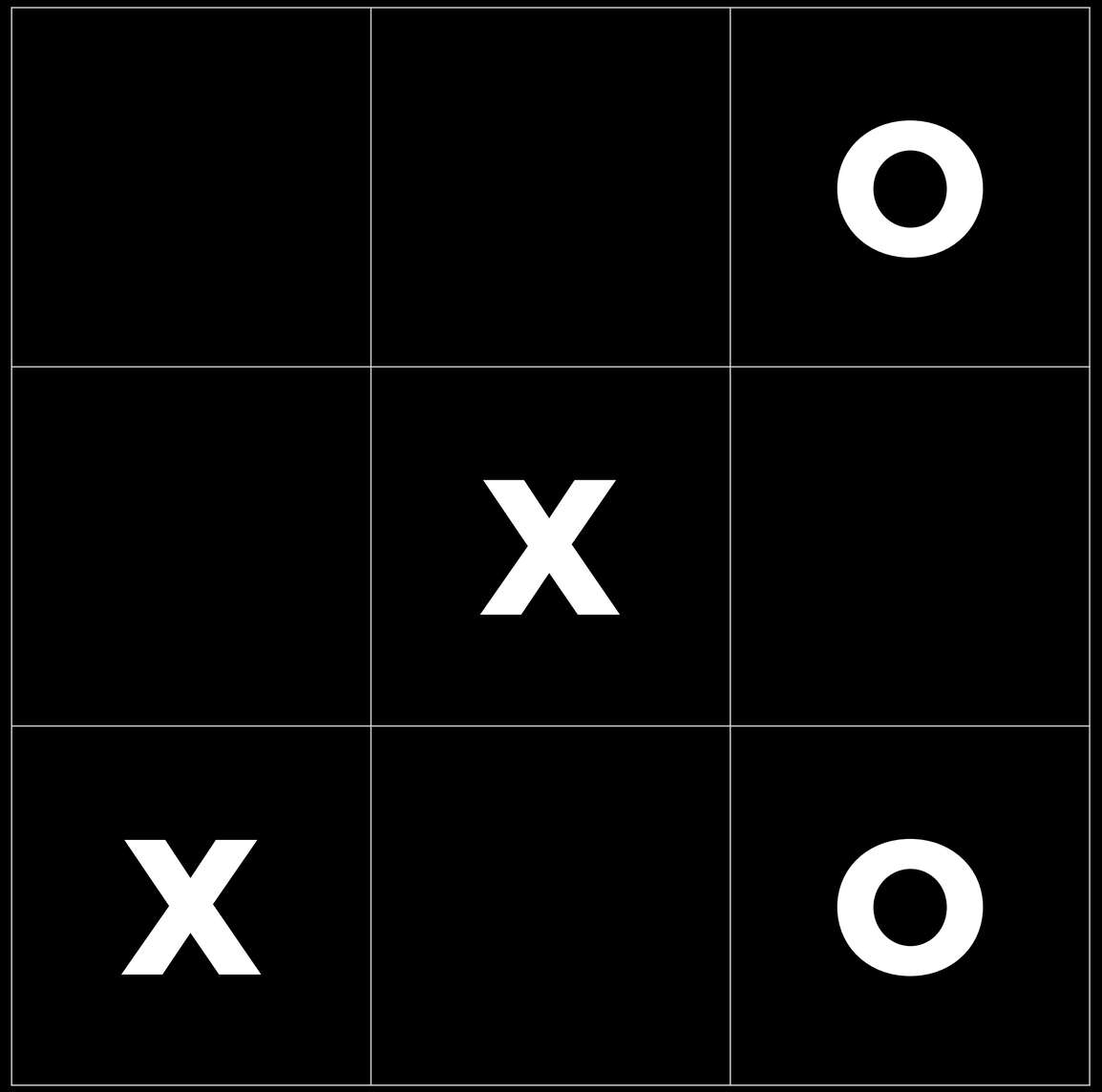
Decision-Making

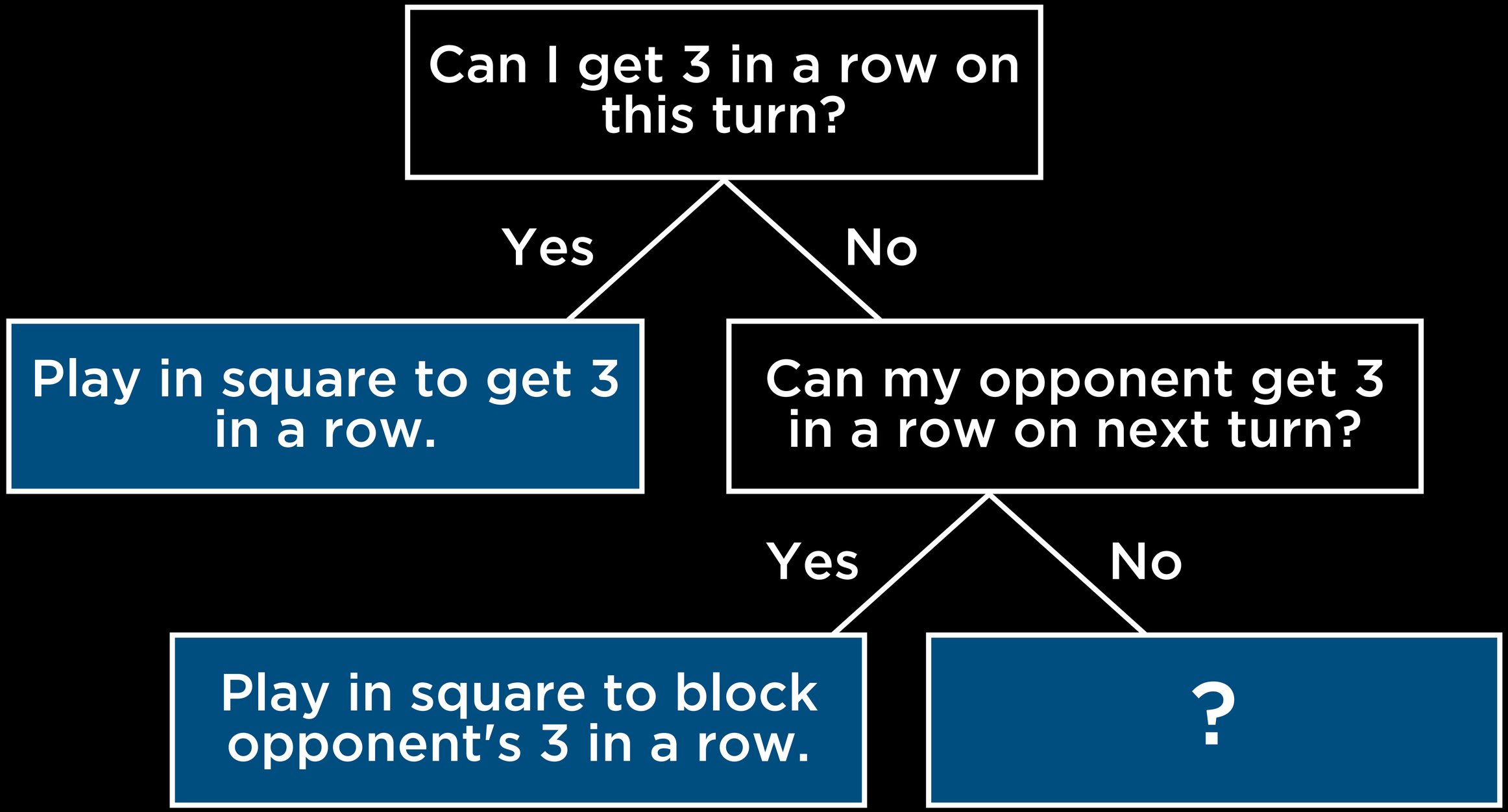


Decision Trees


Is ball left of paddle?
Yes
No
Move paddle left.
Is ball right of paddle?
Yes
No
Move paddle right.
Don't move paddle.

```
while game is ongoing:
    if ball left of paddle:
        move paddle left
    else if ball right of paddle:
        move paddle right
    else:
        don't move paddle
```

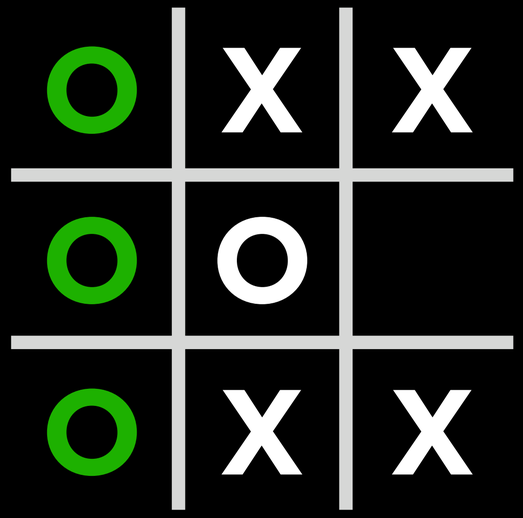

O
X
X
O


Can I get 3 in a row on this turn?
Yes
No
Play in square to get 3 in a row.
Can my opponent get 3 in a row on next turn?
Yes
No
Play in square to block opponent's 3 in a row.
?

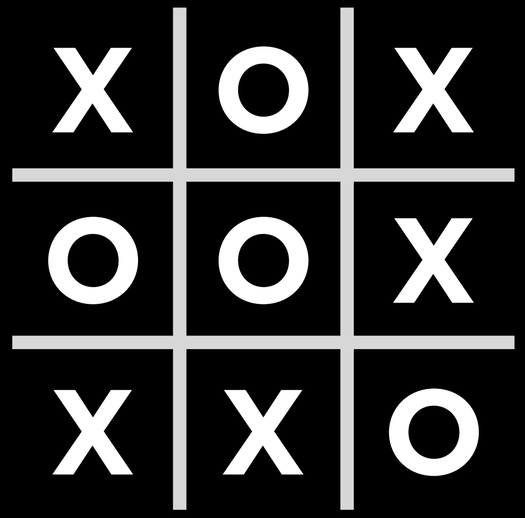
Optimal Decision-Making

Minimax

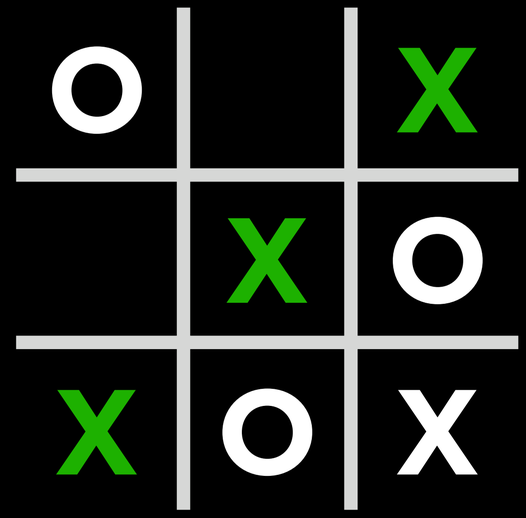
- MAX (X) aims to maximize score.
- MIN (O) aims to minimize score.



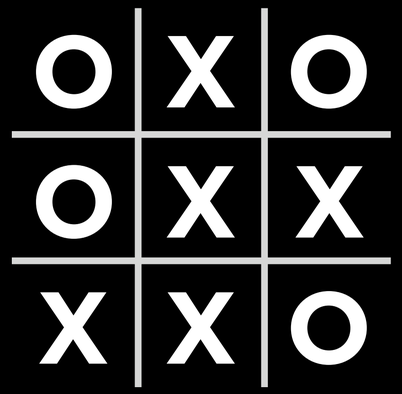
-1



0

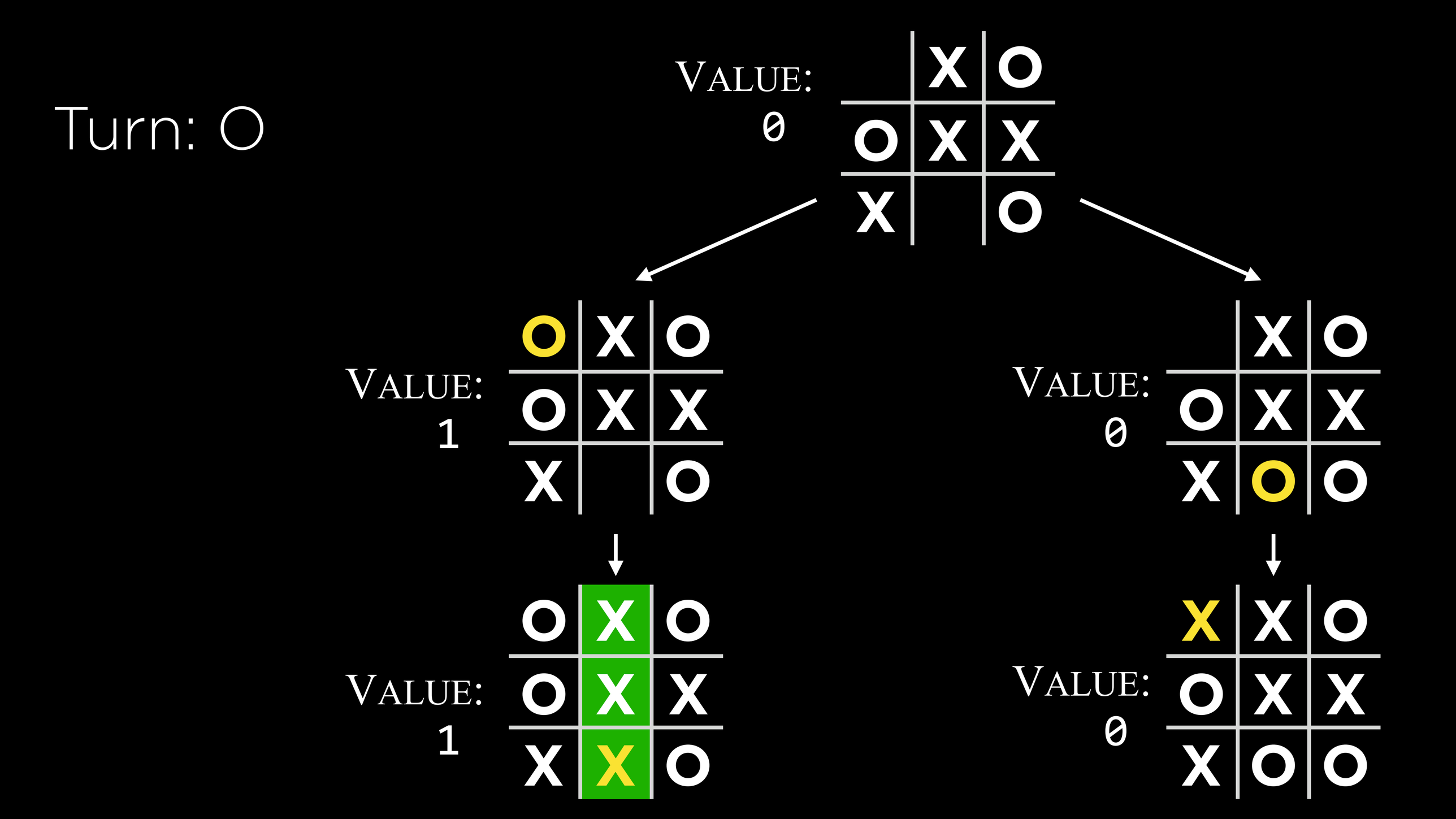


1

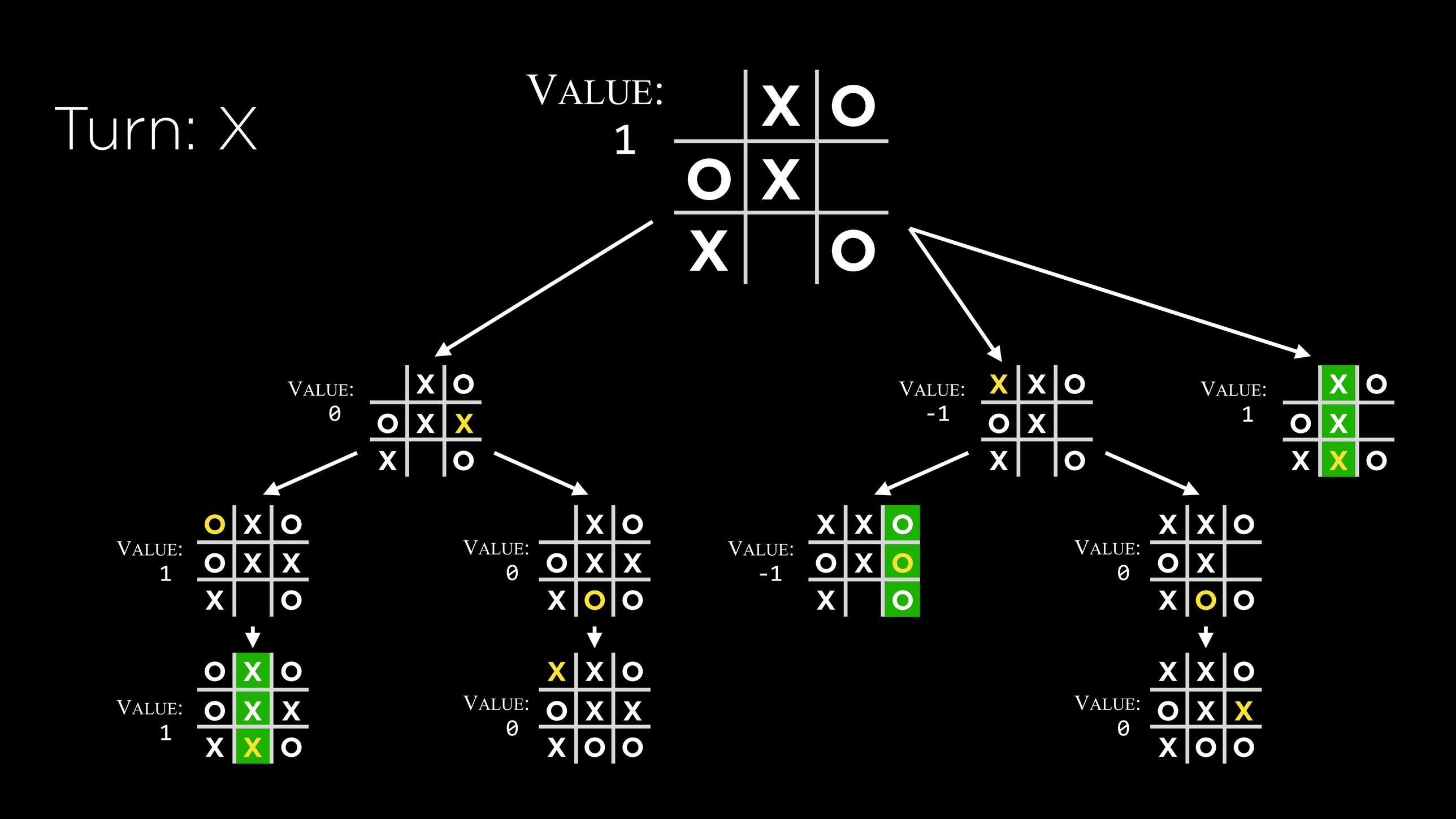


VALUE: 1

Turn: O



Turn: X


Value: 1
Value: 0
Value: -1
Value: 1
Value: 1
Value: 0
Value: -1
Value: 0
Value: 1
Value: 0

Minimax

```
if player is X:
    for all possible moves:
        calculate score for board
    choose move with highest score

else:
    for all possible moves:
        calculate score for board
    choose move with lowest score
```

