UiA University of Agder Norway

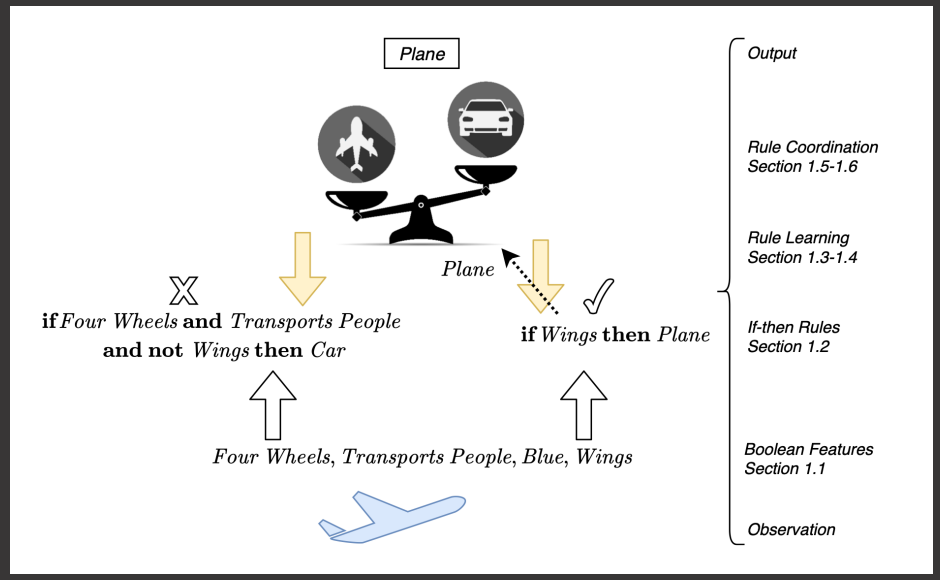
CAIR CENTRE FOR ARTIFICIAL INTELLIGENCE RESEARCH

The Tsetlin machine – inference and learning

Ole-Christoffer Granmo
Centre for AI Research
University of Agder
Norway

Overview

- Machine learning basics
- Tsetlin machine architecture
- Booleanization
- Defining patterns with if-then rules
- Classification
- Learning
  - Recognize feedback
  - Erase feedback
  - Reject feedback (reasoning by elimination)
  - Pattern prioritization



Machine Learning Basics

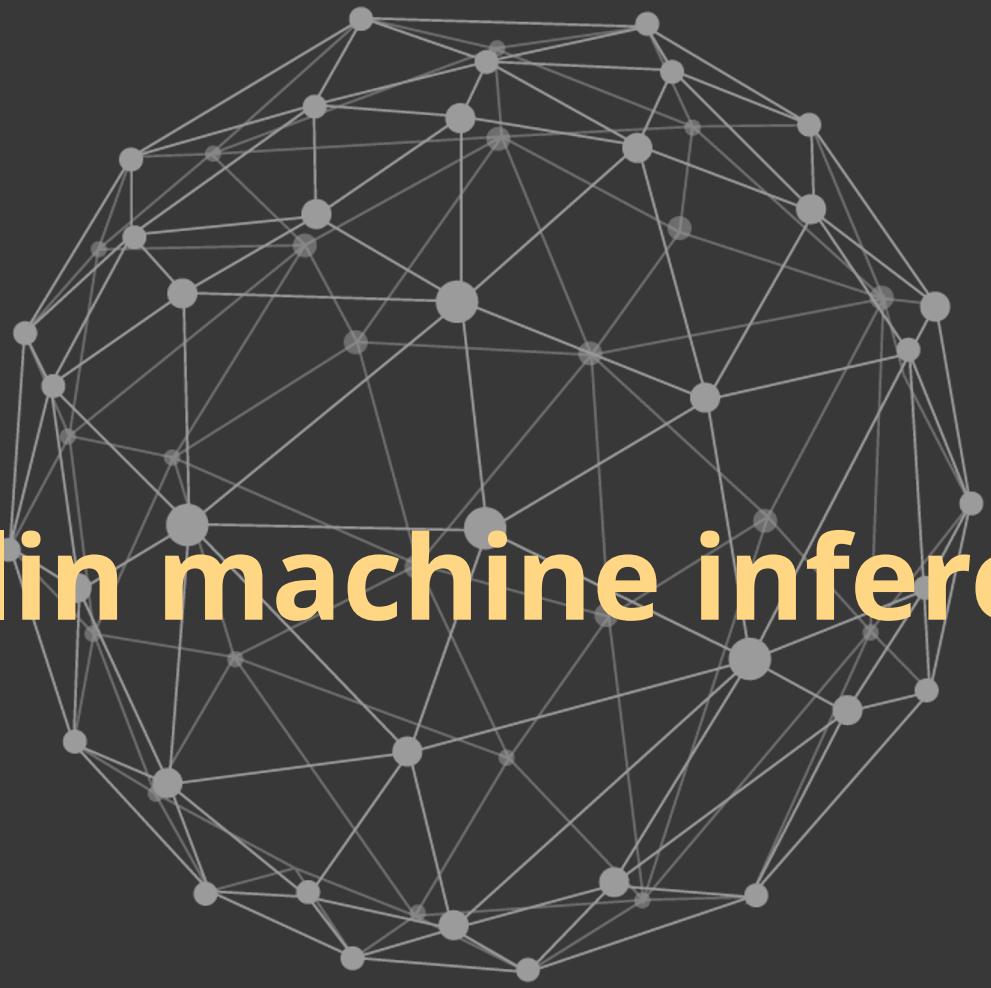
Machine learning deals with how computers can improve from experience and what constitutes the fundamental laws of learning. A typical machine learning system consists of:

- *A **task** such as classification or regression, for instance, photo classification or breast cancer recurrence prediction.*
- ***Training data** with individual cases, which is the experience.*
- *An **algorithm** that learns **supervised** from labelled training data, **unsupervised** from unlabelled data, or from **reinforcement** such as penalties and rewards.*
- ***Testing data** for evaluating the performance of the algorithm after training.*

Tsetlin Machines vs Black Boxes

Recent research has brought increasingly accurate learning algorithms and powerful computation platforms. However, the accuracy gains come with escalating computation costs, and models are getting too complicated for humans to comprehend. Mounting computation costs make AI an asset for the few and impact the environment. Simultaneously, the obscurity of AI-driven decision-making raises ethical concerns regarding unfair, erroneous, and, in high-risk domains, fatal decisions. Tsetlin machines address the following key challenges:

- *They are universal function approximators, like neural networks.*
- *They are rule-based, like decision trees.*
- *They are summation-based, like Naive Bayes classifier and logistic regression.*
- *They are hardware-near, with low energy- and memory footprint.*



Tsetlin machine inference

CAIR
CENTRE FOR ARTIFICIAL INTELLIGENCE RESEARCH

Tsetlin machines reason logically

Logical rules

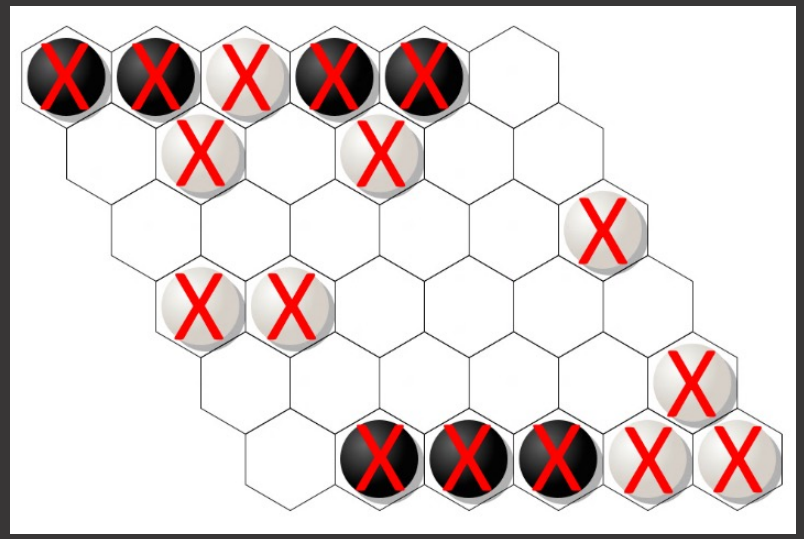
if «rash» **and** «reaction» **and** «penicillin» **then** Allergy