255,168

total possible Tic-Tac-Toe games

288,000,000,000

total possible chess games
after four moves each

10^{29000}

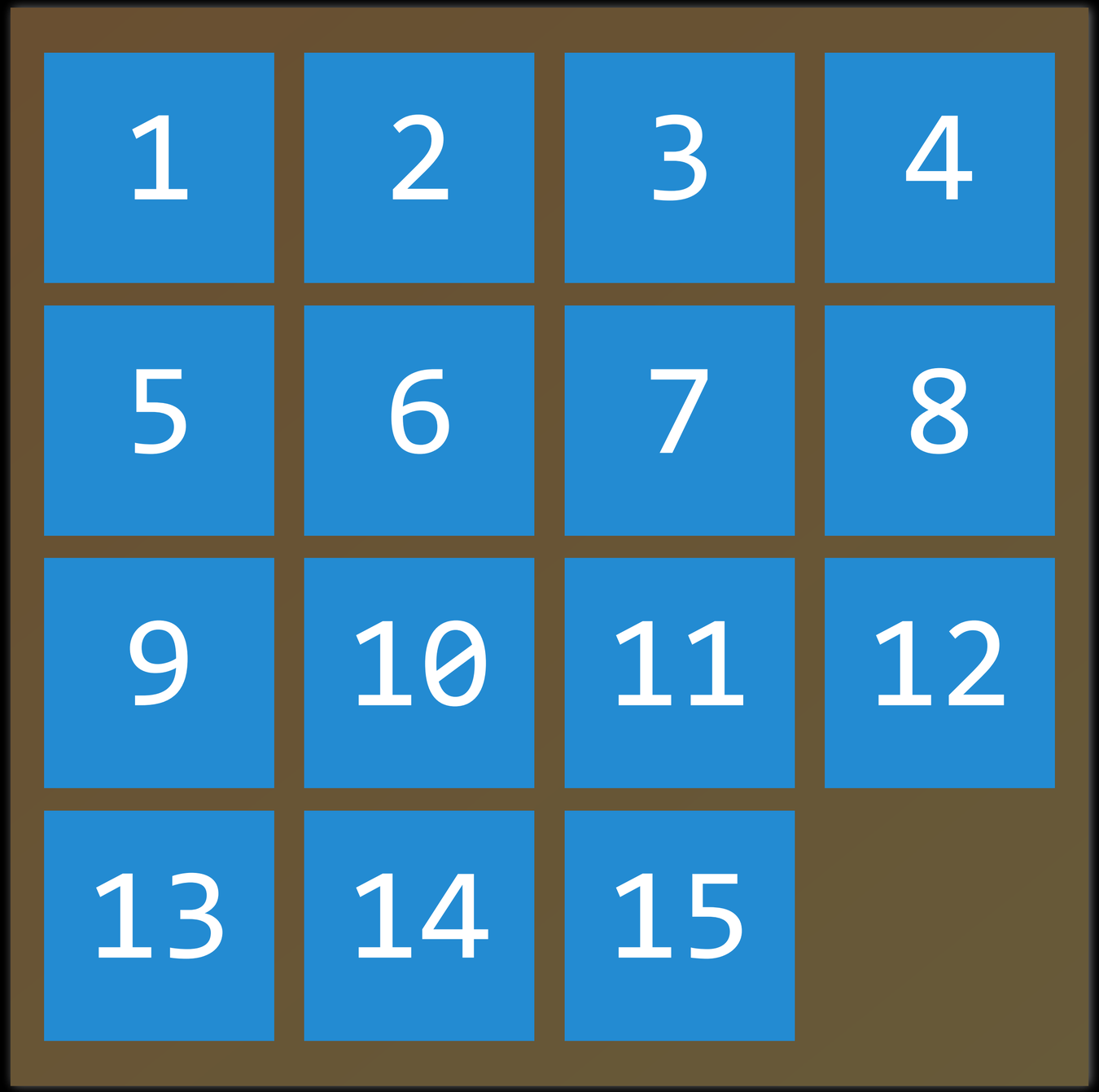
total possible chess games
(lower bound)

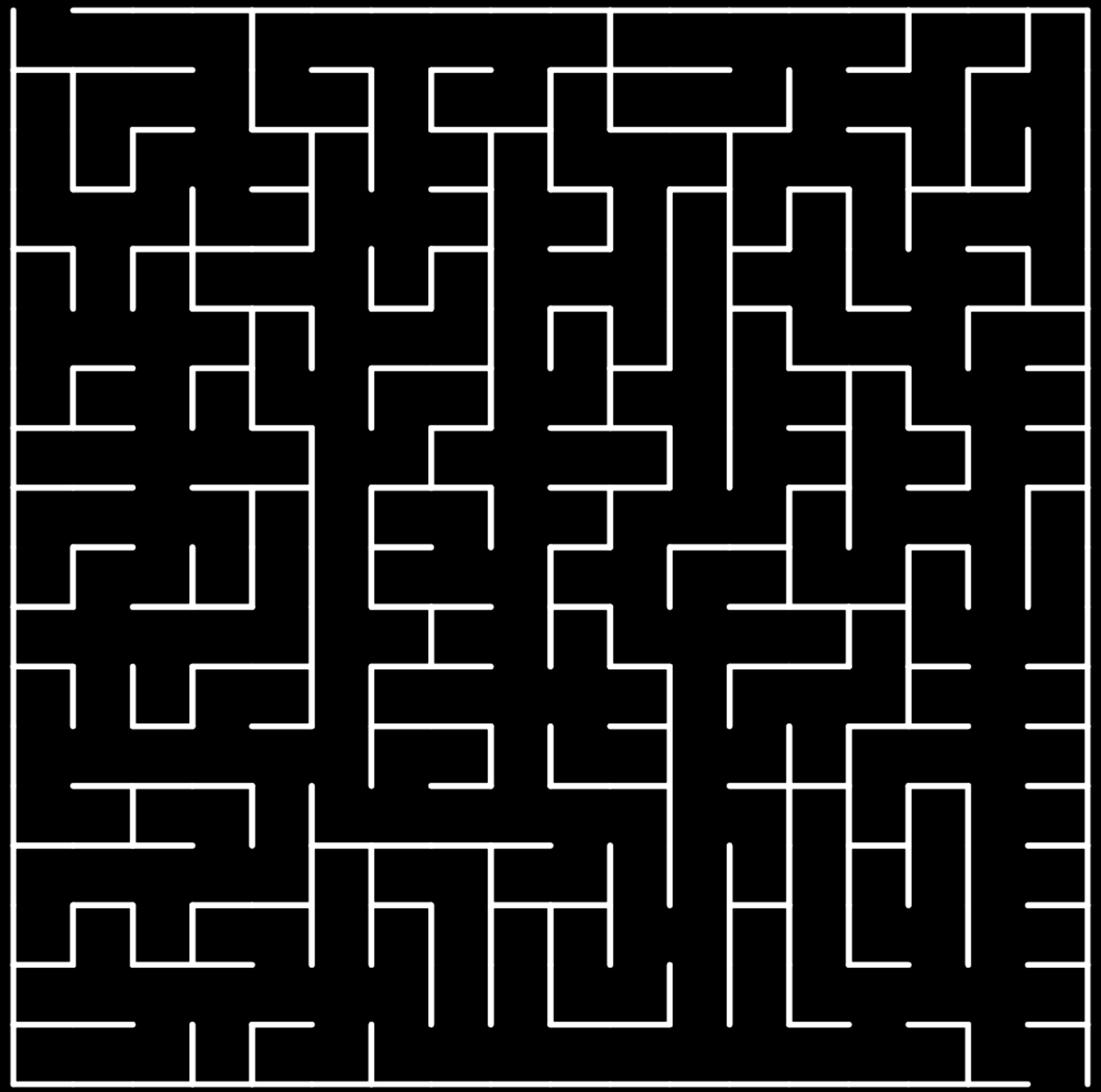
Depth-Limited Minimax

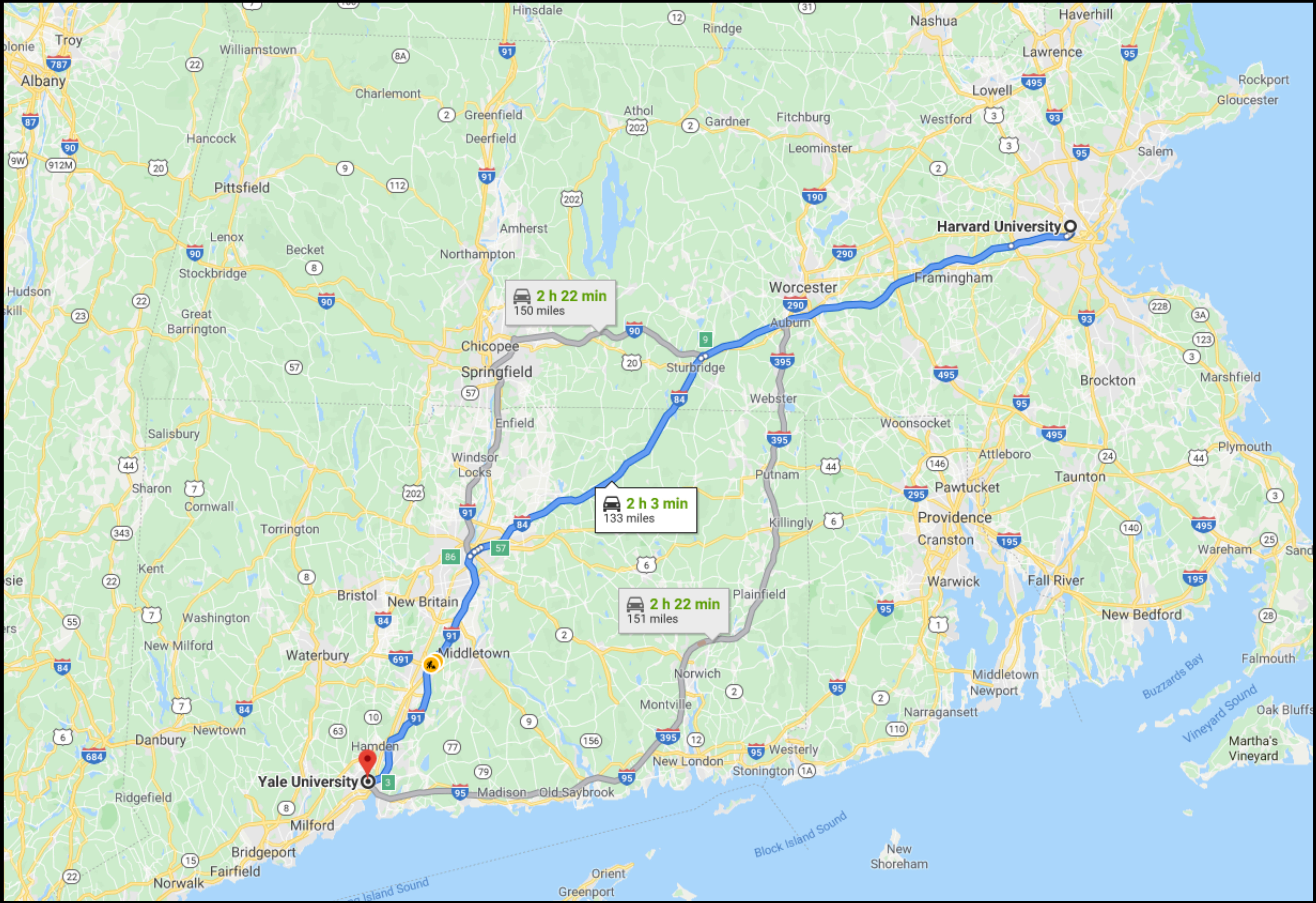
evaluation function

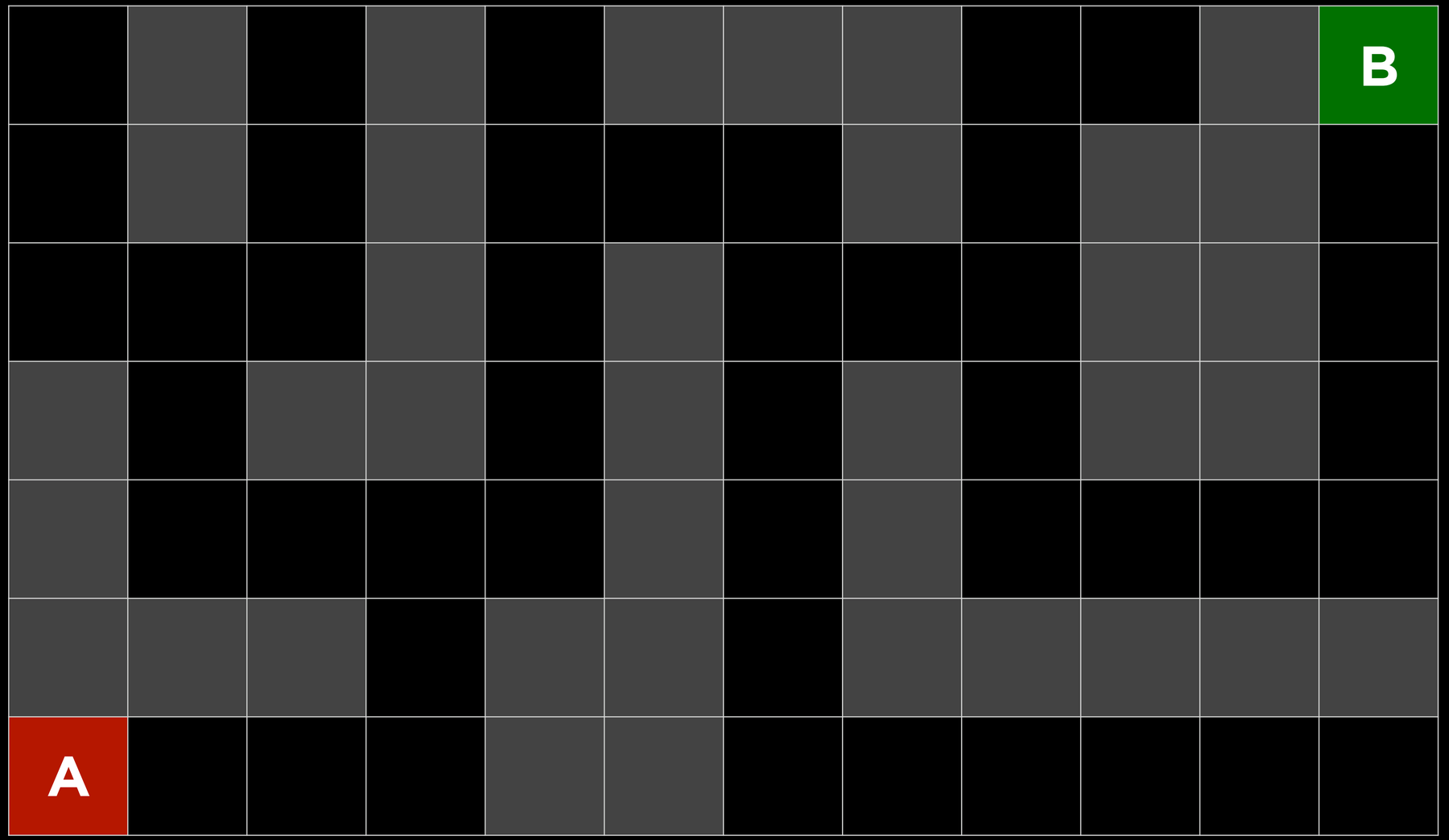
function that estimates the expected utility of the game from a given state

Search


1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

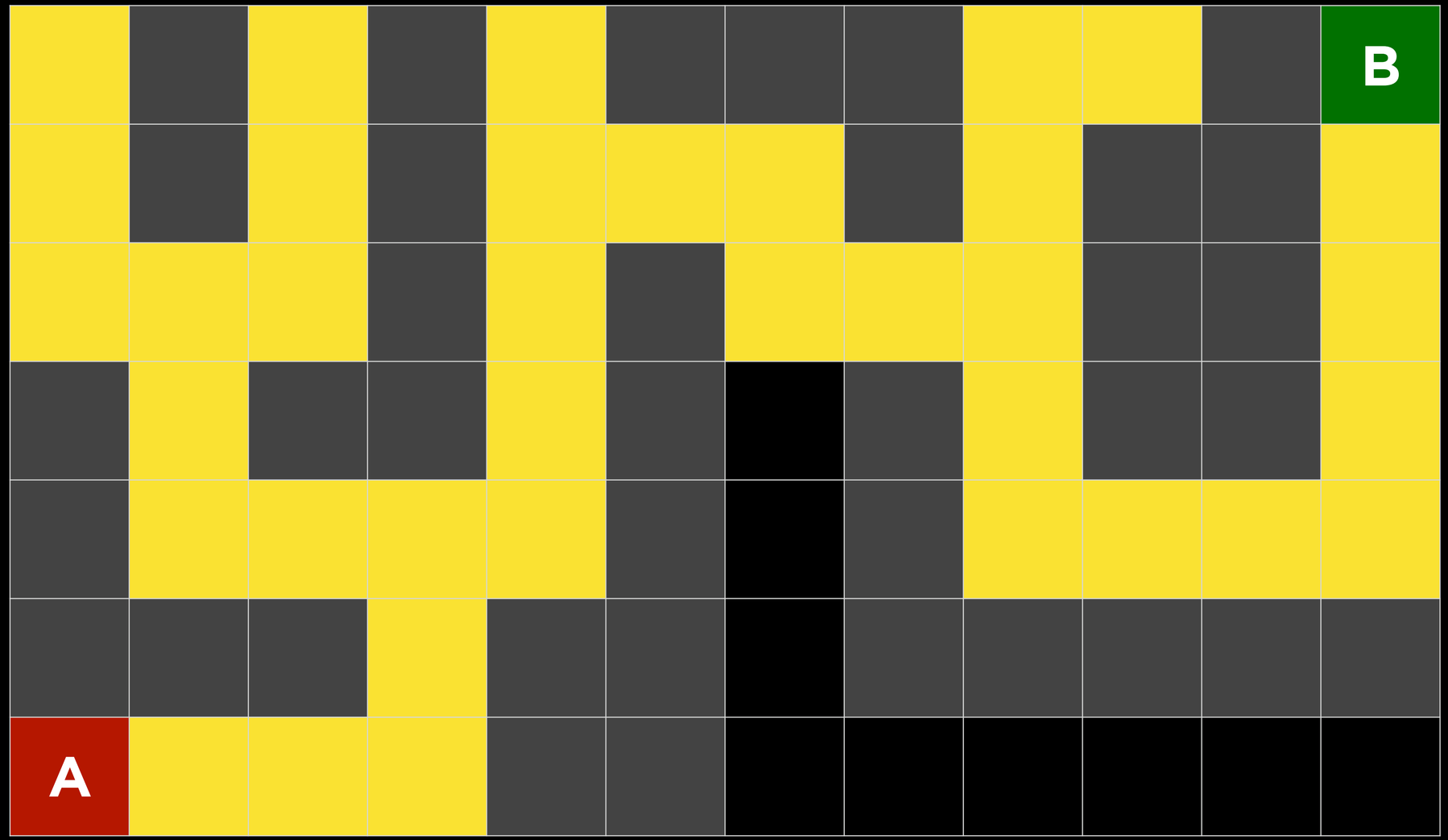



Harvard University
Yale University
2 h 22 min
150 miles
2 h 3 min
133 miles
2 h 22 min
151 miles
Albany
Troy
Williamstown
Hinsdale
Ringe
Nashua
Haverhill
Lawrence
Lowell
Rockport
Gloucester
Charlemont
Greenfield
Deerfield
Athol
Gardner
Fitchburg
Leominster
Westford
Salem
Hancock
Pittsfield
Amherst
Northampton
Lenox
Becket
Stockbridge
Worcester
Framingham
Great Barrington
Chicopee
Springfield
Auburn
Sturbridge
Webster
Brockton
Marshfield
Hudson
Salisbury
Enfield
Windsor Locks
Woonsocket
Attleboro
Taunton
Plymouth
Sharon
Cornwall
Torrington
Putnam
Pawtucket
Providence
Cranston
Killingly
Kent
Bristol
New Britain
Warwick
Fall River
Wareham
Washington
Plainfield
New Bedford
New Milford
Waterbury
Middletown
Norwich
Falmouth
Middletown
Newport
Buzzards Bay
Newtown
Montville
Narragansett
Oak Bluffs
Vineyard Sound
Martha's Vineyard
Danbury
Hamden
New London
Westerly
Stonington
Ridgefield
Madison
Old Saybrook
Milford
Bridgeport
Fairfield
Norwalk
Block Island Sound
New Shoreham
Orient
Greenport

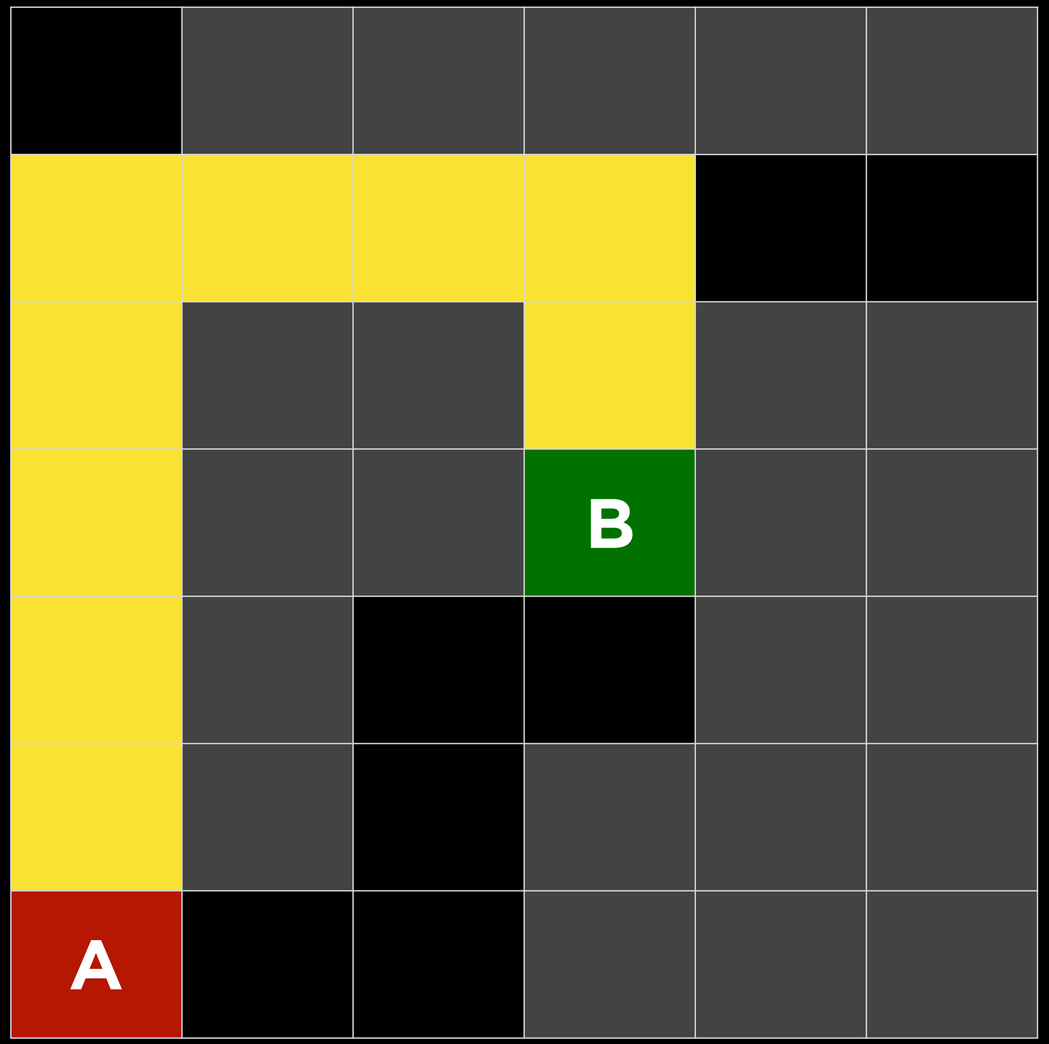

B
A

Depth-First Search

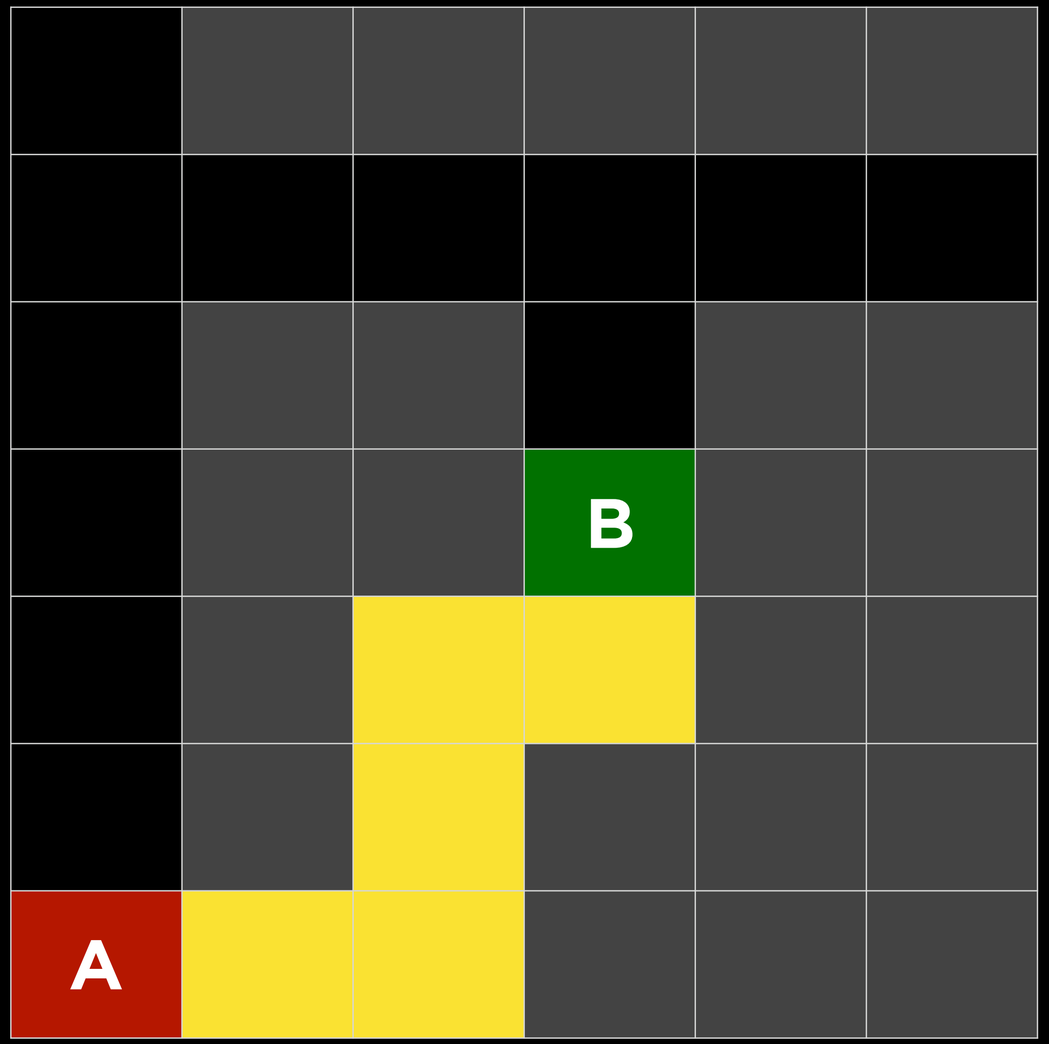
Depth-First Search



Depth-First Search

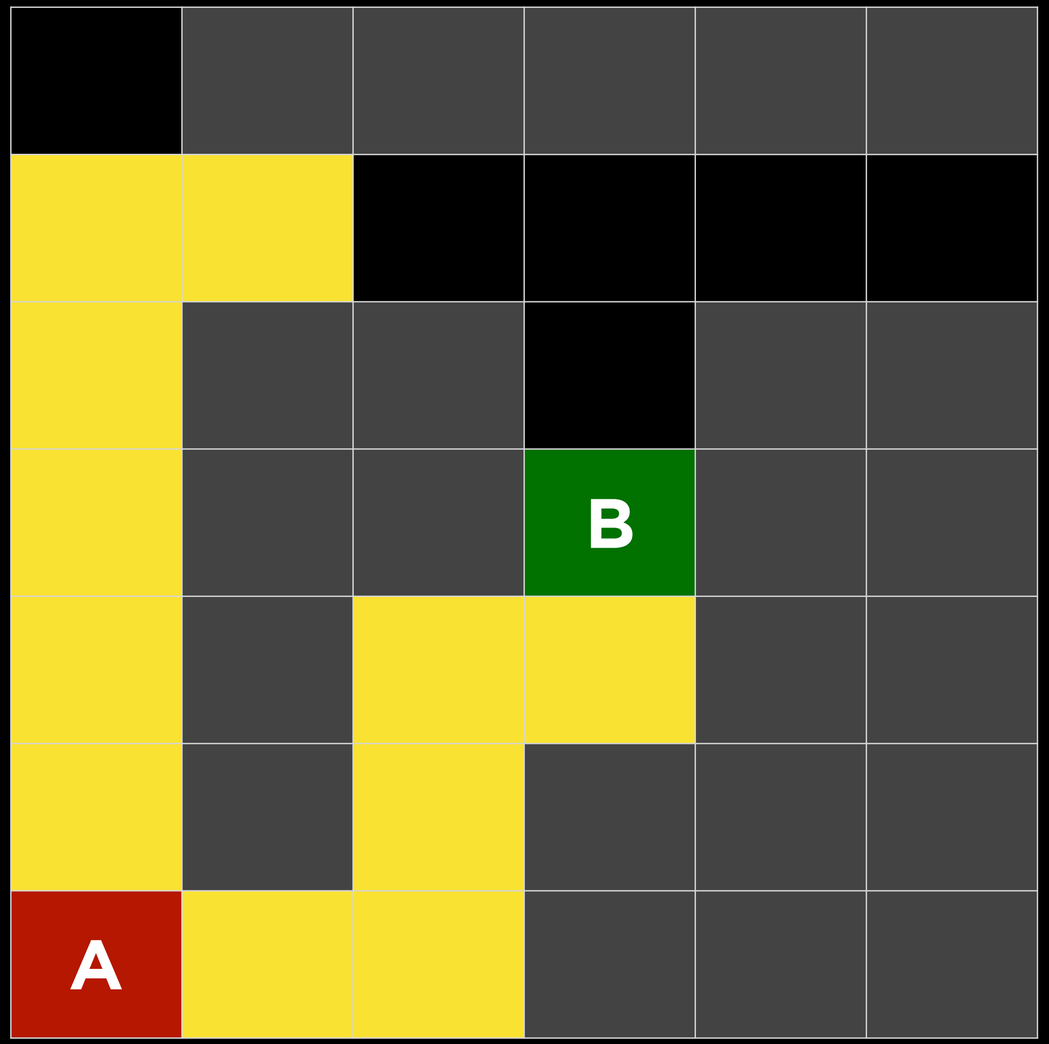


Depth-First Search

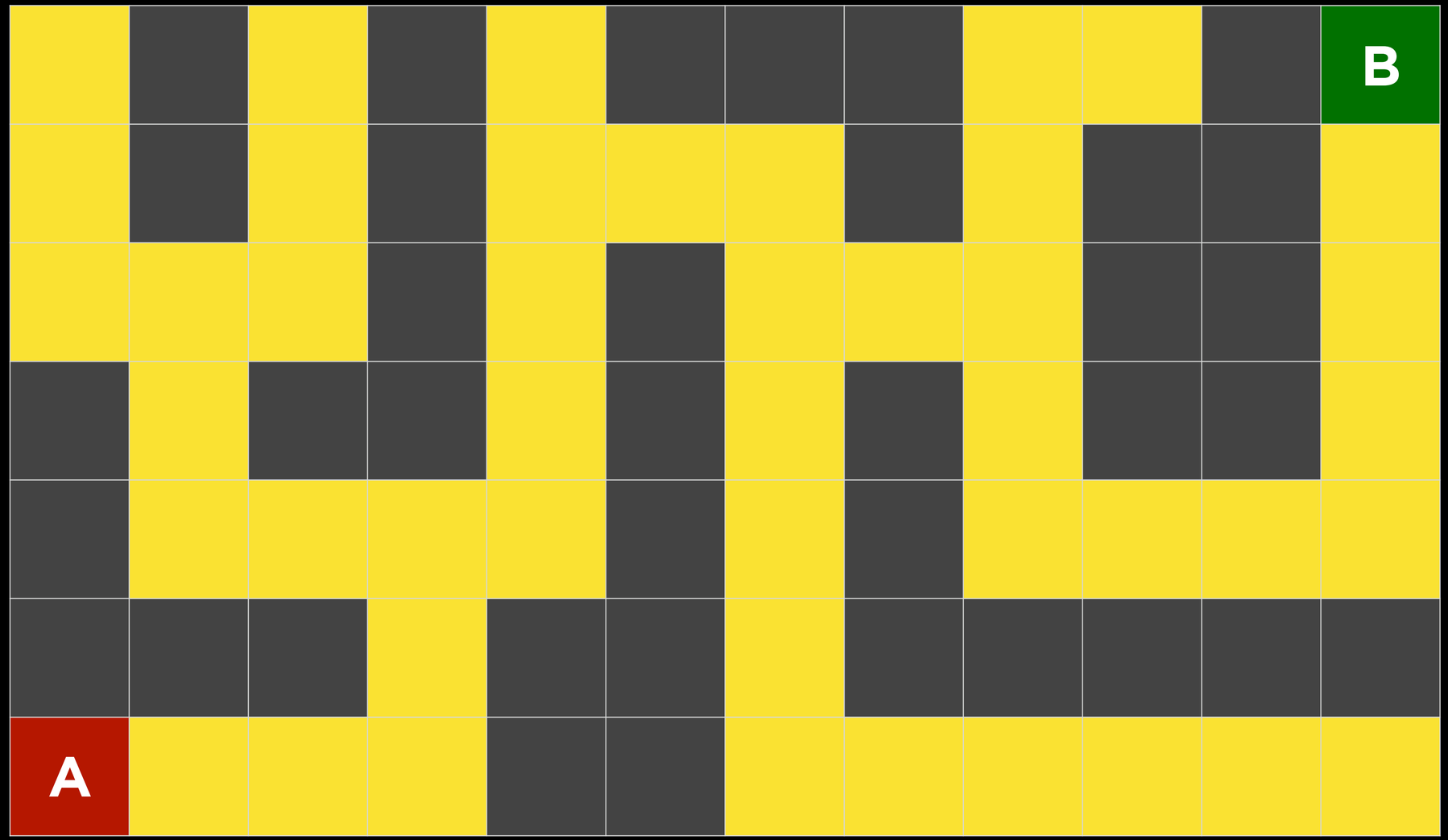


Breadth-First Search

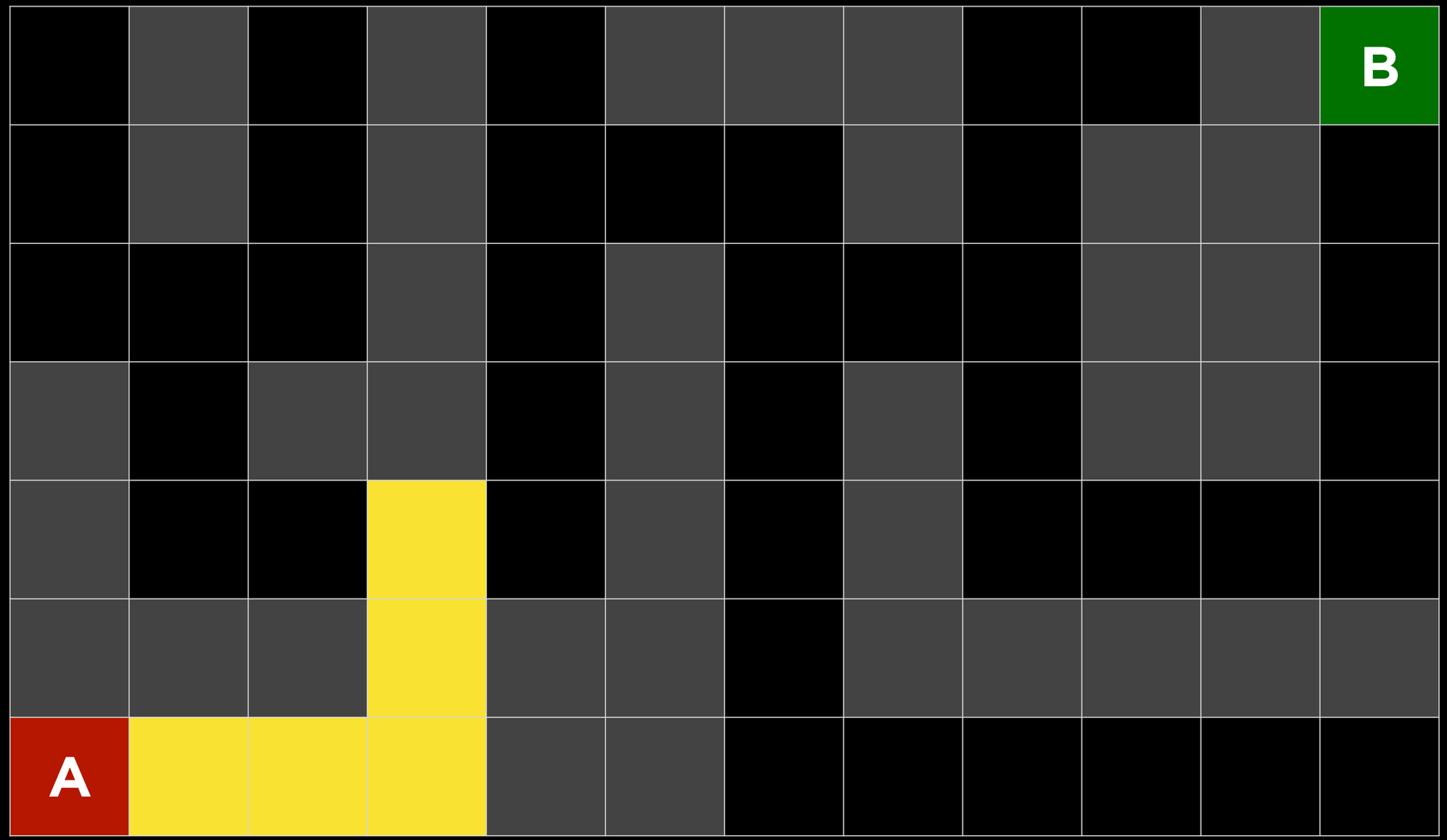
Breadth-First Search



Breadth-First Search



Breadth-First Search



uninformed search

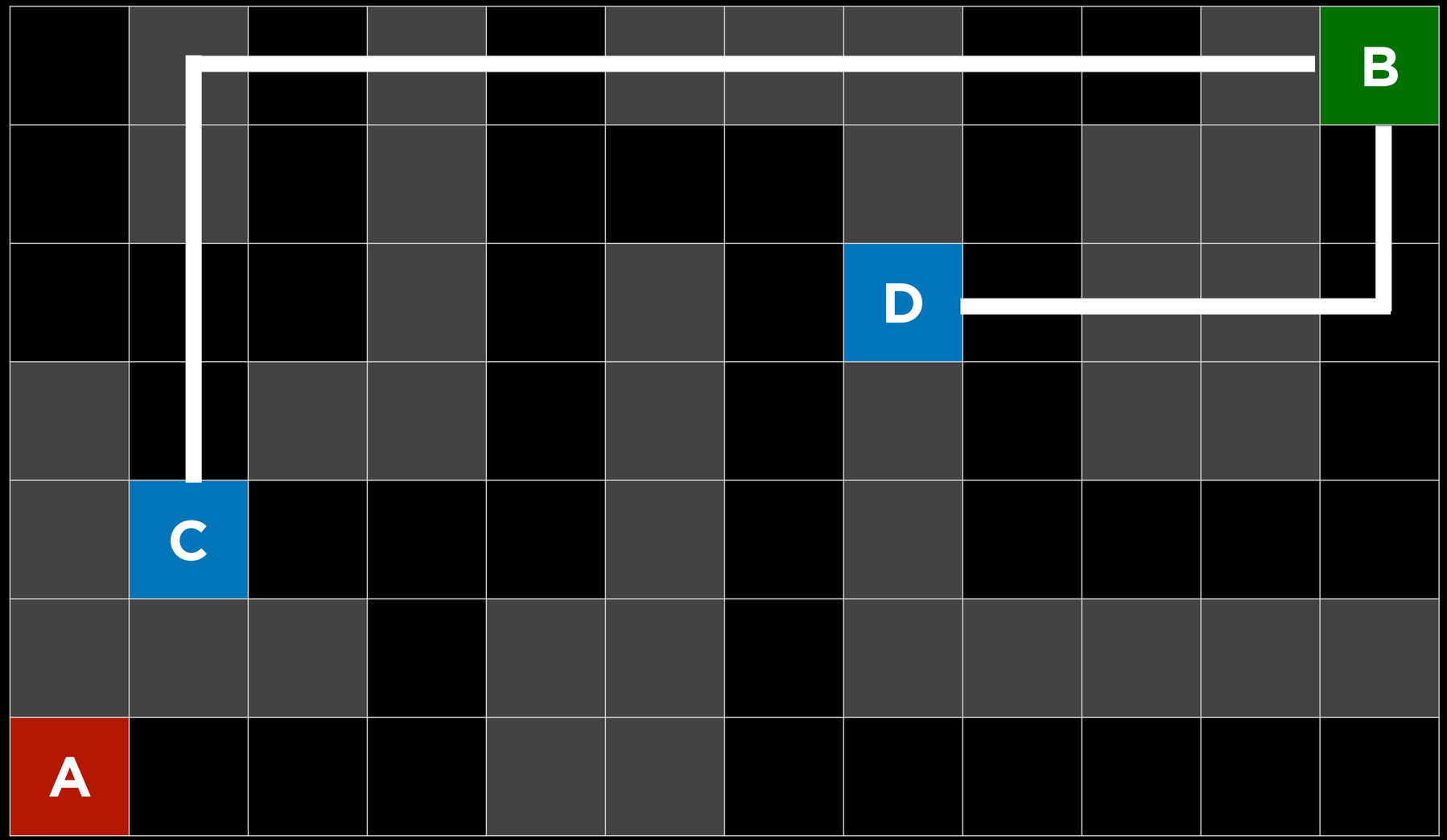
informed search

search strategy that uses problem-specific knowledge to find solutions more efficiently