Antecedent

Consequent

Logical rules

if



Antecedent

then Black Wins

Consequent

Tsetlin machine inference – step by step

Van

Four Wheels

Transport People

Blue

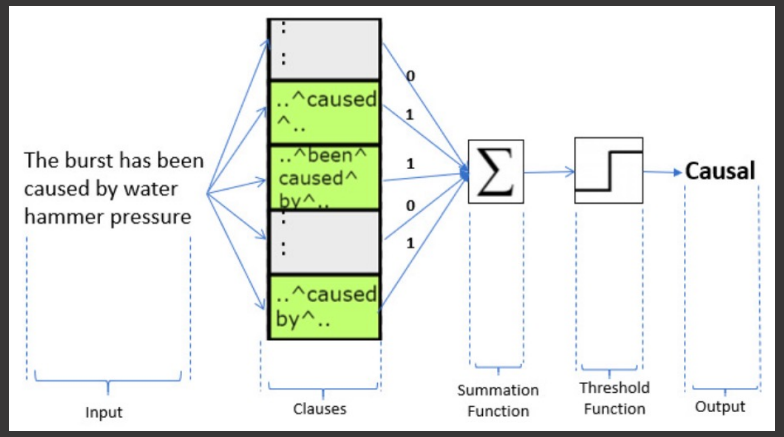
- A Tsetlin machine observes objects, which it recognizes based on the objects' features
- Each feature is an object property that is either True or False, making it *Boolean*

Advantages of Boolean features

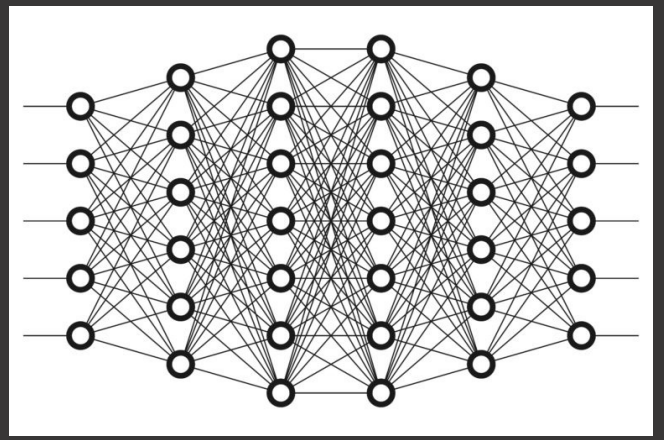
if *Four Wheels* **and** *Transports People* **and not** *Wings* **then** *Car*.

- You can use Boolean algebra - the language of computers - to build patterns
- If you know a programming language, you have probably used Boolean algebra
- You write Boolean expressions to control the flow of a program, for example, with *if-then* constructs

Boolean algebra and interpretability

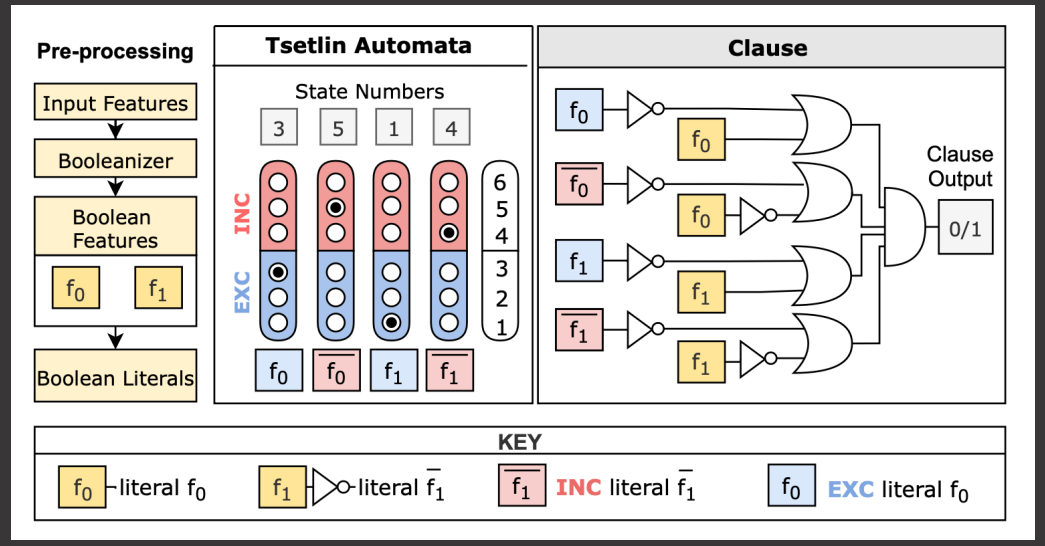


vs



- Boolean algebra is the same as propositional logic, which models logical reasoning with *True* and *False* statements
- Such logical reasoning is easy for humans to understand, making the Tsetlin machine *transparent*
- In comparison, so-called *black box* machine learning models are too complicated for human comprehension

Boolean features and bitwise operators



Abu Bakar, Tousif Rahman, Rishad Shafik, Fahim Kawsar, and Alessandro Montanari. Adaptive Intelligence for Batteryless Sensors Using Software- Accelerated Tsetlin Machines. In Proceedings of SenSys 2022. ACM, 2022.

- Boolean features fit well with a computer because you can store them individually as bits
- Hardware-near bitwise operators then dramatically increase inference speed
- Further, the resulting small energy- and memory footprint is ideal for edge computing and Internet of Things (IoT)