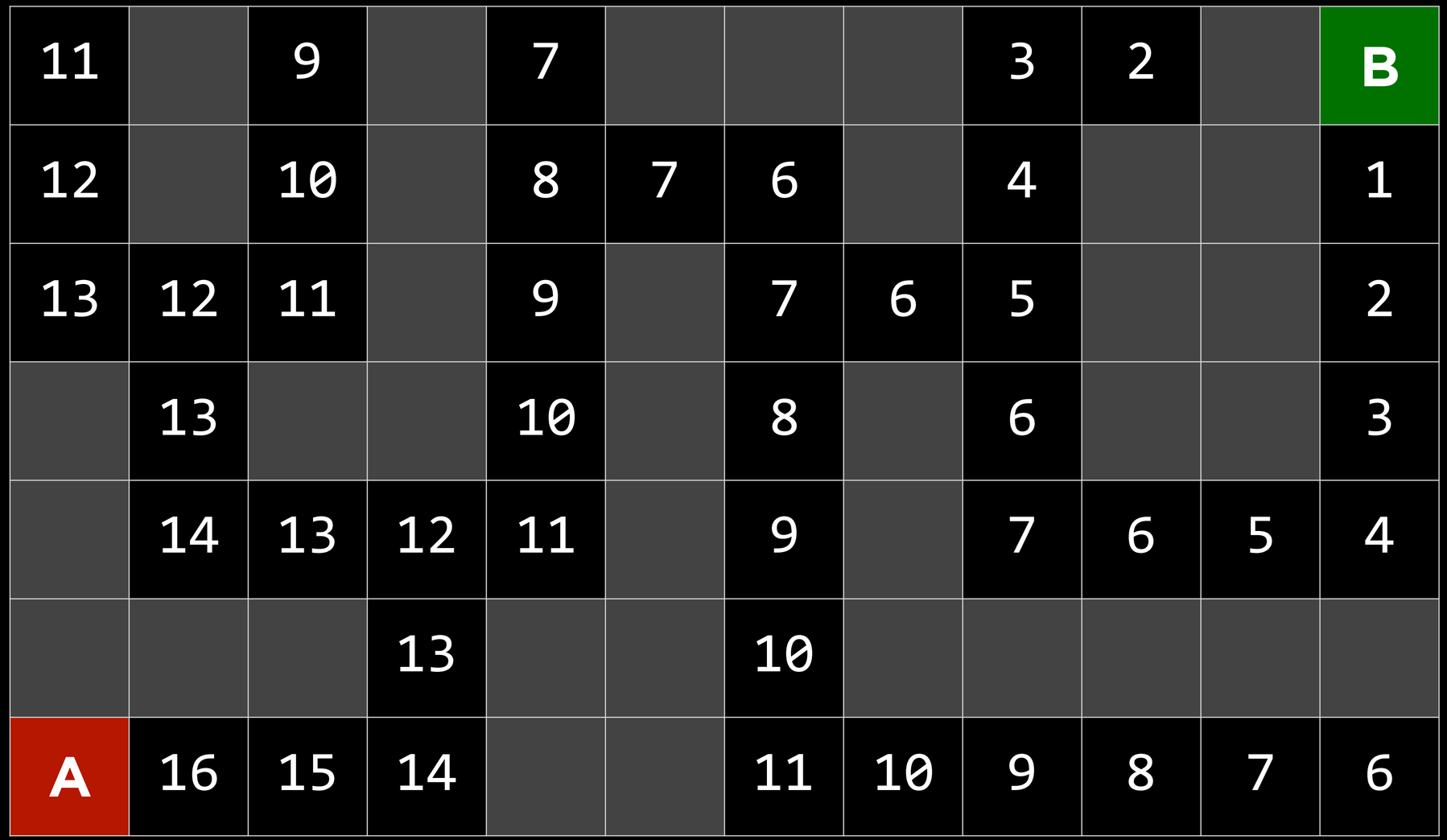
greedy best-first search

search algorithm that chooses what to explore based on a heuristic

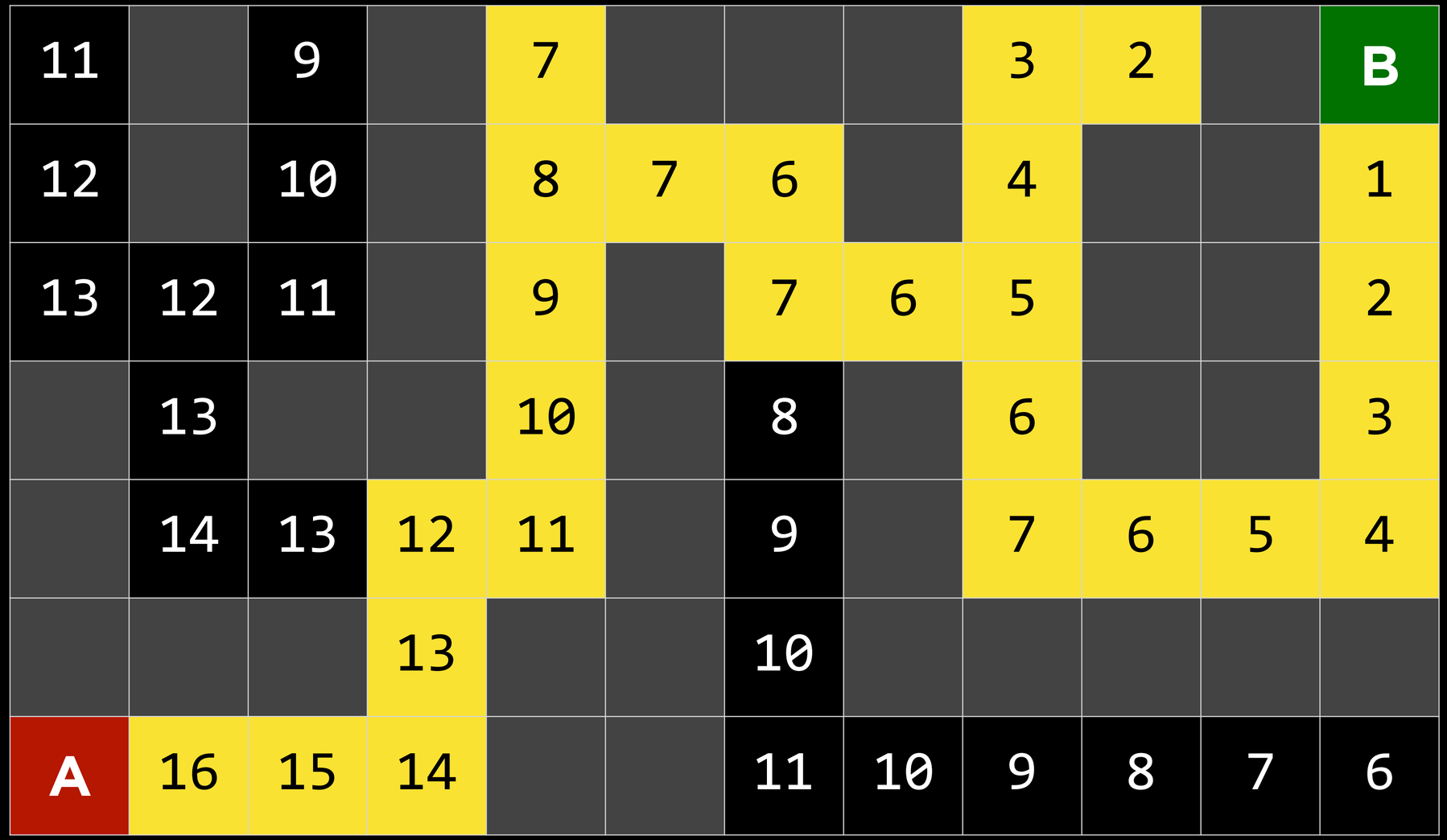
Heuristic function? Manhattan distance.



Greedy Best-First Search



Greedy Best-First Search



Greedy Best-First Search

	10	9	8	7	6	5	4	3	2	1	**B**
	11										1
	12		10	9	8	7	6	5	4		2
	13		11						5		3
	14	13	12		10	9	8	7	6		4
			13		11						5
A	16	15	14		12	11	10	9	8	7	6

Greedy Best-First Search

	10	9	8	7	6	5	4	3	2	1	**B**
	11										1
	12		10	9	8	7	6	5	4		2
	13		11						5		3
	14	13	12		10	9	8	7	6		4
			13		11						5
A	16	15	14		12	11	10	9	8	7	6

Greedy Best-First Search

	10	9	8	7	6	5	4	3	2	1	**B**
	11										1
	12		10	9	8	7	6	5	4		2
	13		11						5		3
	14	13	12		10	9	8	7	6		4
			13		11						5
A	16	15	14		12	11	10	9	8	7	6

A* search

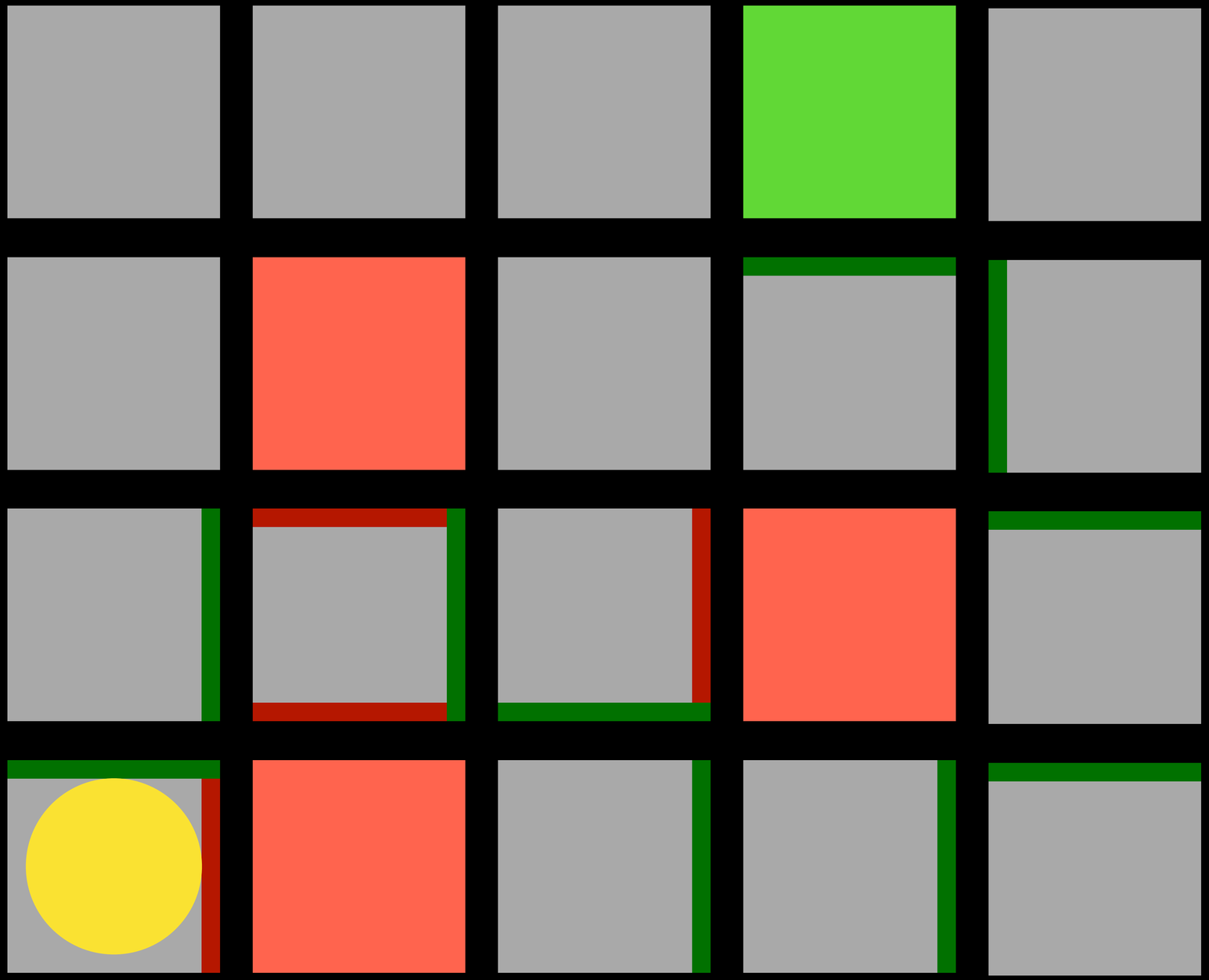
search algorithm that chooses what to explore based on both

1) heuristic
2) number of steps taken so far

A* Search

	11+10	12+9	13+8	14+7	15+6	16+5	17+4	18+3	19+2	20+1	B
	10+11										1
	9+12		7+10	8+9	9+8	10+7	11+6	12+5	13+4		2
	8+13		6+11						14+5		3
	7+14	6+13	5+12		10	9	8	7	15+6		4
			4+13		11						5
A	1+16	2+15	3+14		12	11	10	9	8	7	6

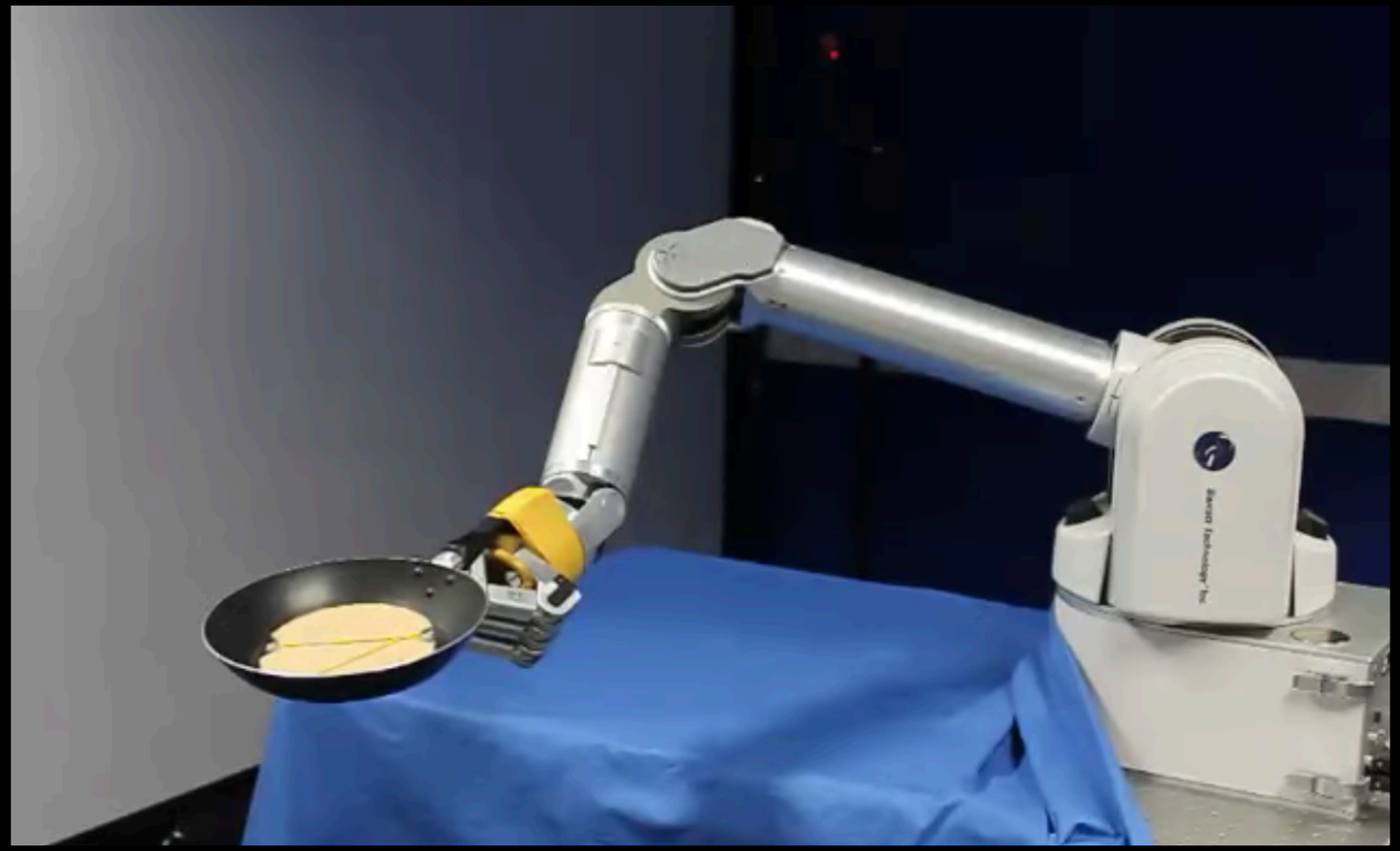
Reinforcement Learning



Explore vs. Exploit

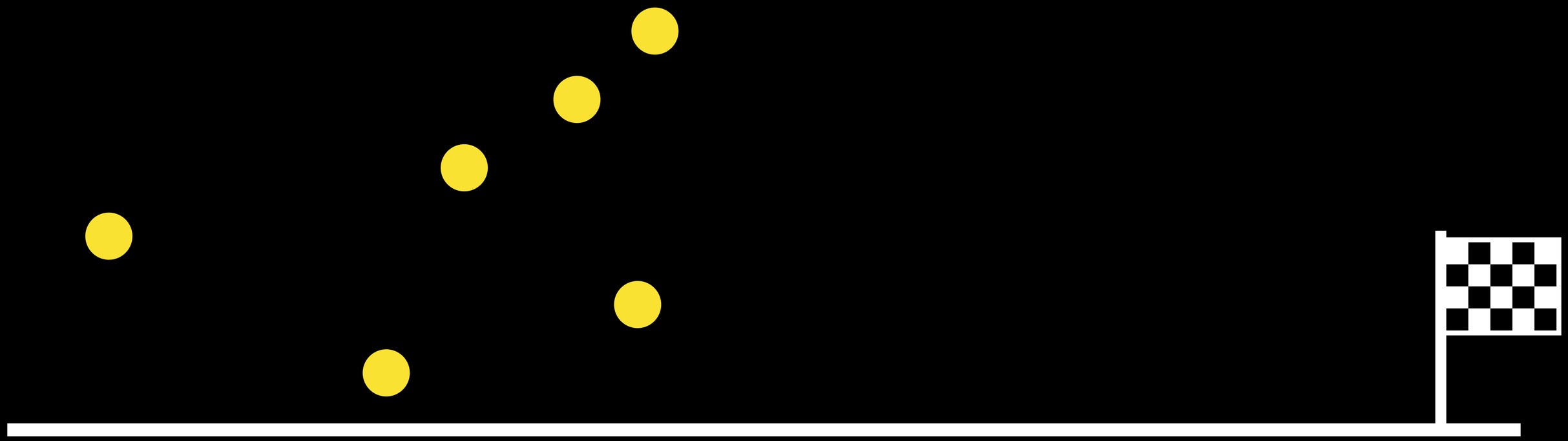
Explore vs. Exploit Strategy

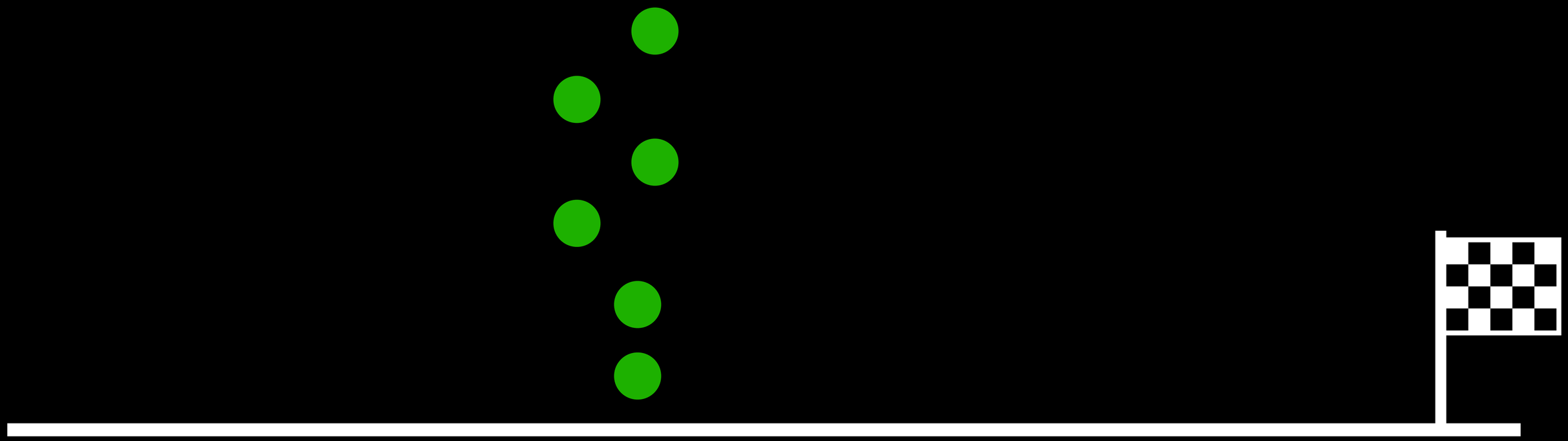
```
epsilon = 0.10

if random() < epsilon:
    make a random move
else:
    make the move with the highest value
```