Predicting breast cancer recurrence

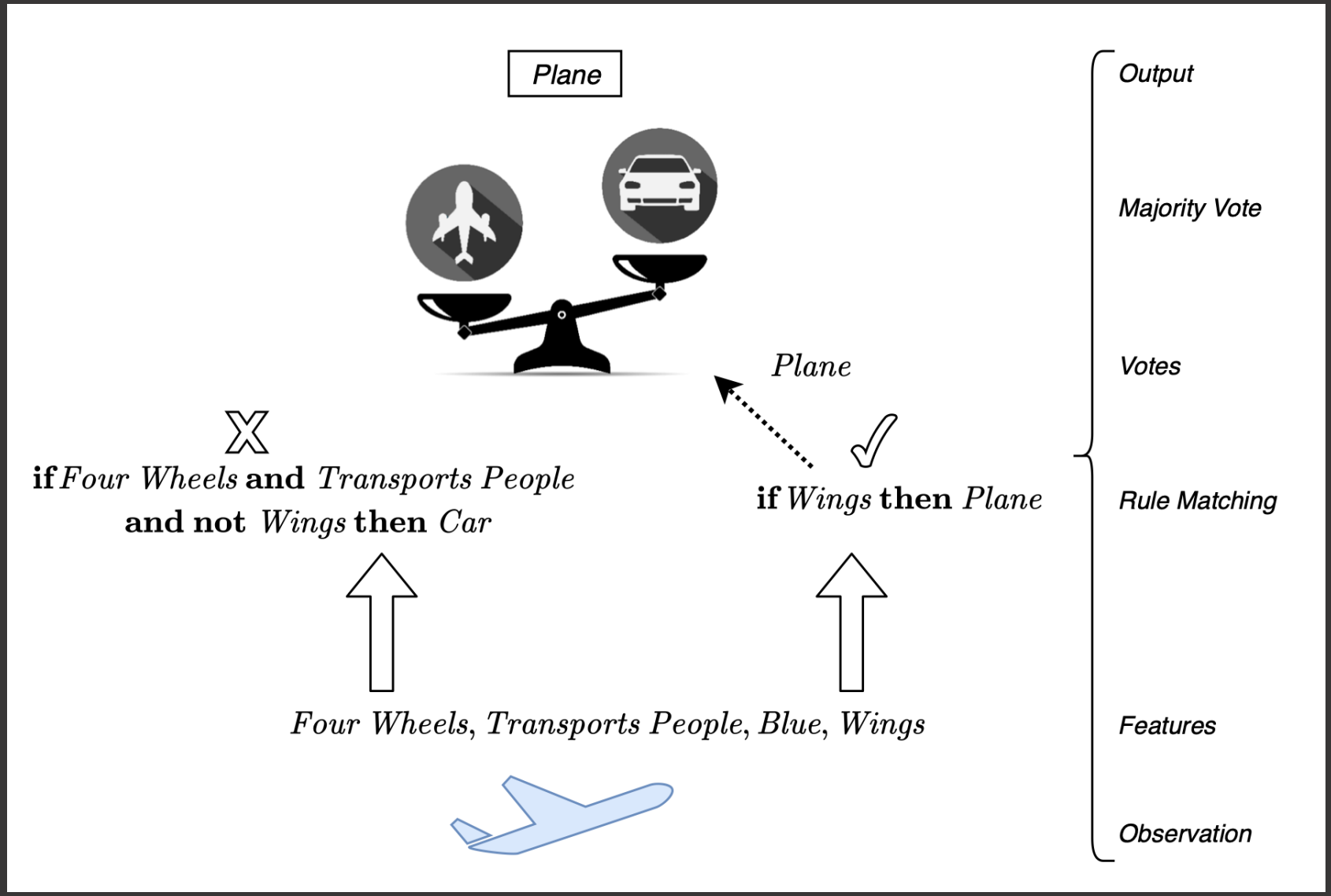
Property	Values	Description
Menopause	lt40, ge40, premeno	Pre- or postmenopausal status at time of diagnosis
Inv-nodes	0-2, 3-5, 6-8	Axillary lymph nodes containing visible metastatic breast cancer
Deg-malig	1, 2, 3	Degree of tumor malignancy
Recurrence	yes, no	Patient recurrence status

#	Menopause	Inv-nodes	Deg-malig	*Recurrence*
1.	ge40	3-5	3	*yes*
2.	lt40	0-2	3	*no*
3.	ge40	6-8	3	*yes*
4.	ge40	0-2	2	*no*
5.	premeno	0-2	3	*yes*
6.	premeno	0-2	1	*no*

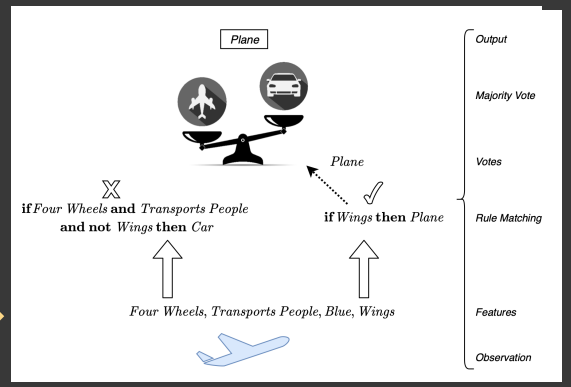
Booleanization

	Menopause			Inv-nodes			Deg-malig			
#	lt40	ge40	premeno	0-2	3-5	6-8	1	2	3	*Recurrence*
1.	·	●	·	·	●	·	·	·	●	●
2.	●	·	·	●	·	·	·	·	●	·
3.	·	●	·	·	·	●	·	·	●	●
4.	·	●	·	●	·	·	·	●	·	·
5.	·	·	●	●	·	·	·	·	●	●
6.	·	·	●	●	·	·	●	·	·	·

Tsetlin machine inference



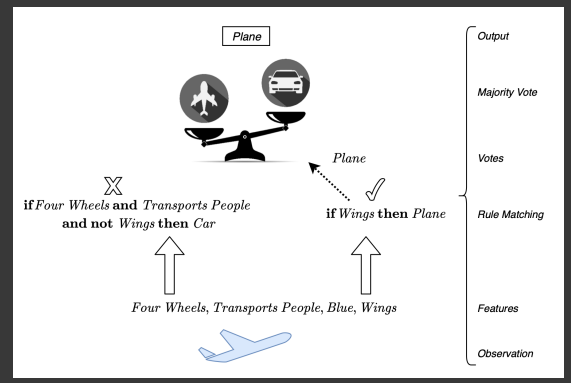
Example input



#	Four Wheels	Transports People	Wings	Yellow	Blue	*Car*
1.	●	●	·	·	●	●
2.	●	●	·	●	·	●
3.	●	●	·	●	·	●
4.	●	●	●	·	●	·
5.	●	·	●	●	·	·
6.	·	●	●	·	●	·

CAIR CENTRE FOR ARTIFICIAL INTELLIGENCE RESEARCH

Describing patterns with if-then rules



if *condition* **then** *class*.

Composing patterns with AND

Four Wheels **and** *Transports People*

Example if-then rule

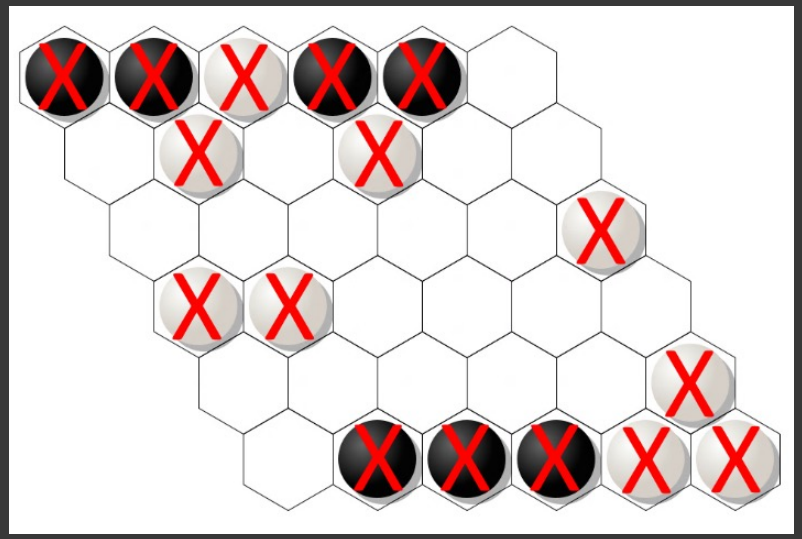
if *Four Wheels* **and** *Transports People* **then** *Car*

Refining patterns with negation

if *Four Wheels* **and** *Transports People* **and not** *Wings* **then** *Car*.

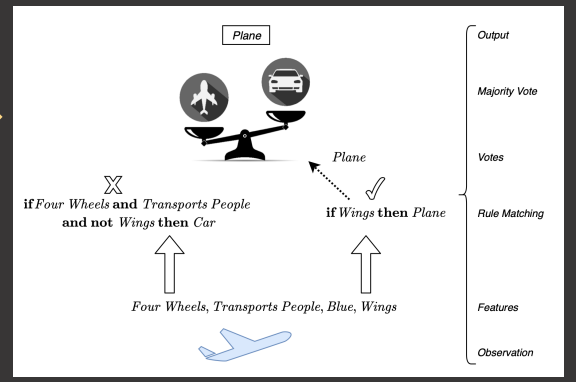
Example patterns

0	0	*	1	*	0	0	0	
0	0	*	1	*	0	0	0	
0	*	*	1	*	*	*	0	
0	*	*	*	*	0	0	*	
0	0	0	*	*	0	0	0	
0	*	0	*	*	*	0	0	
0	0	*	1	*	*	*	0	
0	0	0	*	1	*	*	*	



if «rash» **and** «reaction» **and** «penicillin» **then** Allergy

Coordinating multiple rules – classification procedure

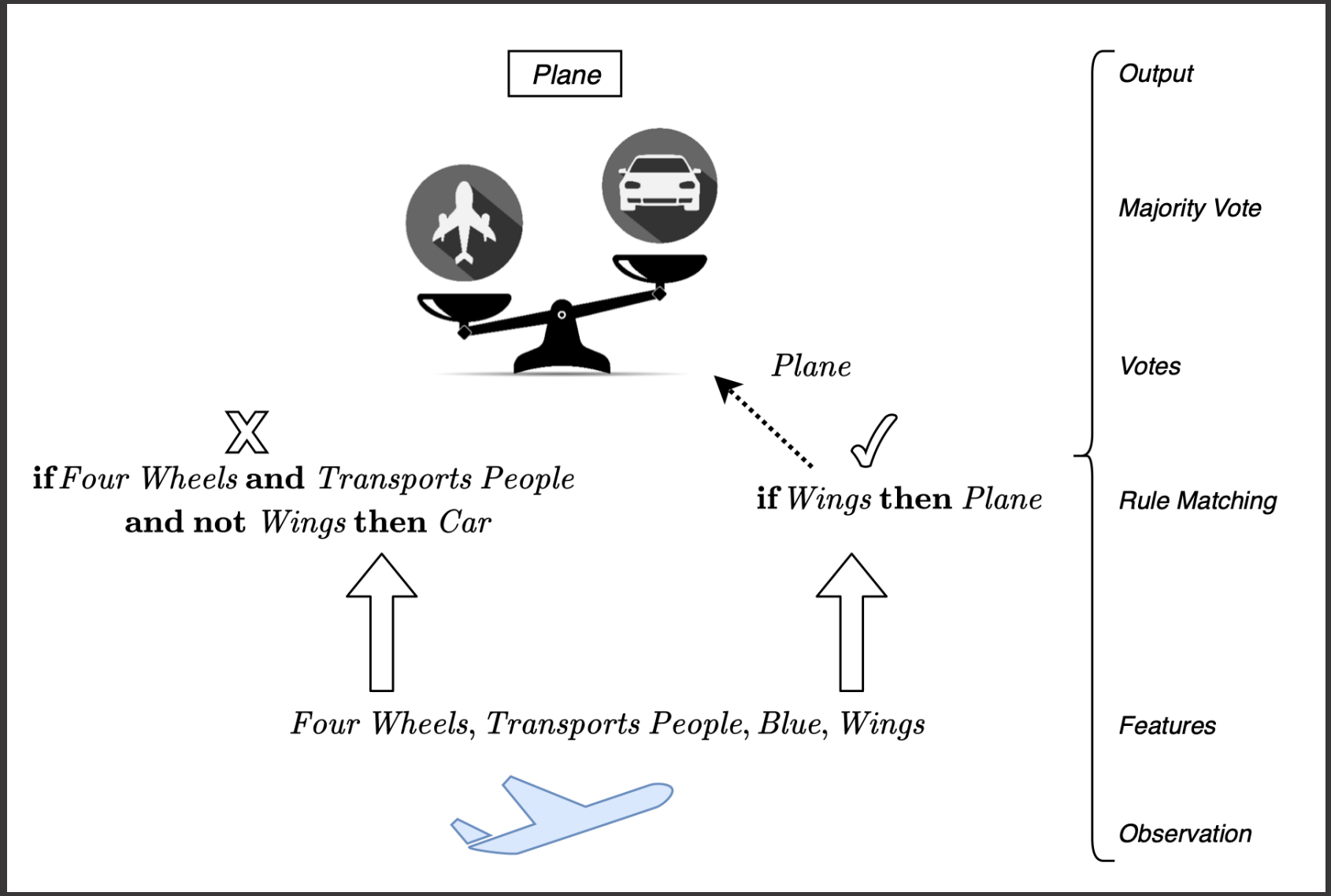


R1 **if** *Four Wheels* **and** *Transports People* **and not** *Wings* **then** *Car*.

R2 **if** *Wings* **then** *Plane*.

- A single rule does not get the final say on the class by itself. Instead, it casts a vote for its class.
- To classify, you count how many rules vote for each class.
- The Tsetlin machine then outputs the class with the most votes.
- *In other words, a majority vote among the rules decides the output.*

Tsetlin machine inference



	Menopause			Inv-nodes			Deg-malig			
#	**lt40**	**ge40**	**premeno**	**0-2**	**3-5**	**6-8**	**1**	**2**	**3**	***Recurrence***
1.	·	●	·	·	●	·	·	·	●	●
2.	●	·	·	●	·	·	·	·	●	·
3.	·	●	·	·	·	●	·	·	●	●
4.	·	●	·	●	·	·	·	●	·	·
5.	·	·	●	●	·	·	·	·	●	●
6.	·	·	●	●	·	·	●	·	·	·

R1 **if** *Deg-malign 3* **and not** *Menopause lt40* **then** *Recurrence*,

R2 **if** *Deg-malign 3* **and not** *Menopause lt40* **then** *Recurrence*,

R3 **if** *Inv-nodes 0-2* **then** *Non-Recurrence*.