https://www.youtube.com/watch?v=W_gxLKSsSIE

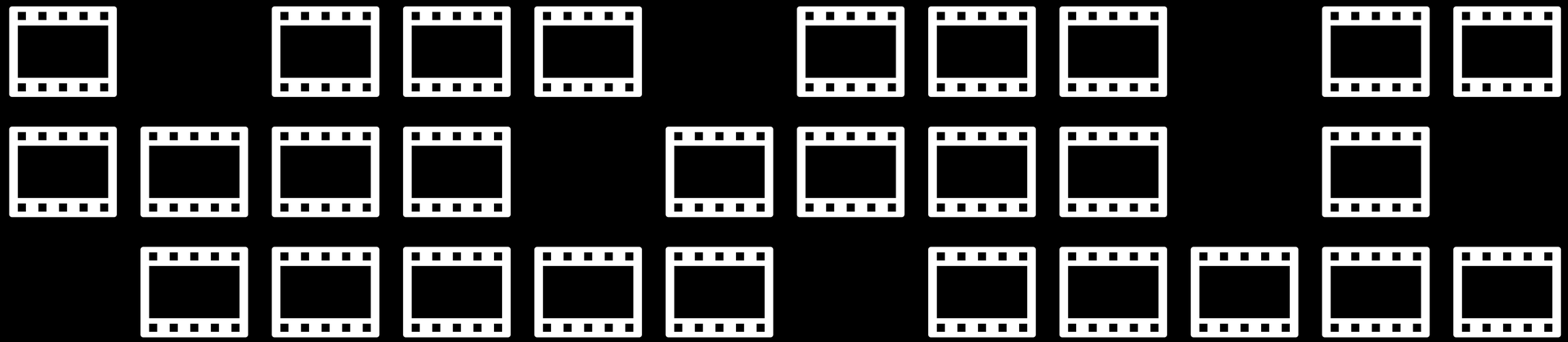
Genetic Algorithms

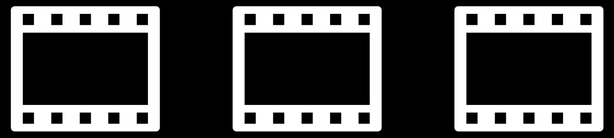




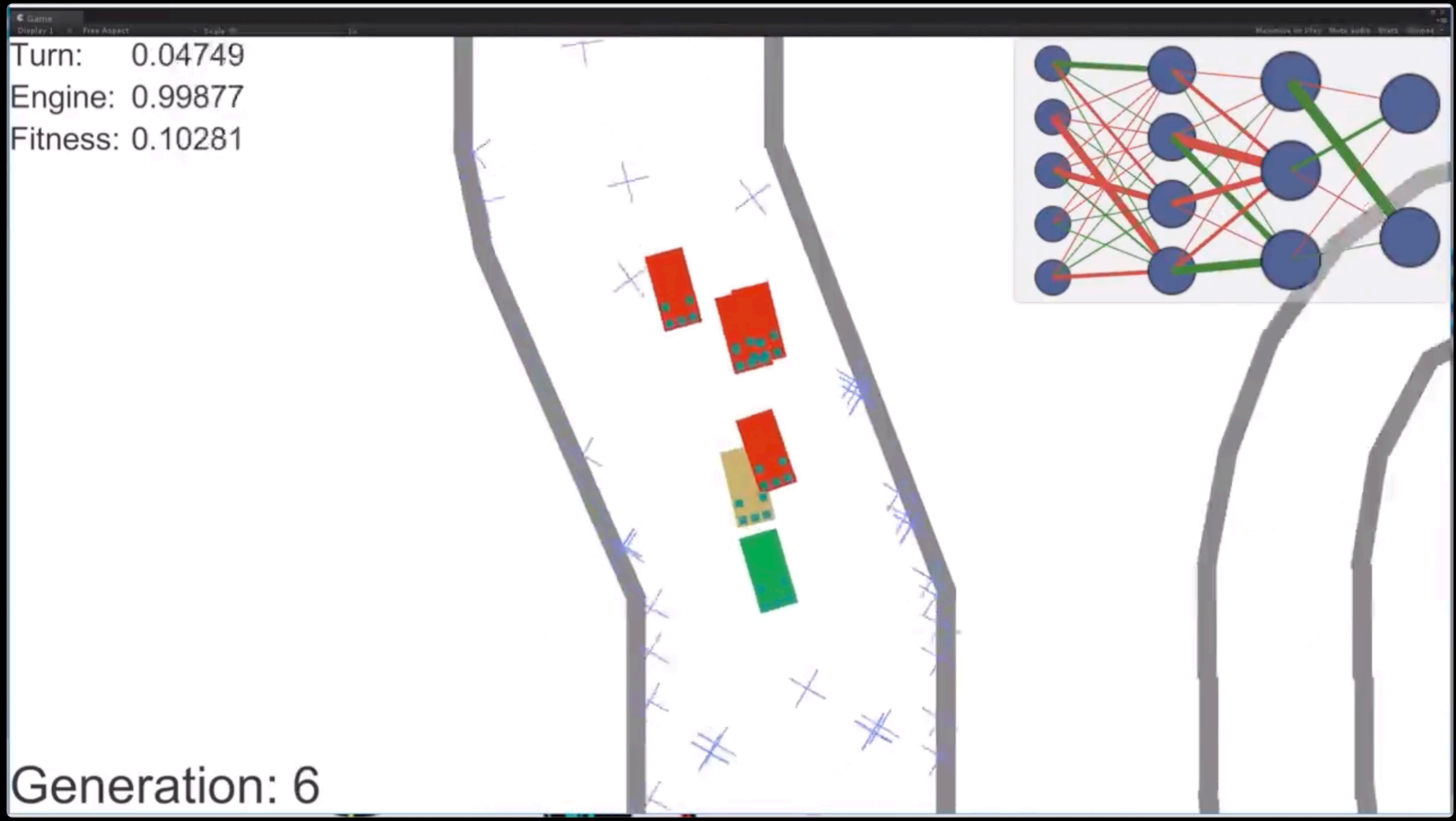
Genetic Algorithm

```
make initial generation of candidates randomly
repeat until successful:
    for each candidate:
        calculate candidate's fitness
    remove least fit candidates
    make new generation from remaining candidates
```



Watch History 

Recommended 


Turn: 0.04749
Engine: 0.99877
Fitness: 0.10281
Generation: 6

Classification

Date	**Humidity** (relative humidity)	**Pressure** (sea level, mb)	**Rain**
January 1	93%	999.7	Rain
January 2	49%	1015.5	No Rain
January 3	79%	1031.1	No Rain
January 4	65%	984.9	Rain
January 5	90%	975.2	Rain

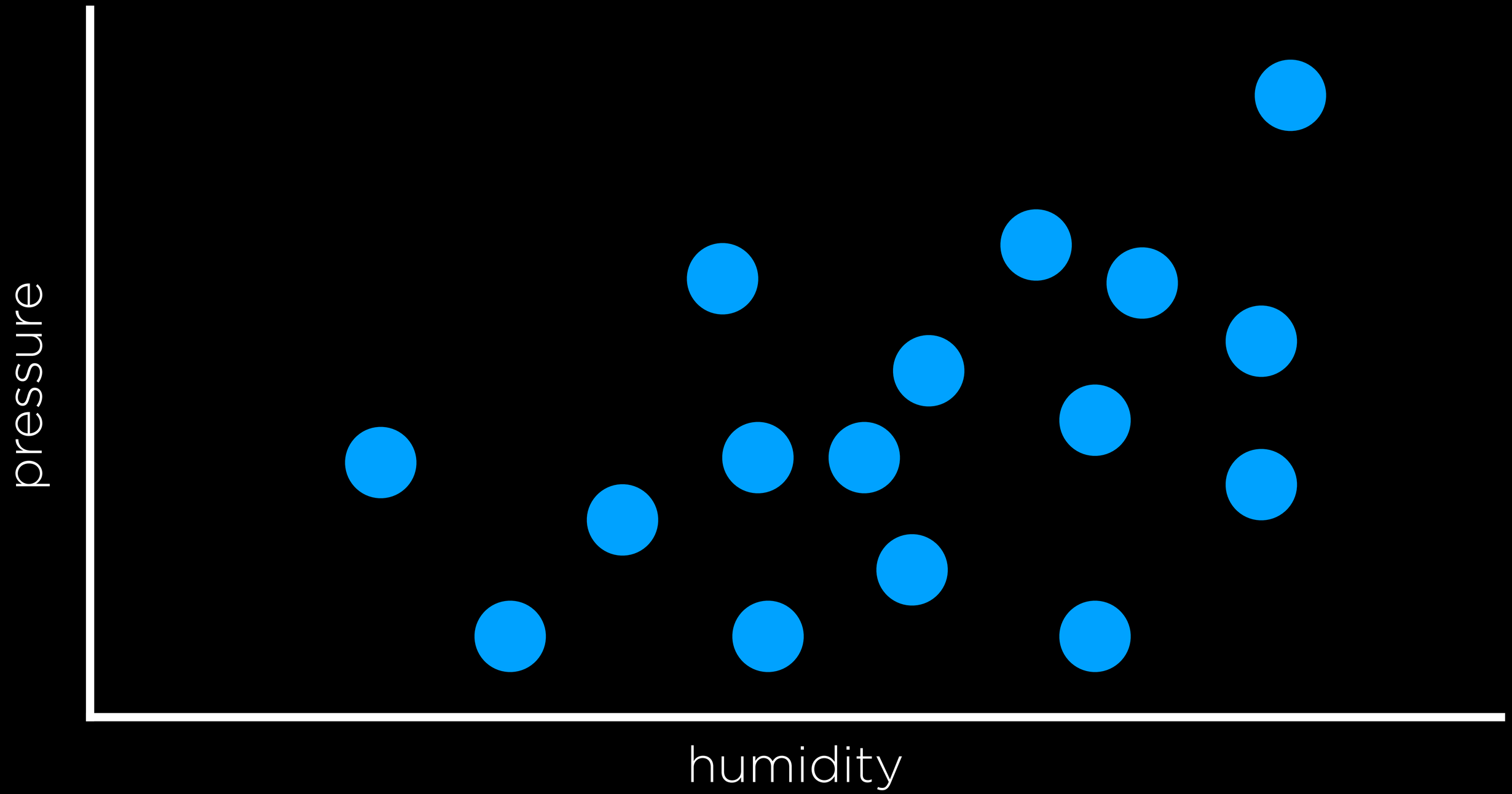
$f(humidity, pressure)$

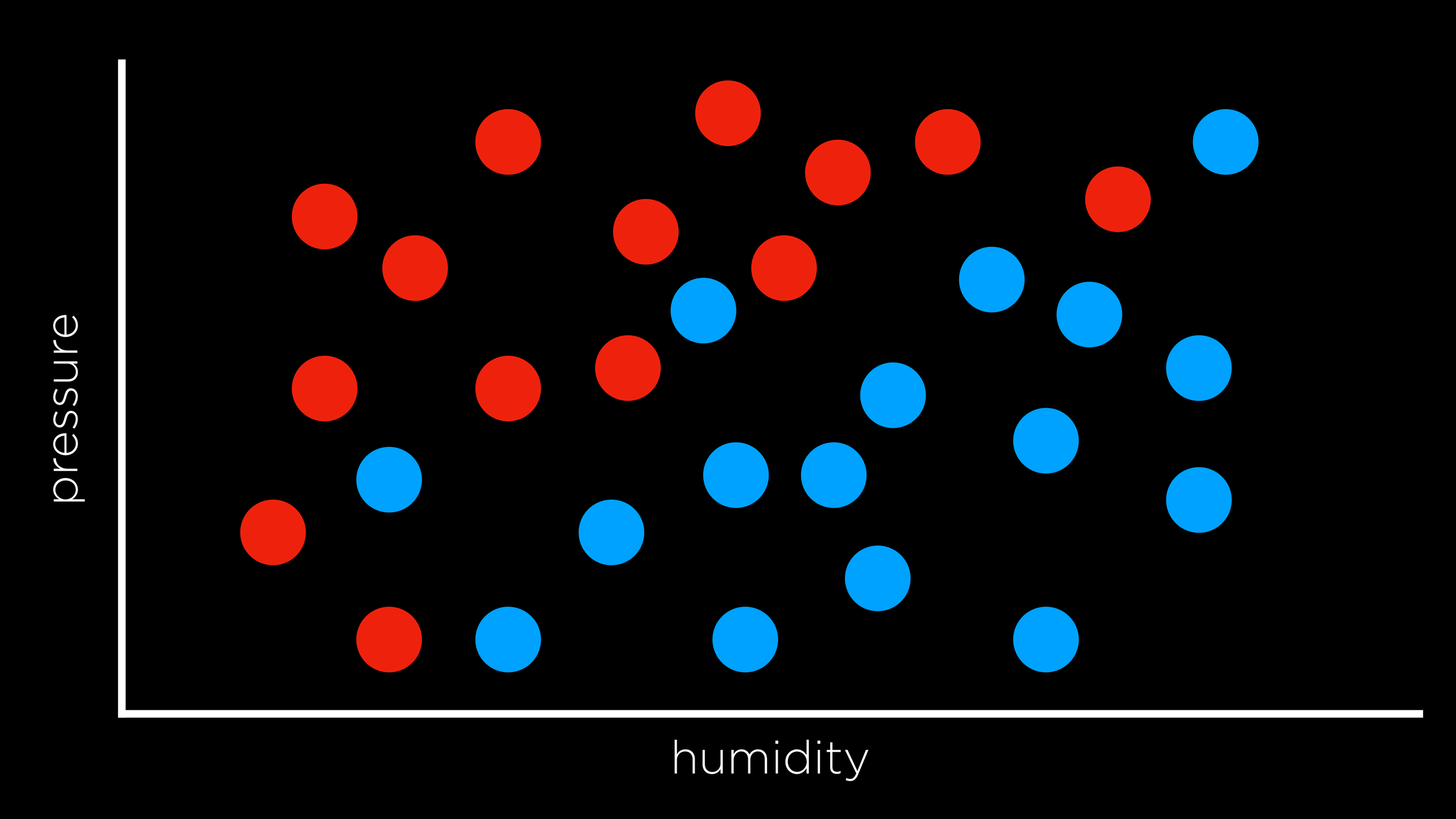
$f(93, 999.7) = \text{Rain}$

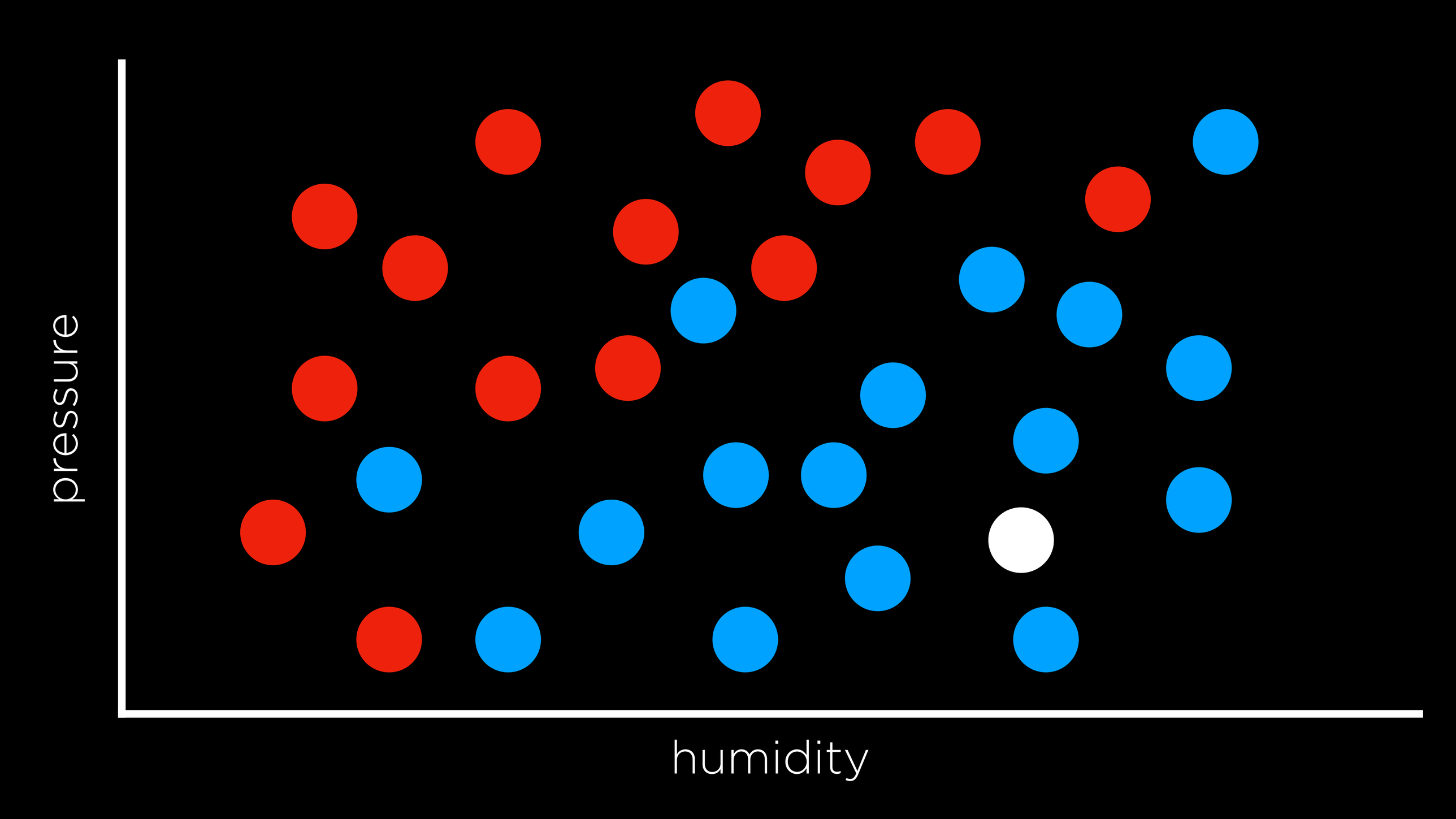
$f(49, 1015.5) = \text{No Rain}$

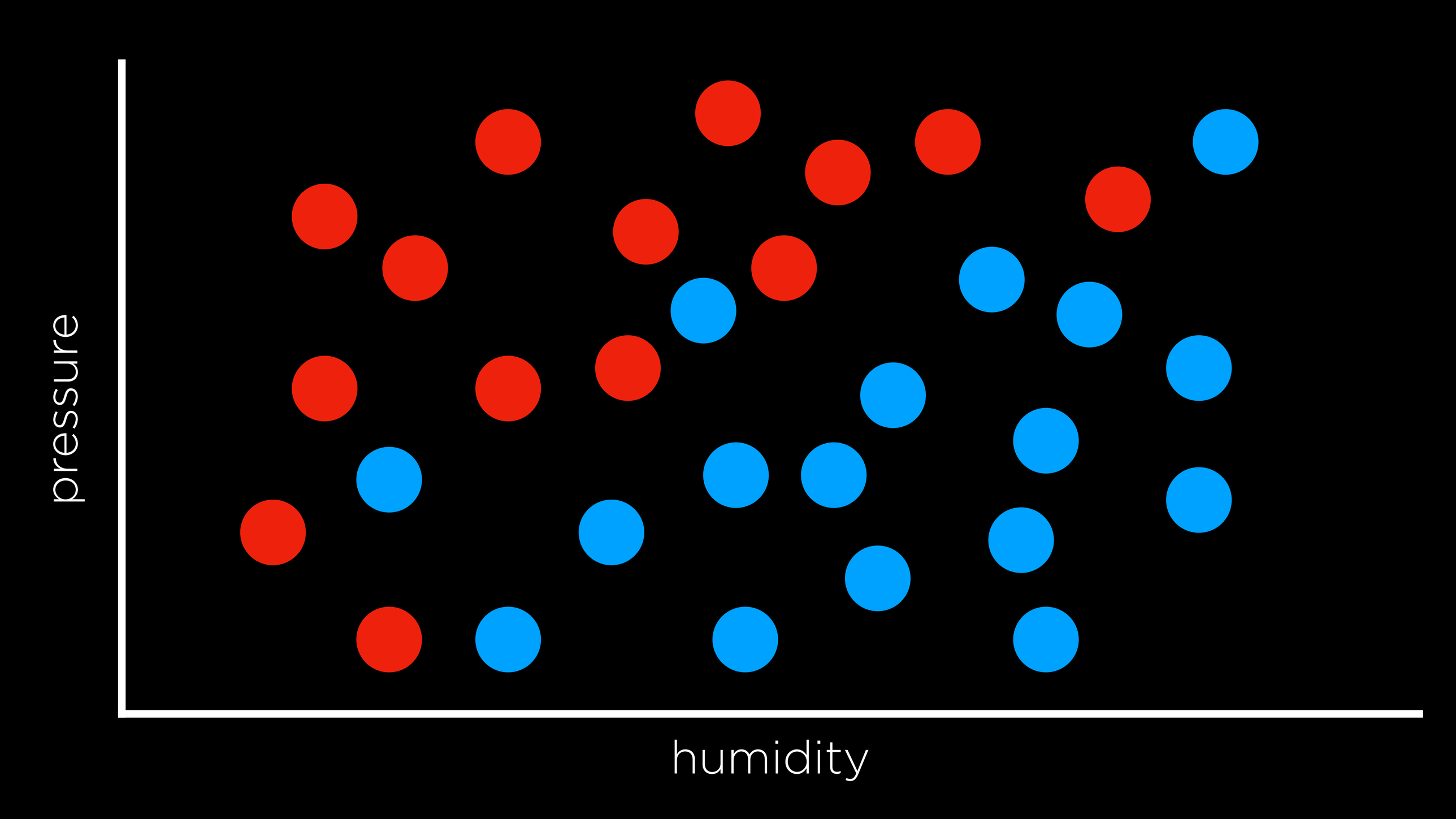
$f(79, 1031.1) = \text{No Rain}$


pressure
humidity


pressure
humidity

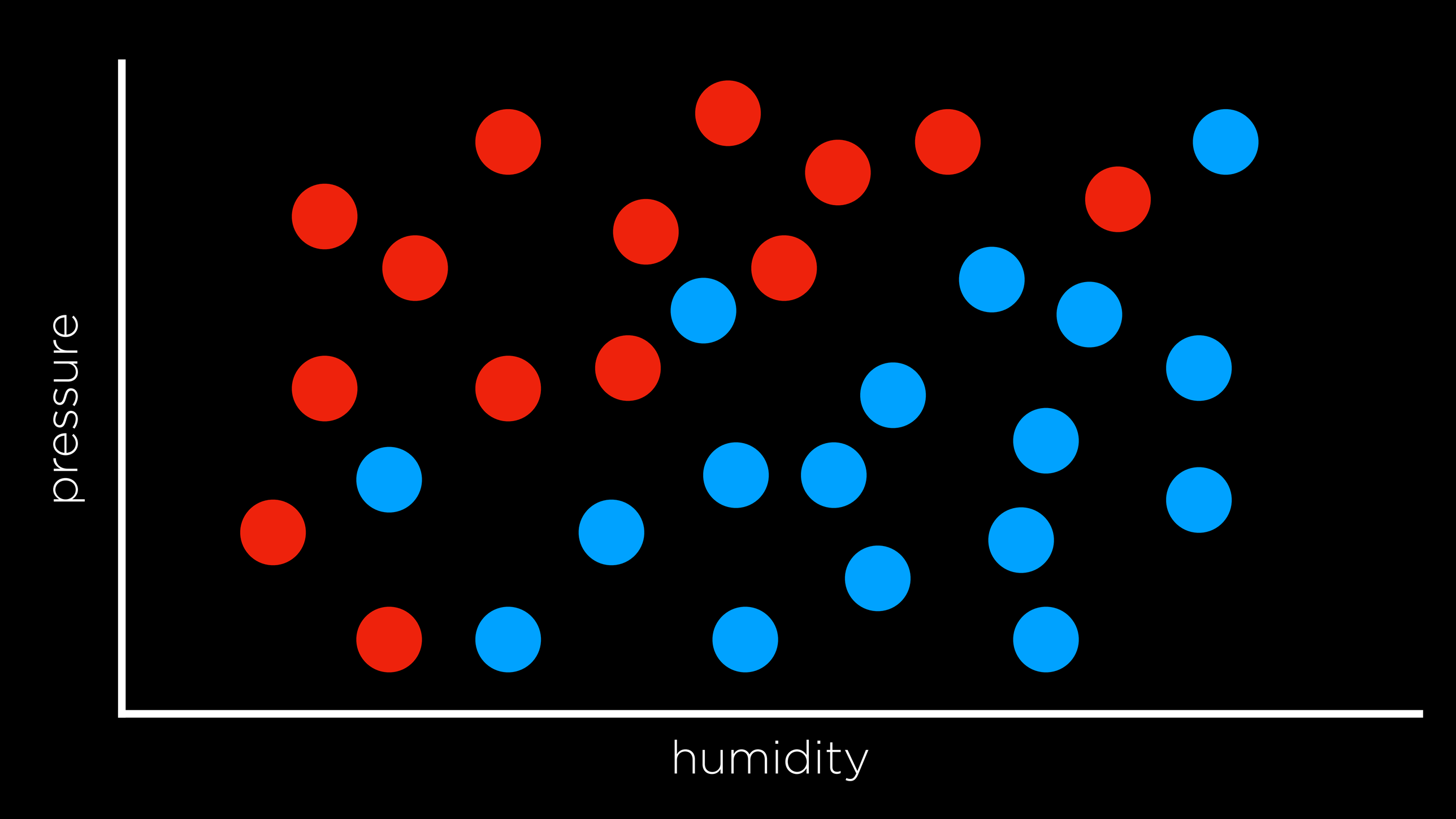

pressure
humidity

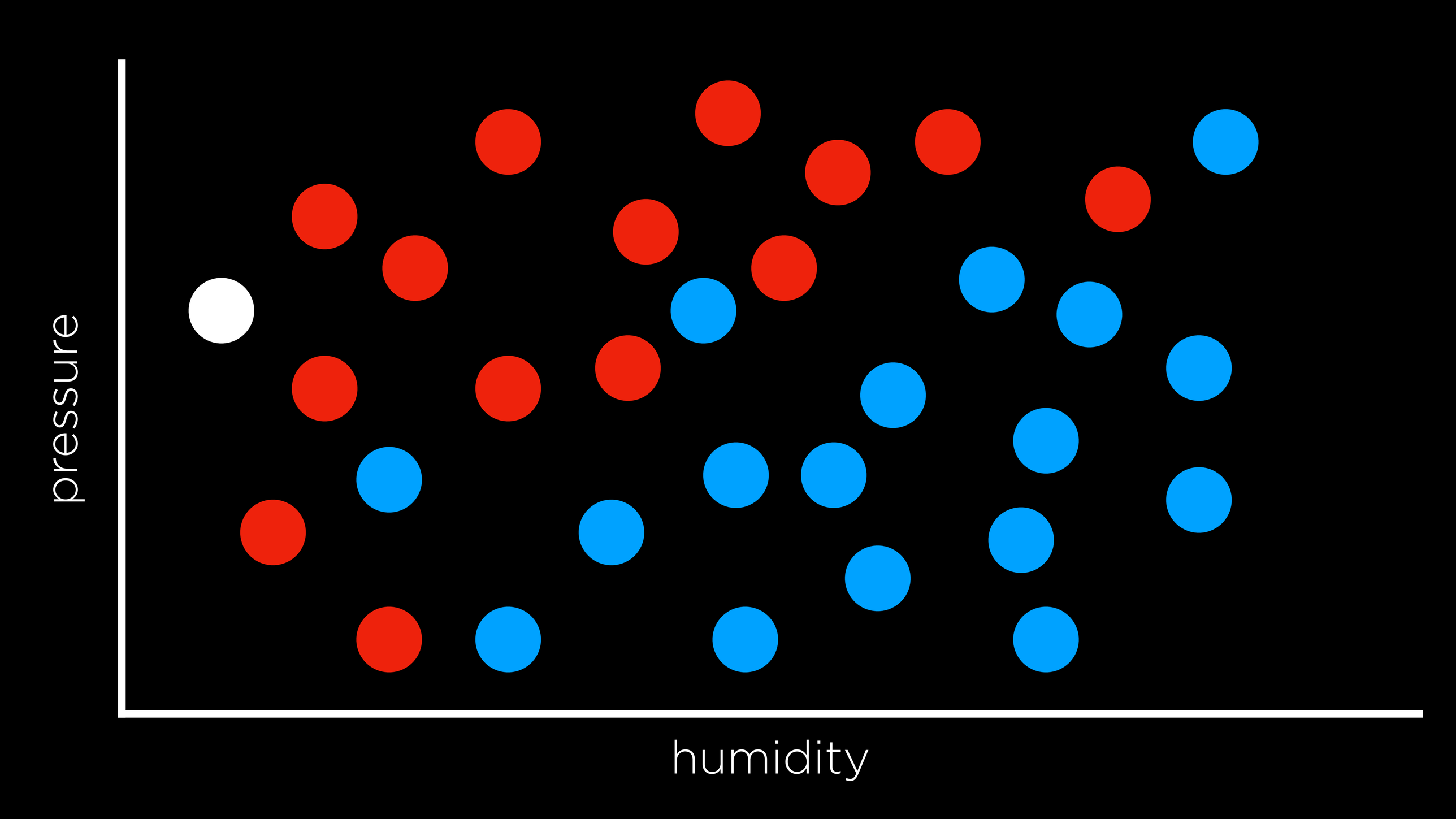

pressure
humidity

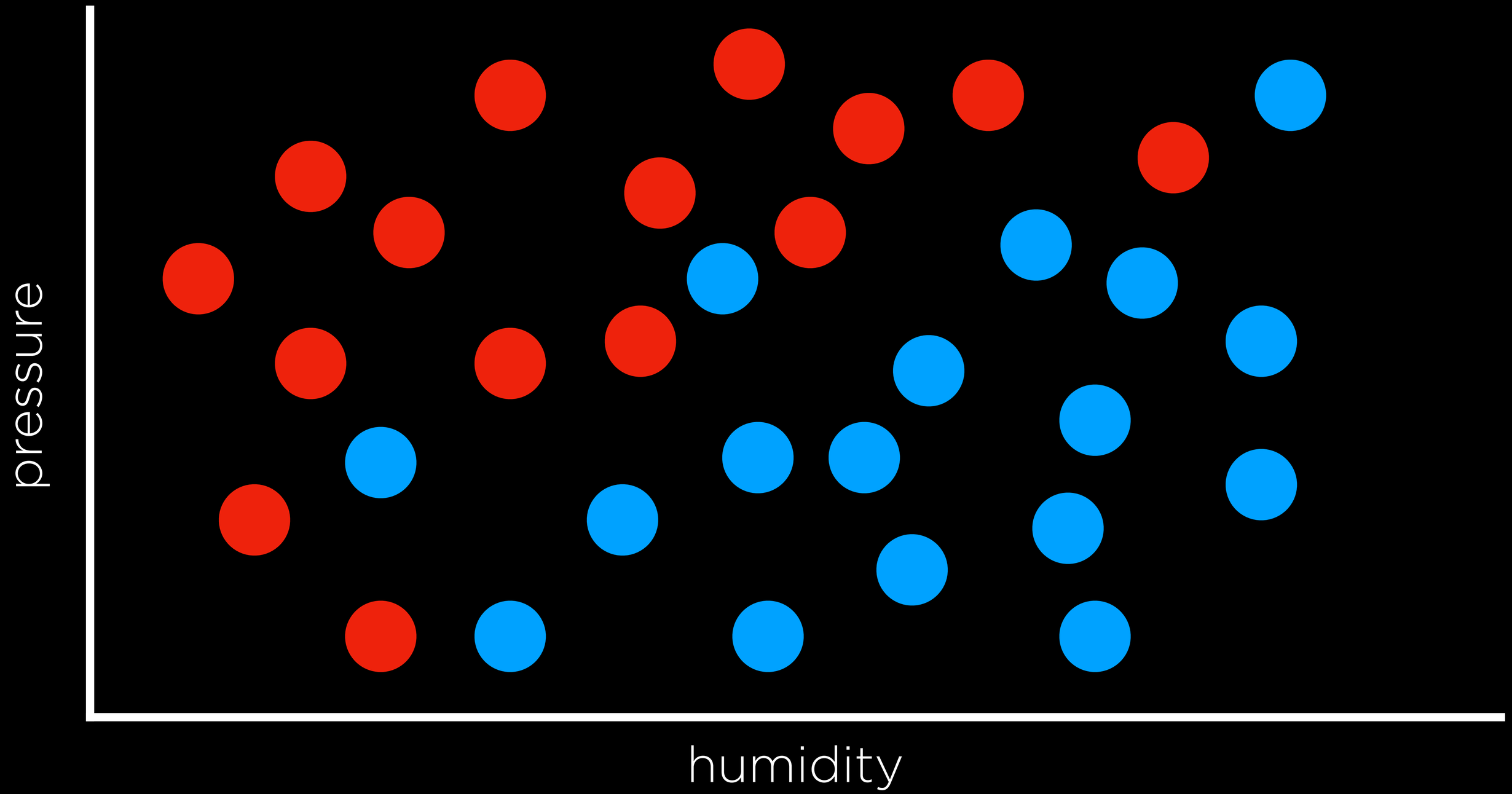

pressure
humidity

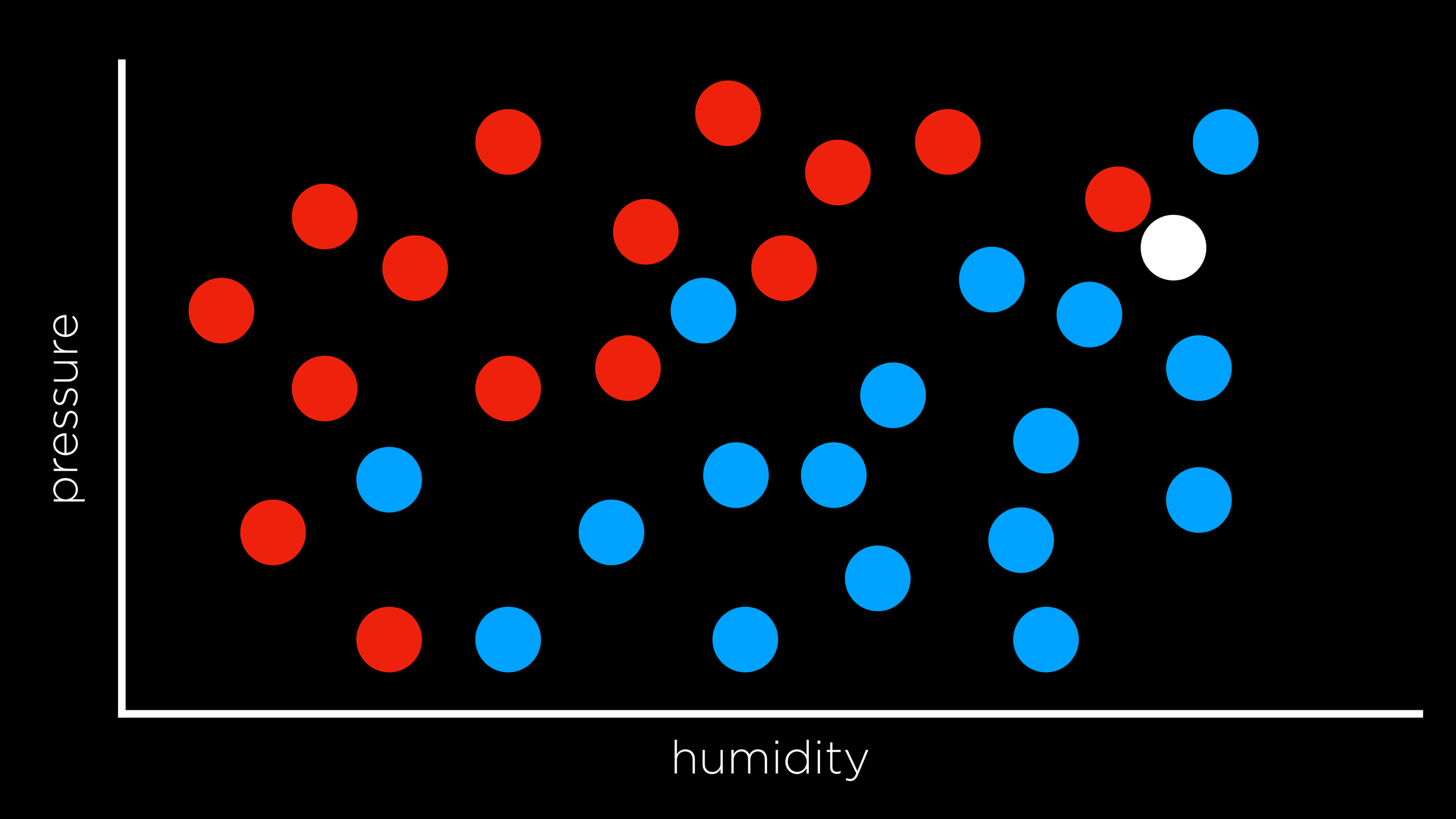
nearest-neighbor classification

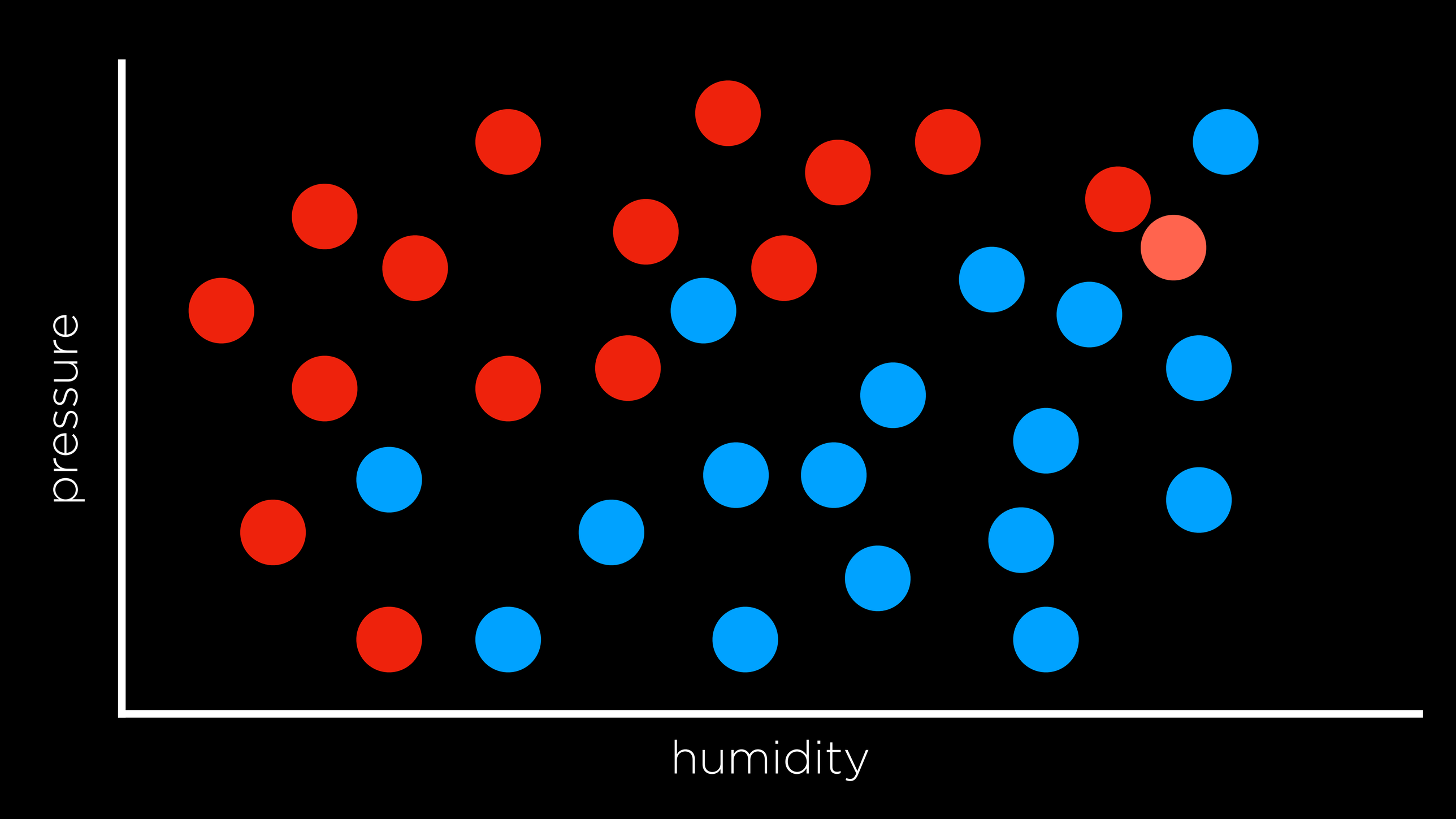
algorithm that, given an input, chooses the class of the nearest data point to that input