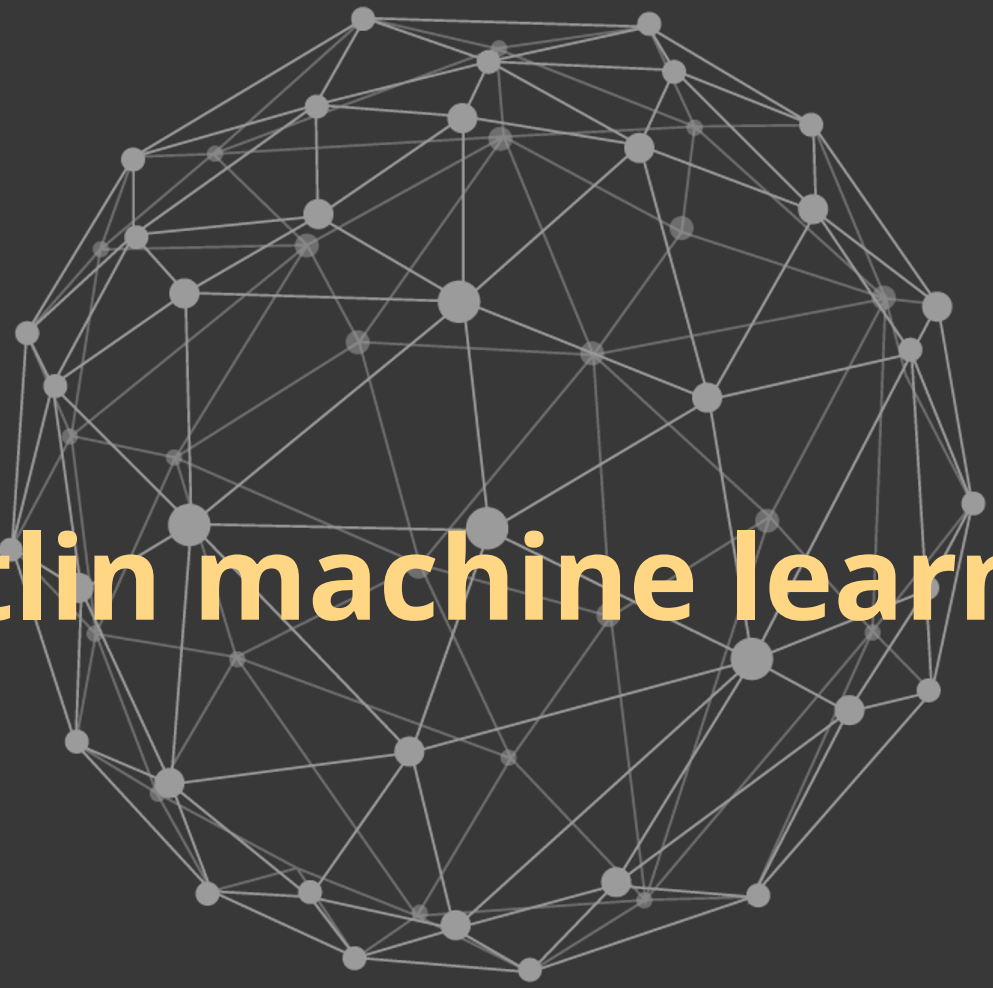
#	Menop.	Inv-nodes	Deg-malig	*Recur.*	R1	R2	R3	v
1.	ge40	3-5	3	*yes*	●	●	·	+2
2.	lt40	0-2	3	*no*	·	·	●	−1
3.	ge40	6-8	3	*yes*	●	●	·	+2
4.	ge40	0-2	2	*no*	·	·	●	−1
5.	premeno	0-2	3	*yes*	●	●	●	+1
6.	premeno	0-2	1	*no*	·	·	●	−1

R1 **if** *Deg-malign 3* **and not** *Menopause lt40* **then** *Recurrence*,

R2 **if** *Deg-malign 3* **and not** *Menopause lt40* **then** *Recurrence*,

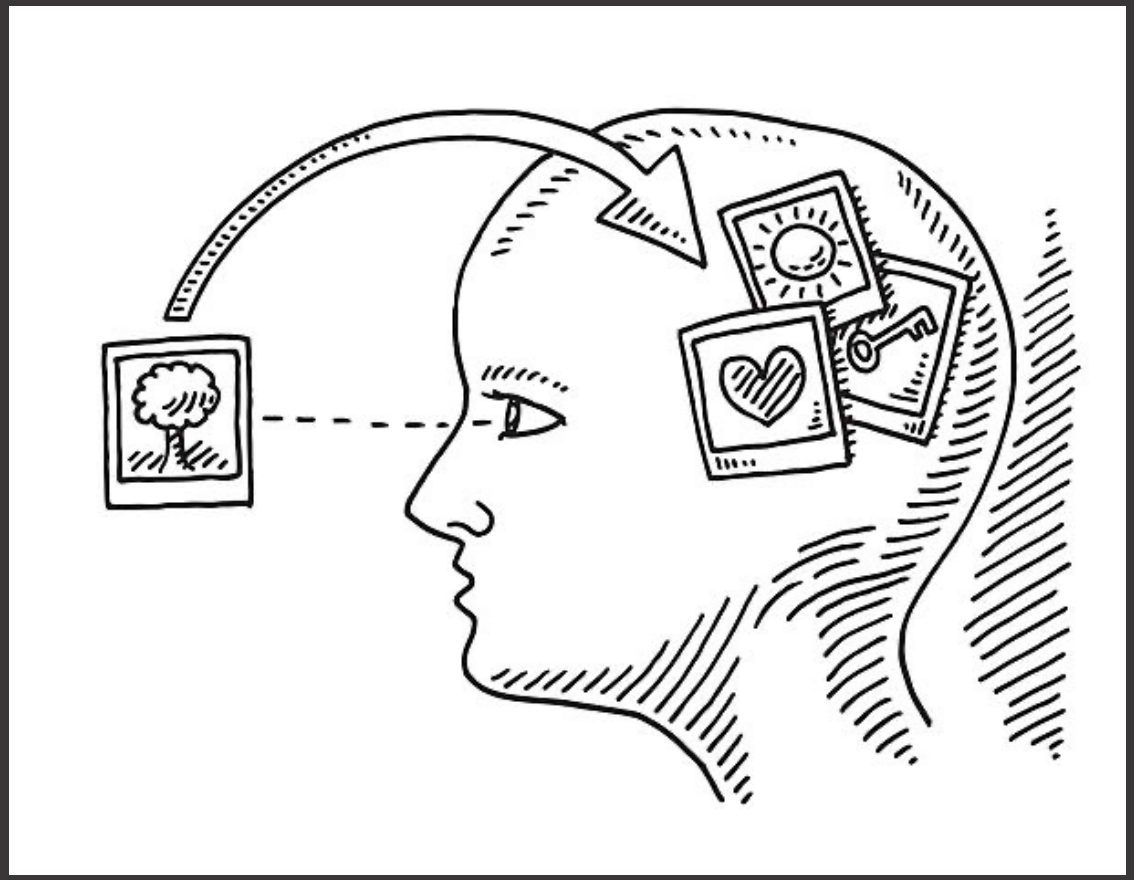
R3 **if** *Inv-nodes 0-2* **then** *Non-Recurrence*.

CAIR
CENTRE FOR ARTIFICIAL INTELLIGENCE RESEARCH



Tsetlin machine learning

Principle I: Memorization and generalization

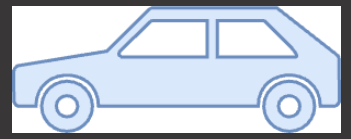


common denominator noun

Synonyms of *common denominator* ›

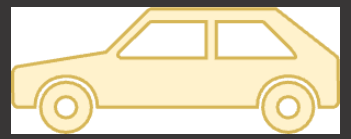
2 : a common trait or theme

> Drugs seem to be the ***common denominator*** in these crimes.



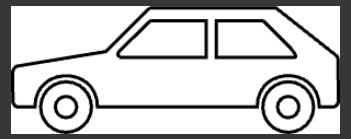
Four Wheels, Transports People, **not** Wings, **not** Yellow, Blue

AND



Four Wheels, Transports People, **not** Wings, Yellow, **not** Blue

=



Four Wheels, Transports People, **not** Wings

Principle II: Elimination

"When you have eliminated all which is impossible, then whatever remains, however improbable, must be the truth."

Sherlock Holmes

A **or** B
not A

B

not *Wings*

Eliminates planes

Tsetlin machine learning – step by step

A Tsetlin machine learns by remembering observations (input)

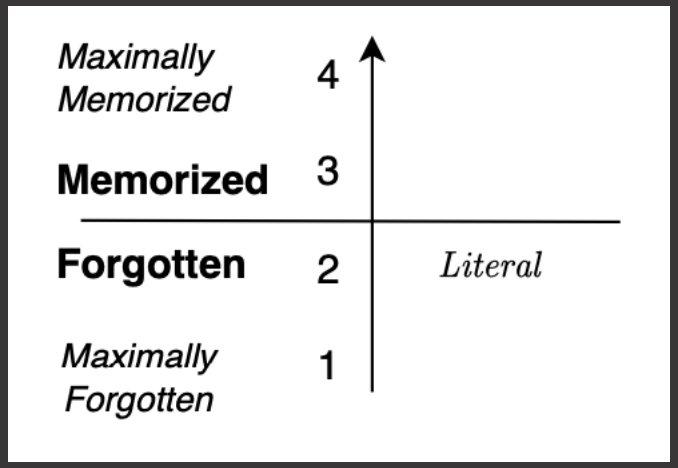
#	Four Wheels	Transports People	Wings	Yellow	Blue	*Car*
1.	●	●	·	·	●	●
2.	●	●	·	●	·	●
3.	●	●	·	●	·	●
4.	●	●	●	·	●	·
5.	●	·	●	●	·	·
6.	·	●	●	·	●	·

CAIR CENTRE FOR ARTIFICIAL INTELLIGENCE RESEARCH

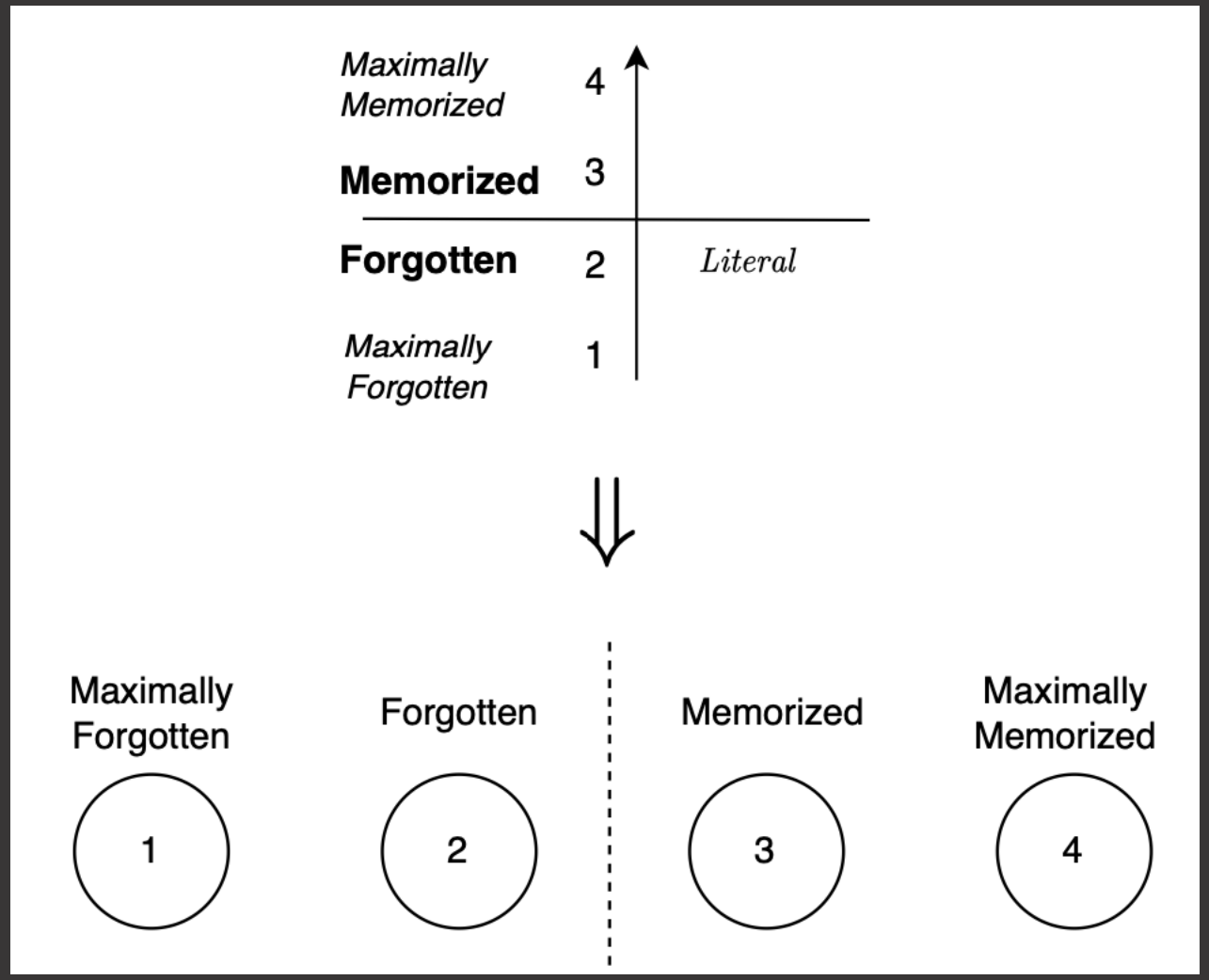
Traditional memory

In Memory	*Four Wheels*	*Transports People*				*Blue*	
Not in Memory	**not** *Four Wheels*	**not** *Transports People*	*Wings*	**not** *Wings*	*Yellow*	**not** *Yellow*	**not** *Blue*