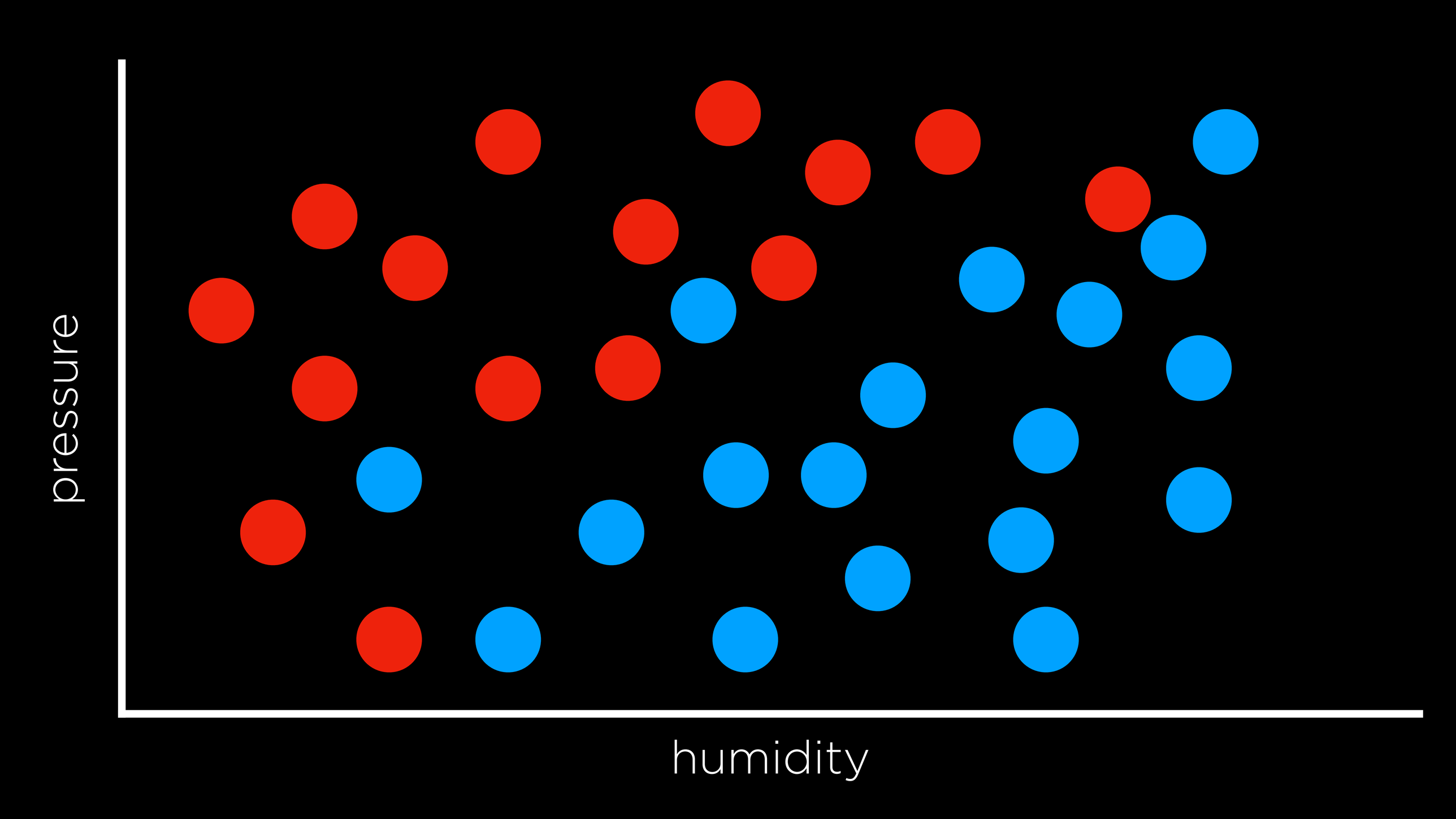

pressure
humidity


pressure
humidity


pressure
humidity

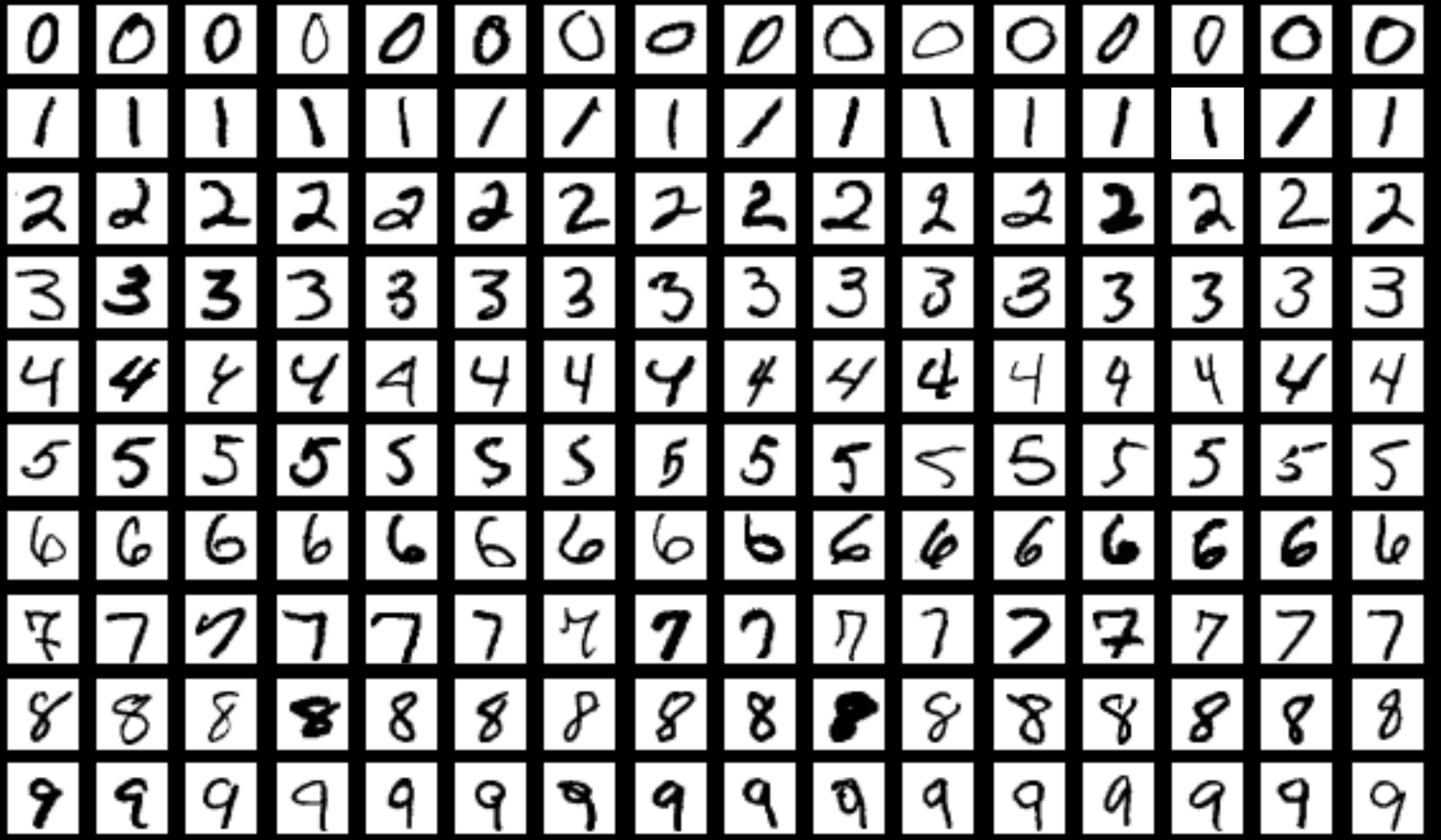

pressure
humidity


pressure
humidity

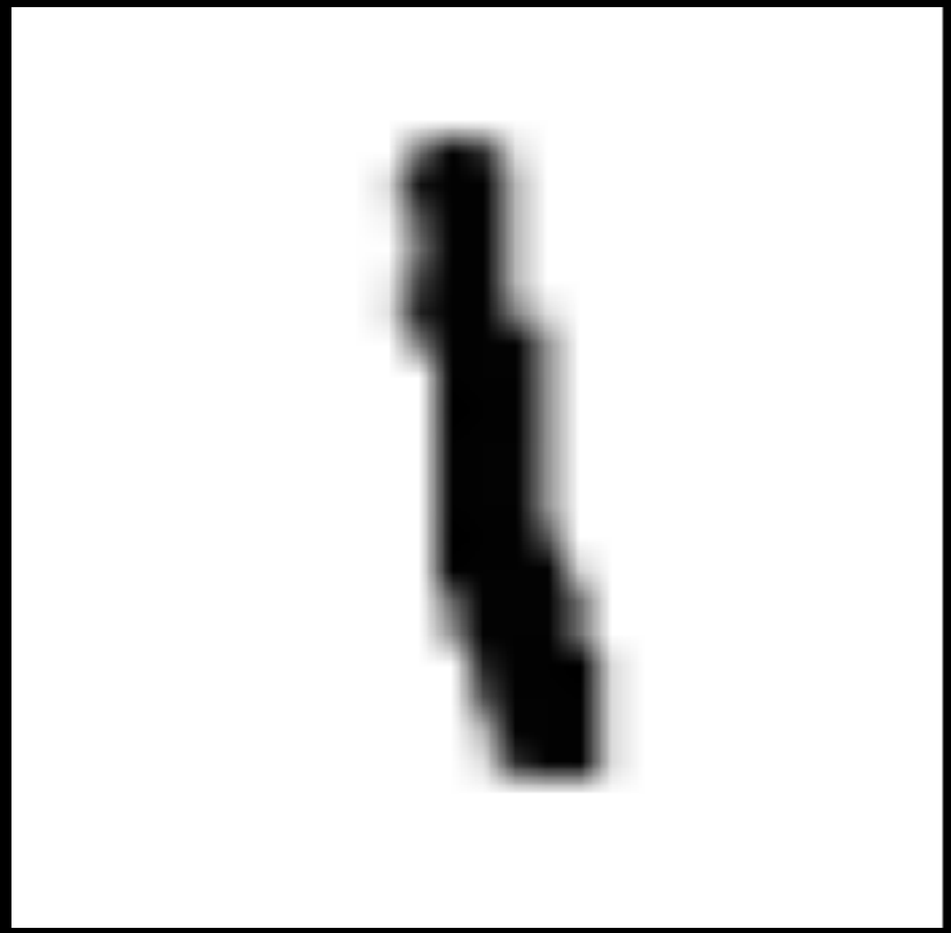

pressure
humidity

k-nearest-neighbor classification

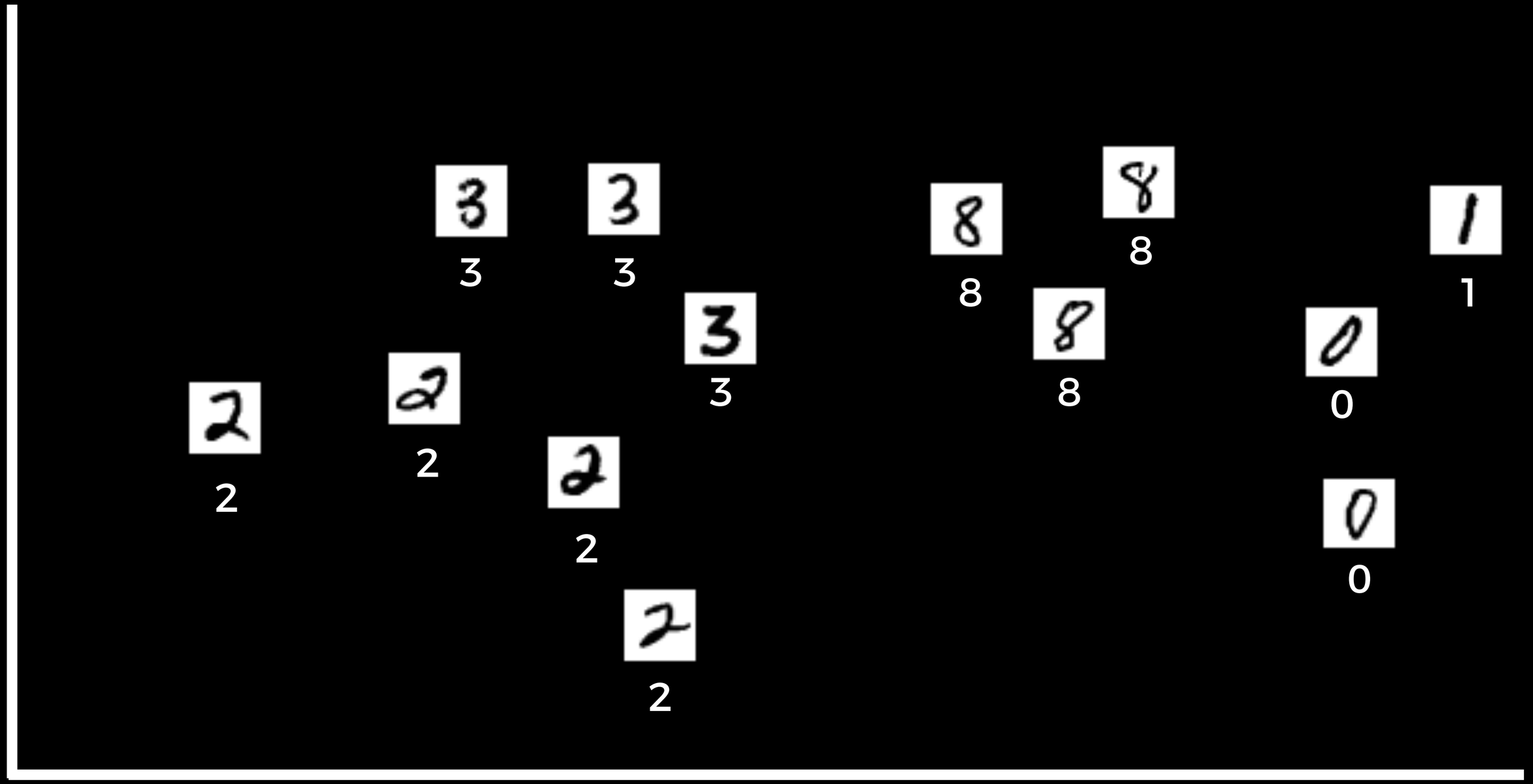
algorithm that, given an input, chooses the most common class out of the k nearest data points to that input

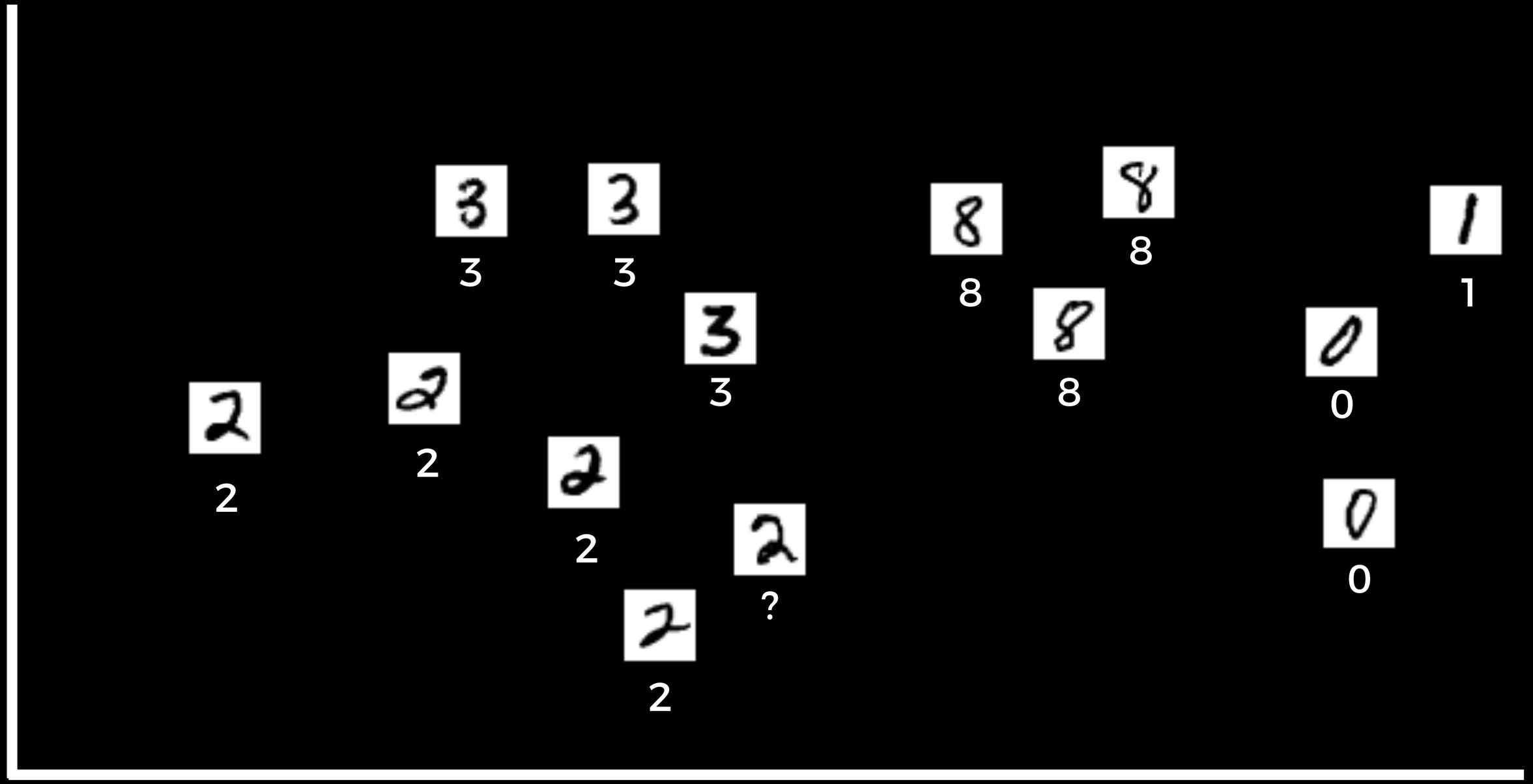


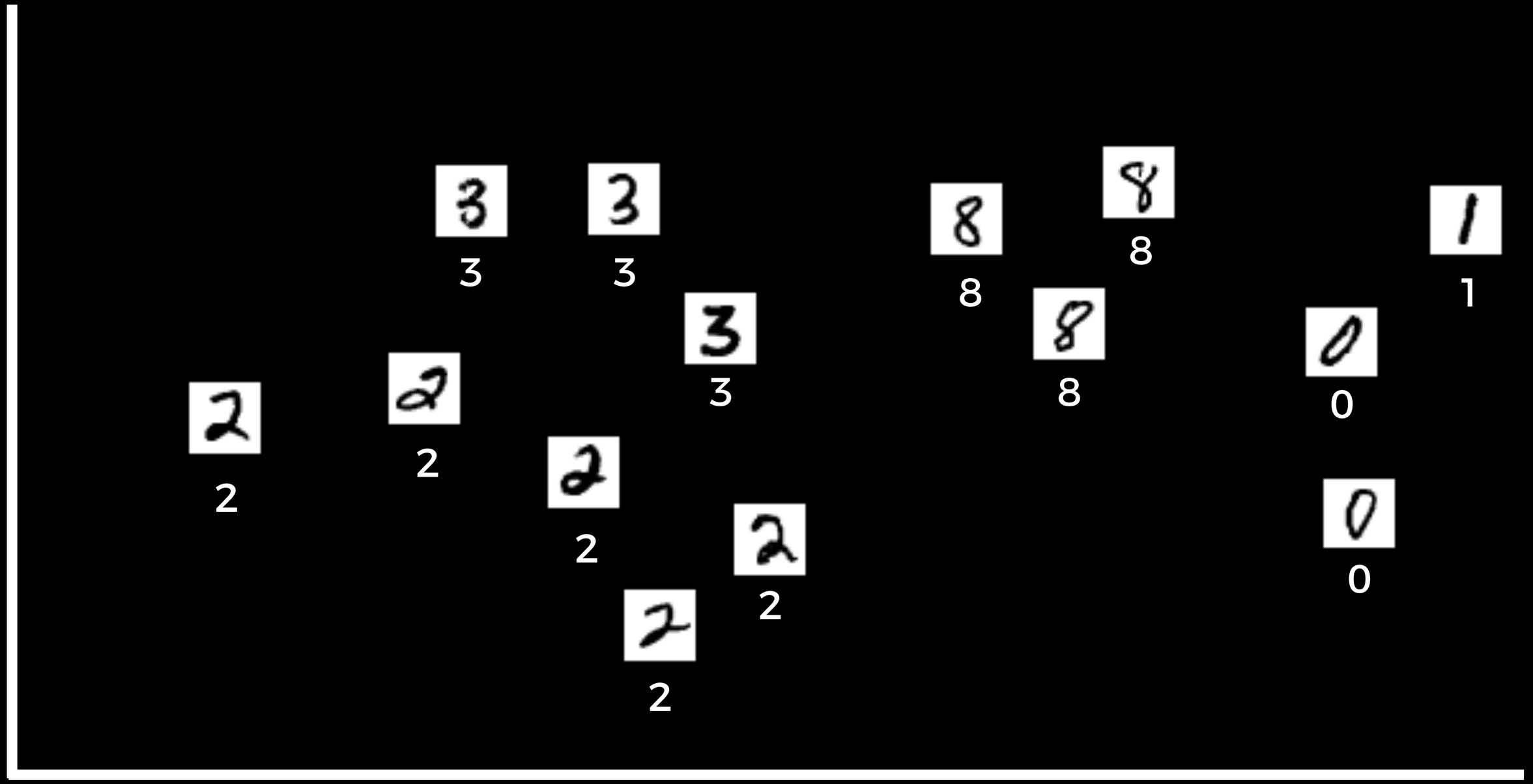
0	0	0	0	0	0	0	0	0
0	0	0	0.6	0.8	0	0	0	0
0	0	0	0.8	0.9	0.9	0	0	0
0	0	0	0.9	0.9	0.9	0	0	0
0	0	0	0.9	0.9	0.9	0	0	0
0	0	0	0.9	0.9	0.9	0	0	0
0	0	0	0.9	0.9	0.9	0	0	0
0	0	0	0.9	0.9	0.9	0	0	0
0	0	0	0.9	0.9	0.9	0	0	0
0	0	0	0	0.8	0.6	0	0	0
0	0	0	0	0	0	0	0	0