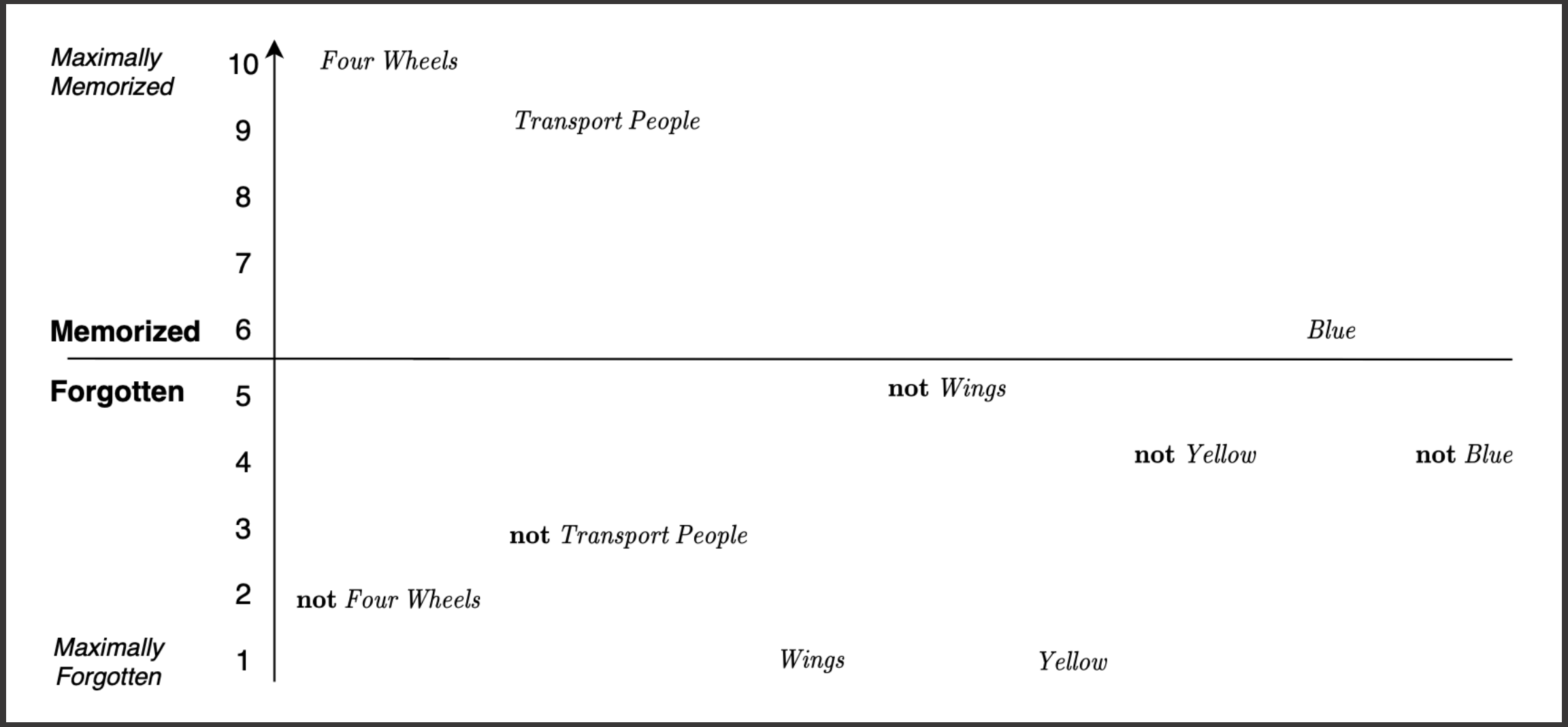
Tsetlin machine memory



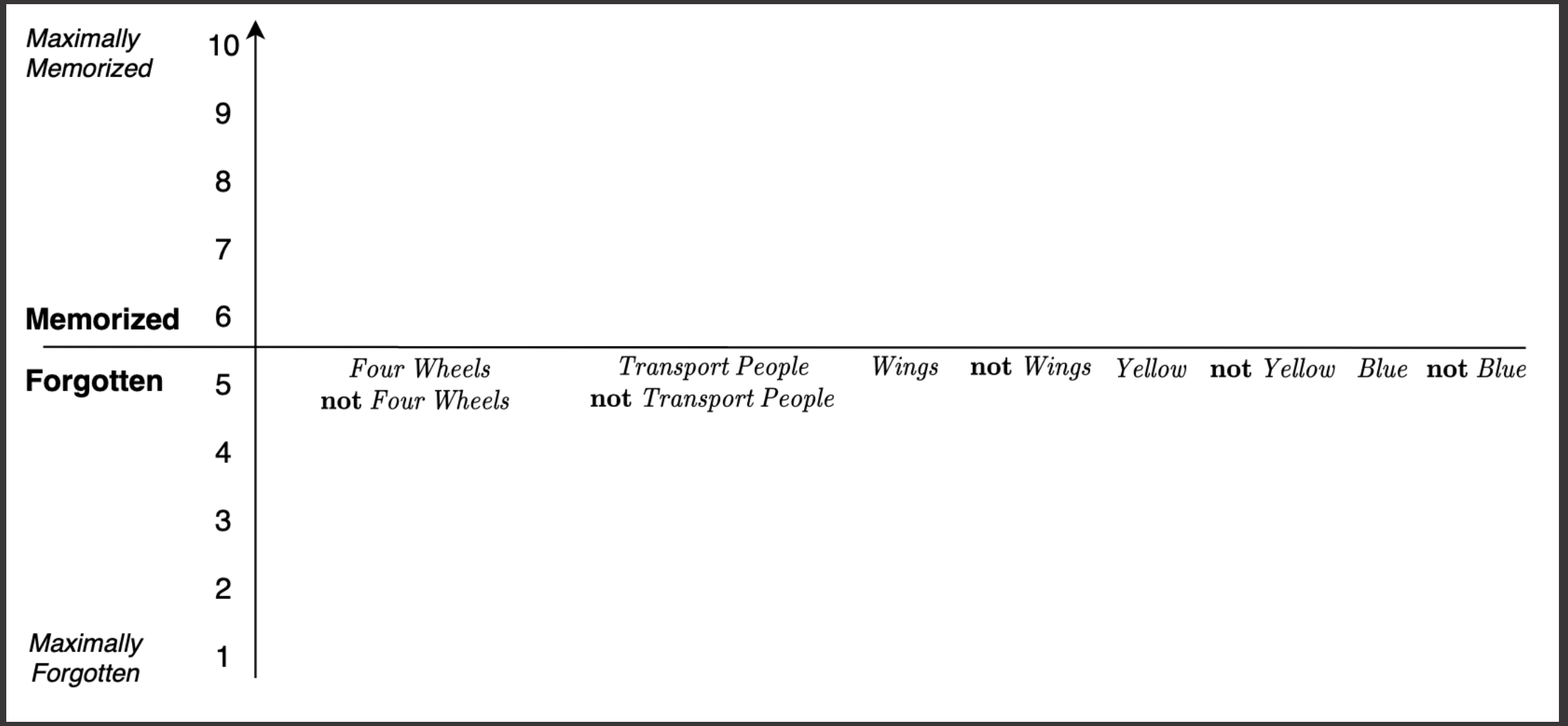
Literal automaton

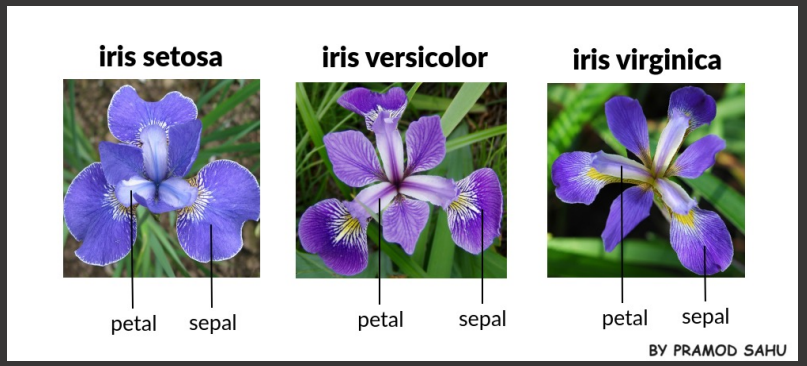


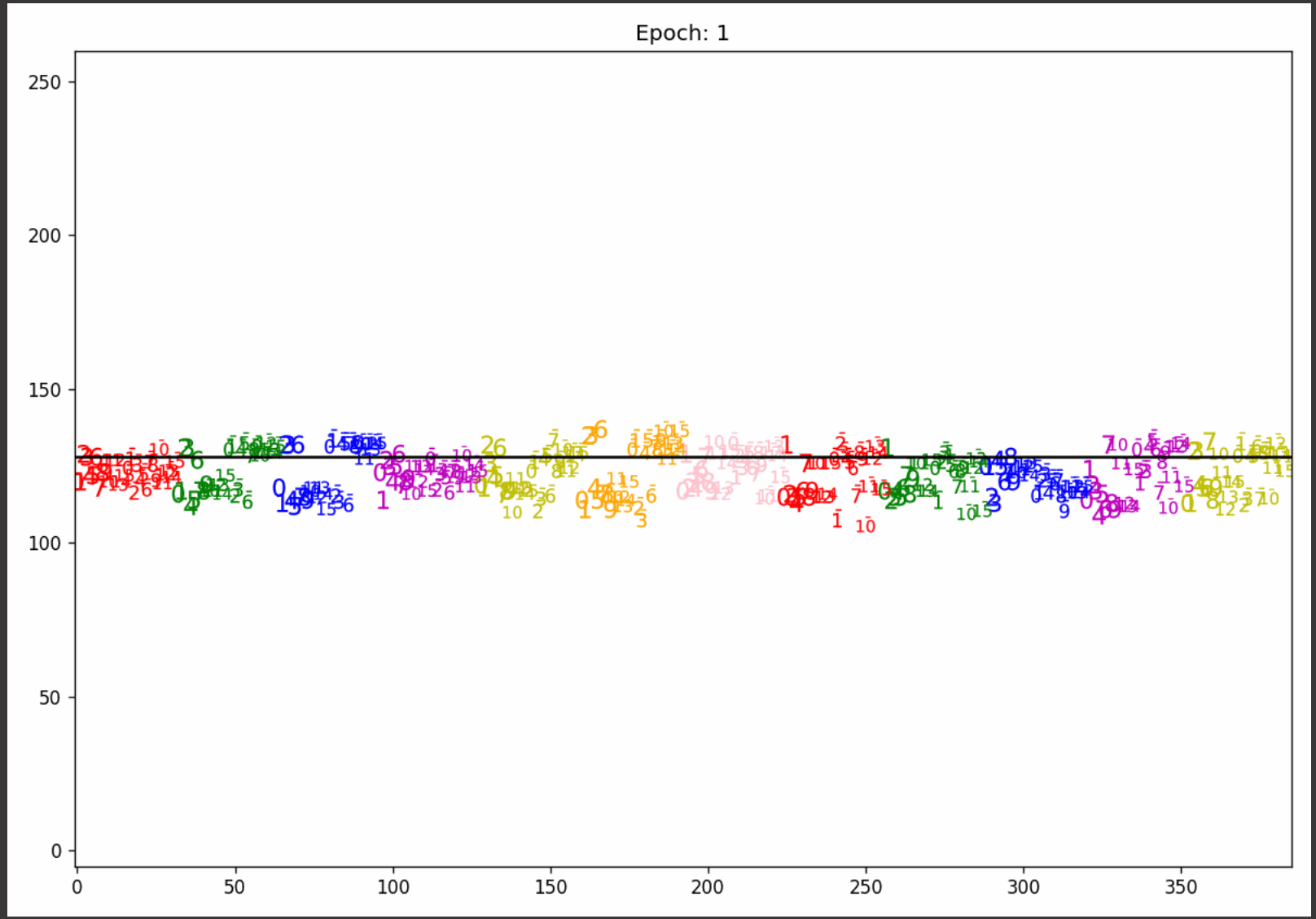
Tsetlin machine memory - example



Initialization



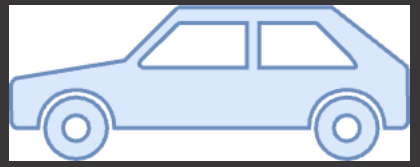




By Runar Helin, 2024

First step: evaluate rule on observation

Observation:



Four Wheels, Transports People, not Wings, not Yellow, Blue

Rule: **if** *Four Wheels* **and** *Transports People* **then** *Car*

Evaluation: *Four Wheels* **and** *Transports People* = *True*

Second step: feedback