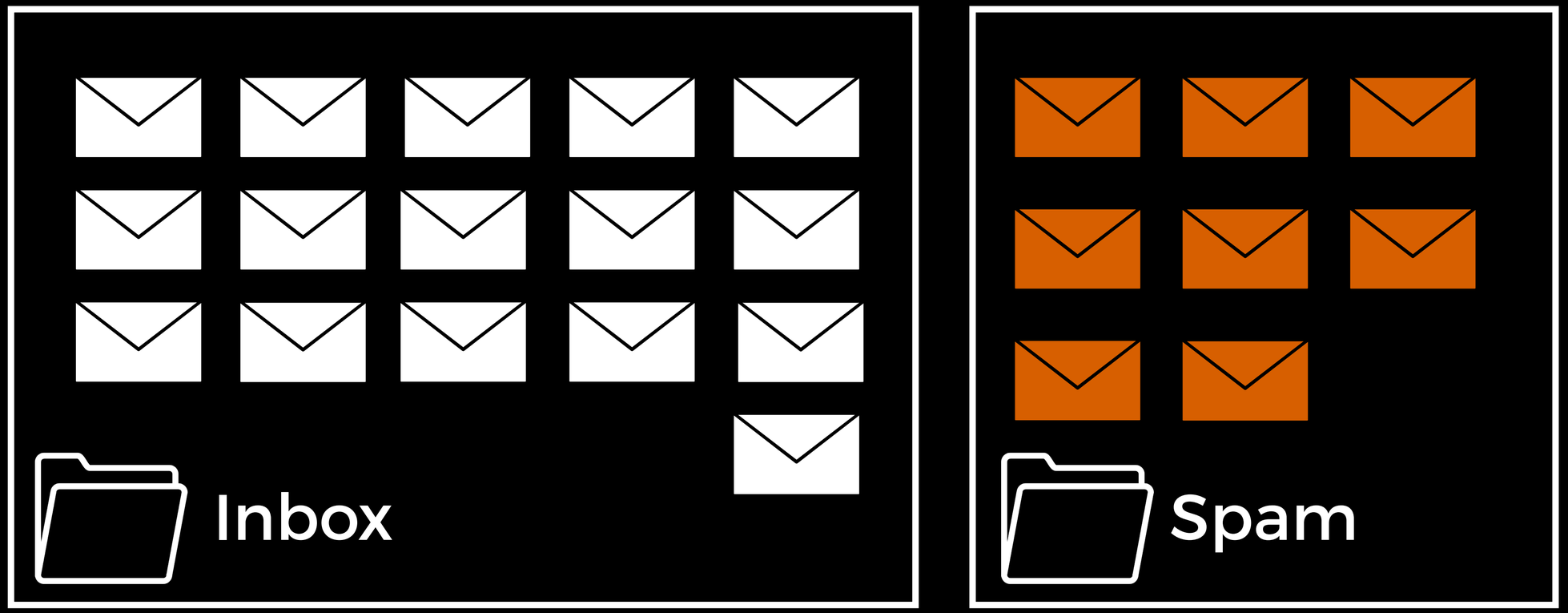


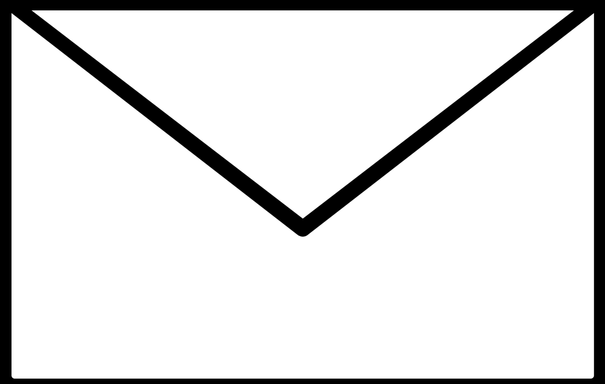
0	0	0	0	0	0	0	0	0
0	0	0	0.6	0.8	0	0	0	0
0	0	0	0.8	0.9	0.9	0	0	0
0	0	0	0.9	0.9	0.9	0	0	0
0	0	0	0.9	0.9	0.9	0	0	0
0	0	0	0.9	0.9	0.9	0	0	0
0	0	0	0.9	0.9	0.9	0	0	0
0	0	0	0.9	0.9	0.9	0	0	0
0	0	0	0.9	0.9	0.9	0	0	0
0	0	0	0	0.8	0.6	0	0	0
0	0	0	0	0	0	0	0	0


3
3
3
3
8
8
8
8
1
1
2
2
2
2
2
2
0
0
0
0

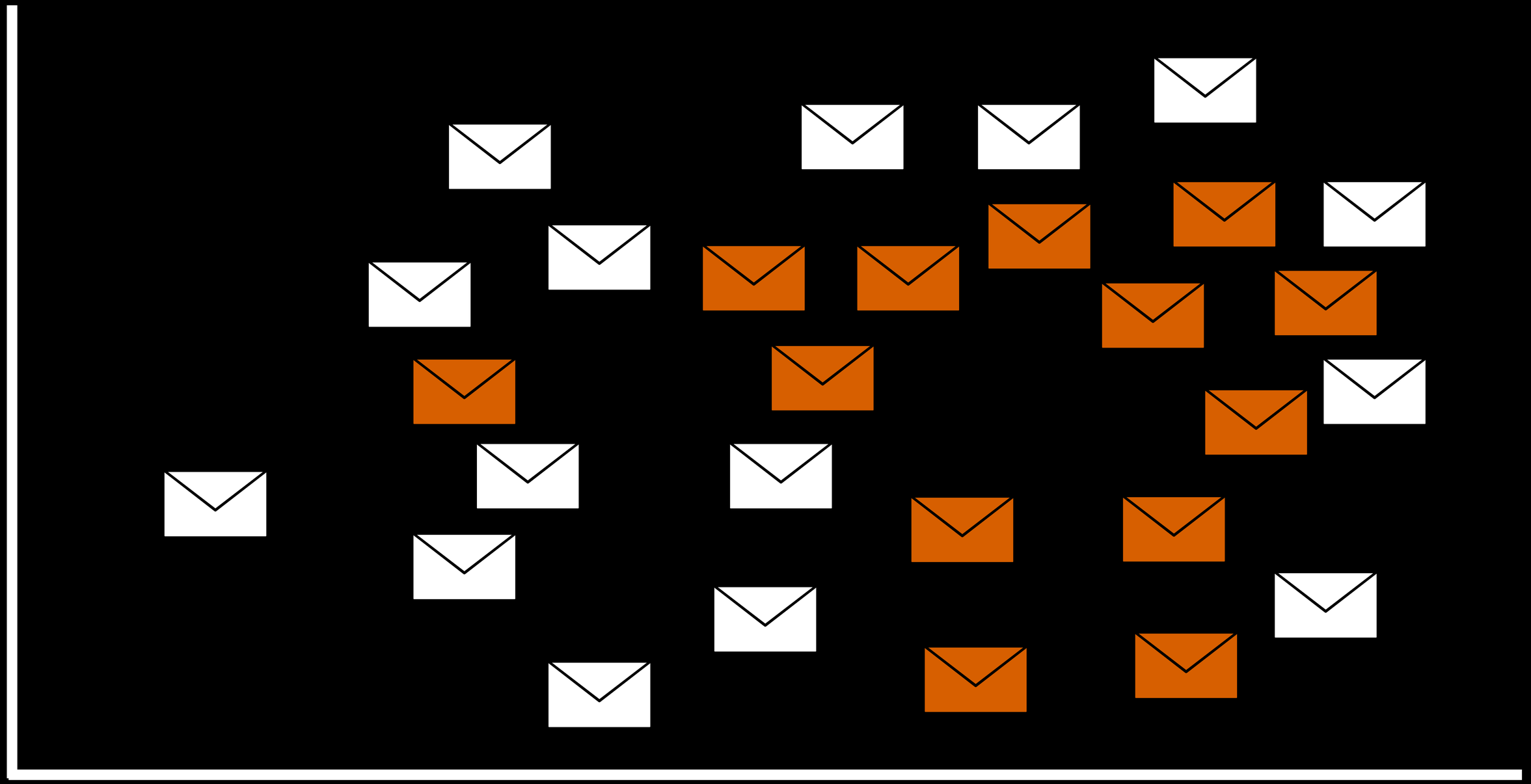

3
3
3
3
8
8
8
8
1
1
2
2
2
2
2
2
2
?
0
0
0
0

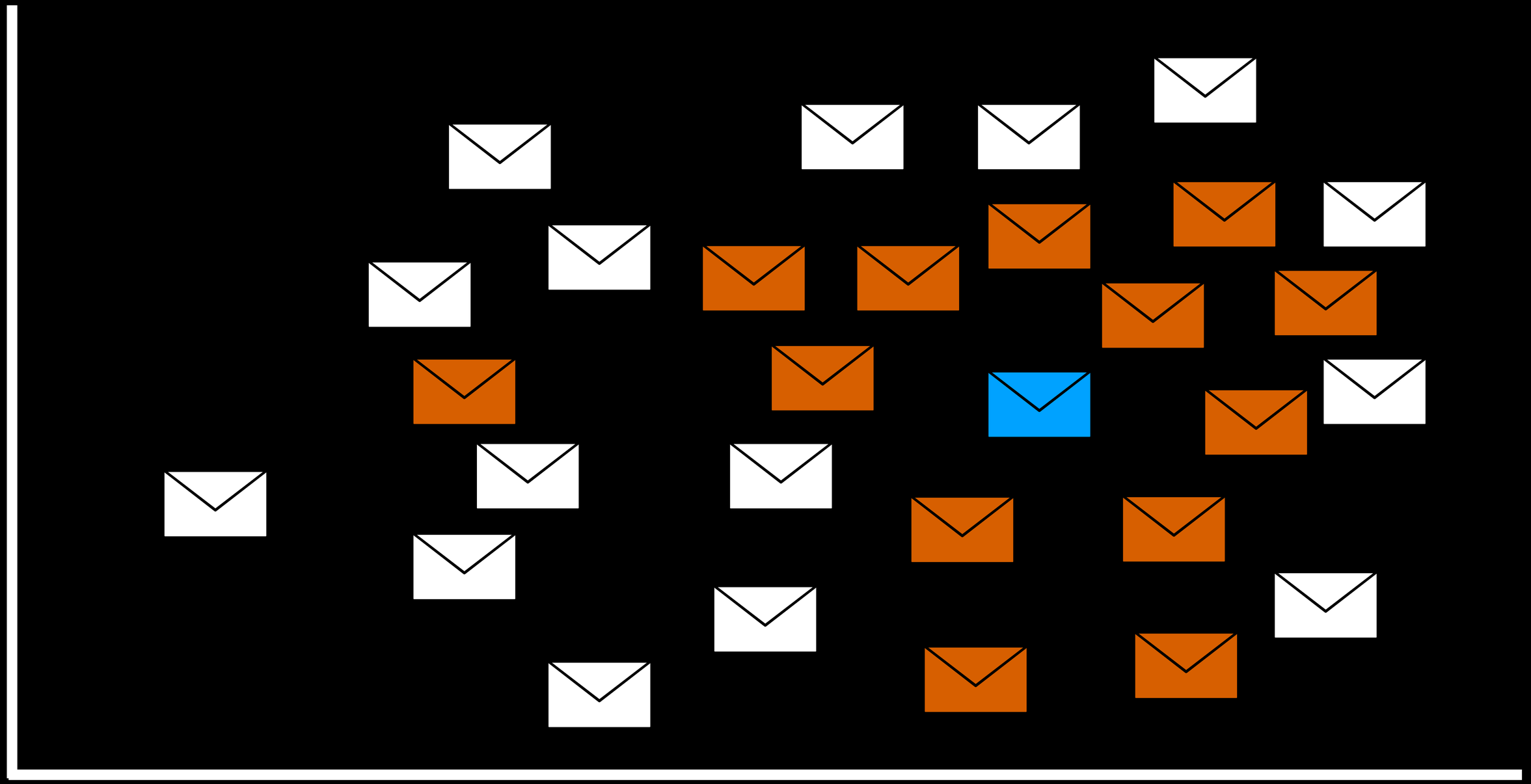

3
3
3
3
8
8
8
8
1
1
2
2
2
2
2
2
2
2
0
0
0
0

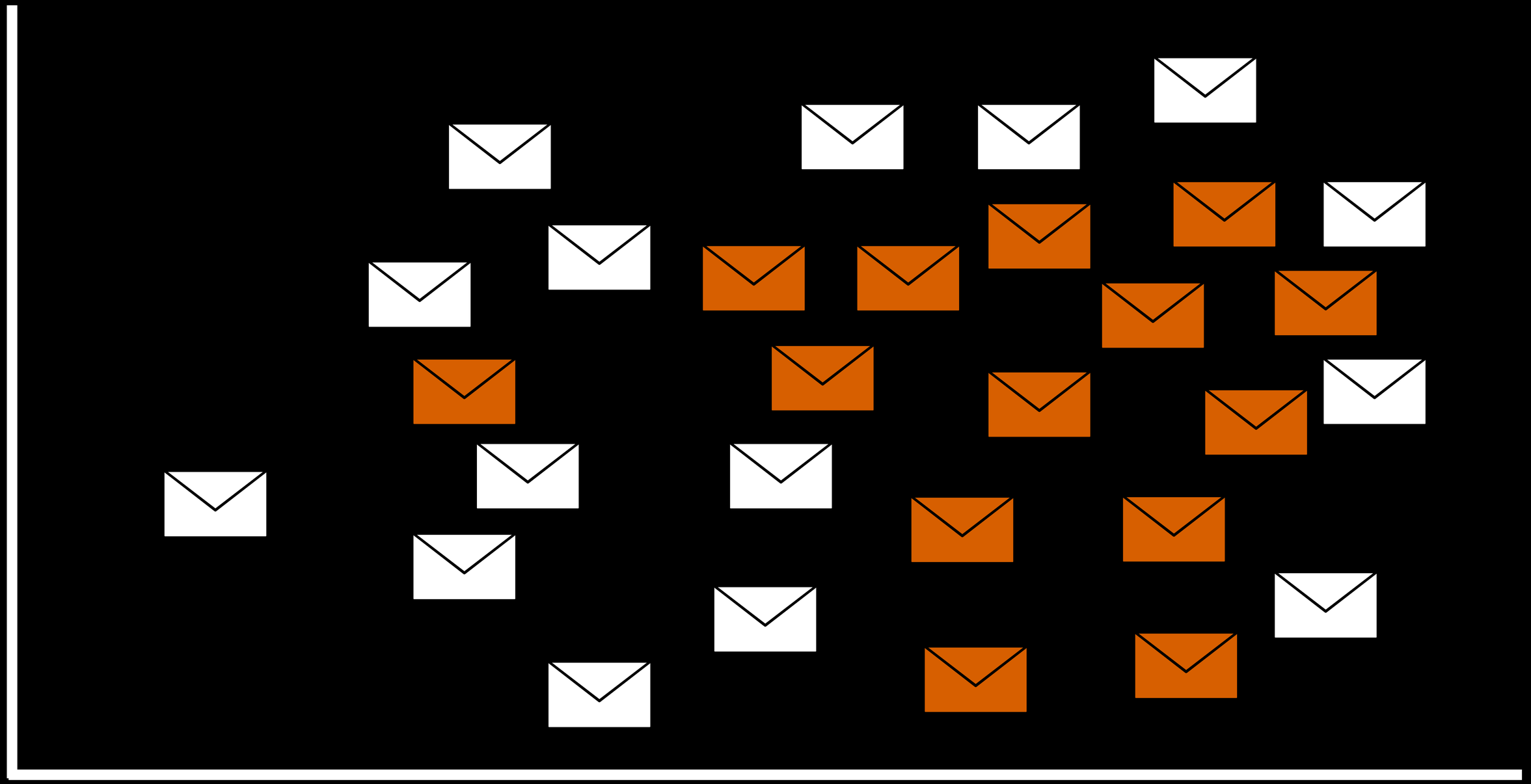

Inbox
Spam

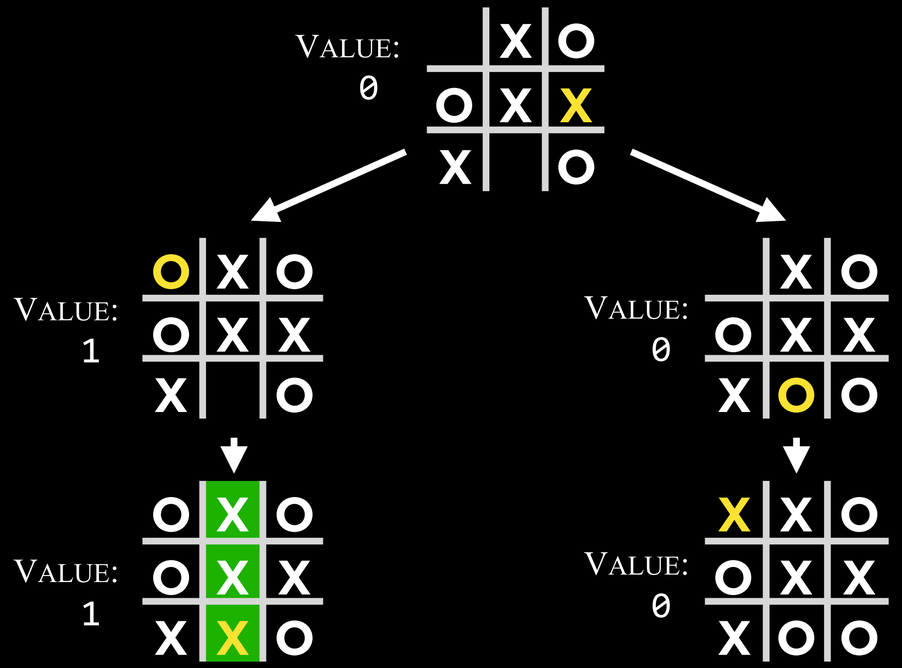


[1, 2, 5, 2, 3, 1, 2, 8, 1, 3]

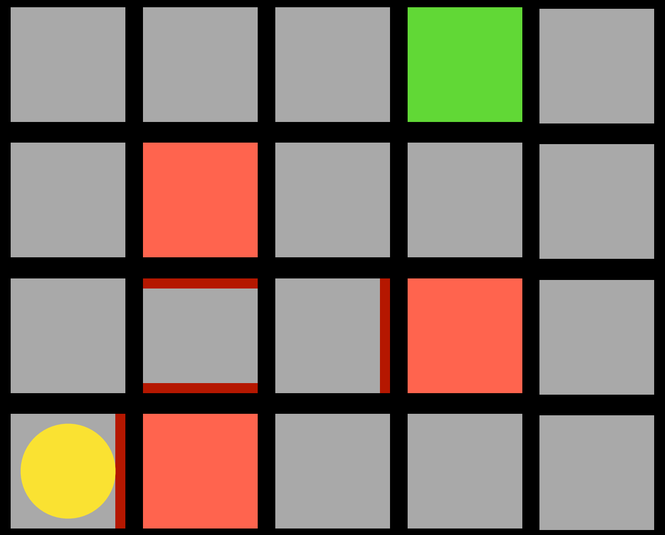


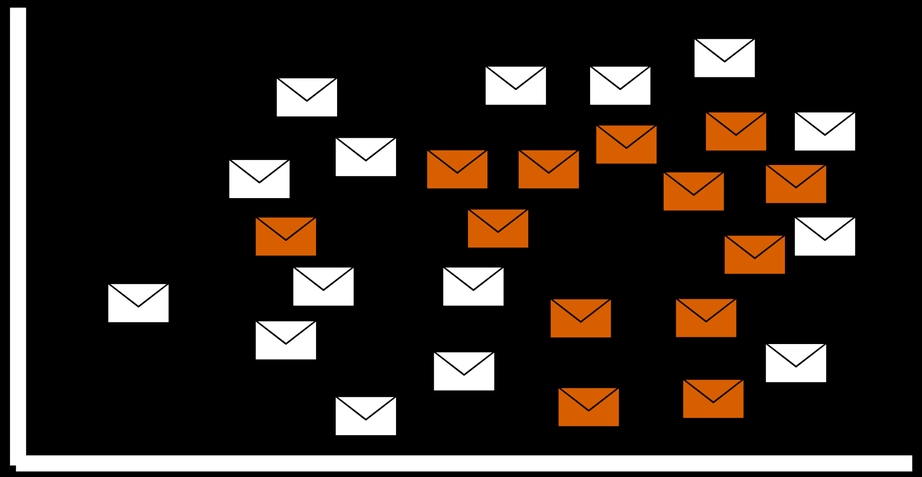






	11+10	9	8	7	6	5	4	3	2	1	B
	10+11										1
	9+12		7+10	8+9	9+8	10+7	11+6	12+5	13+4		2
	8+13		6+11						14+5		3
	7+14	6+13	5+12		10	9	8	7	15+6		4
			4+13		11						5
A	1+16	2+15	3+14		12	11	10	9	8	7	6





Artificial Intelligence