Rule's condition

Rule's class

if *Four Wheels* **and** *Transports People* **then** *Car*

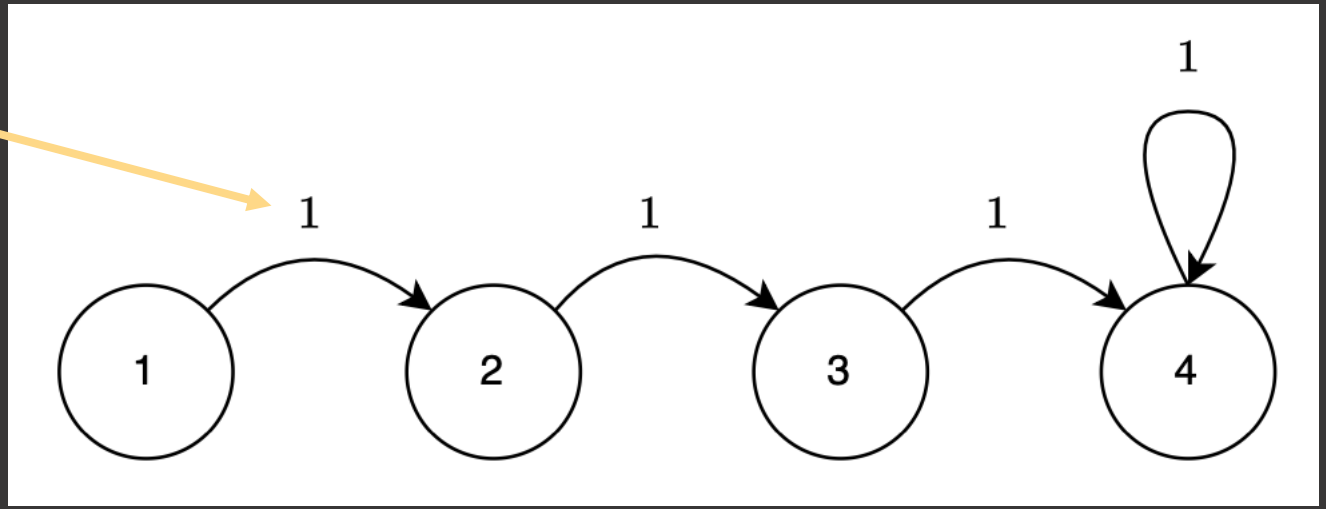
Observed class

1. Rule's class matches the observed class:
   a. *Recognize feedback* when the condition is *True*
   b. *Erase feedback* when the condition is *False*
2. Rule's class does not match the observed class:
   Reject feedback when the condition is *True*

Recognize feedback (rule's condition is *True*)

Memorize true literals by pushing them towards Maximally Memorized

Memorize Value

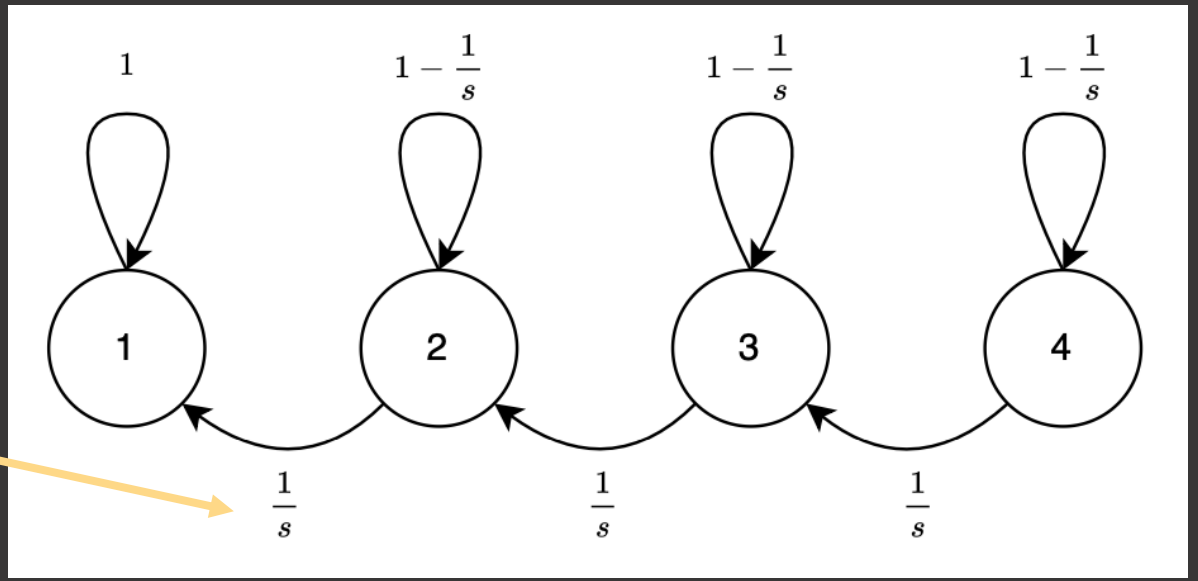


True literals

False literals

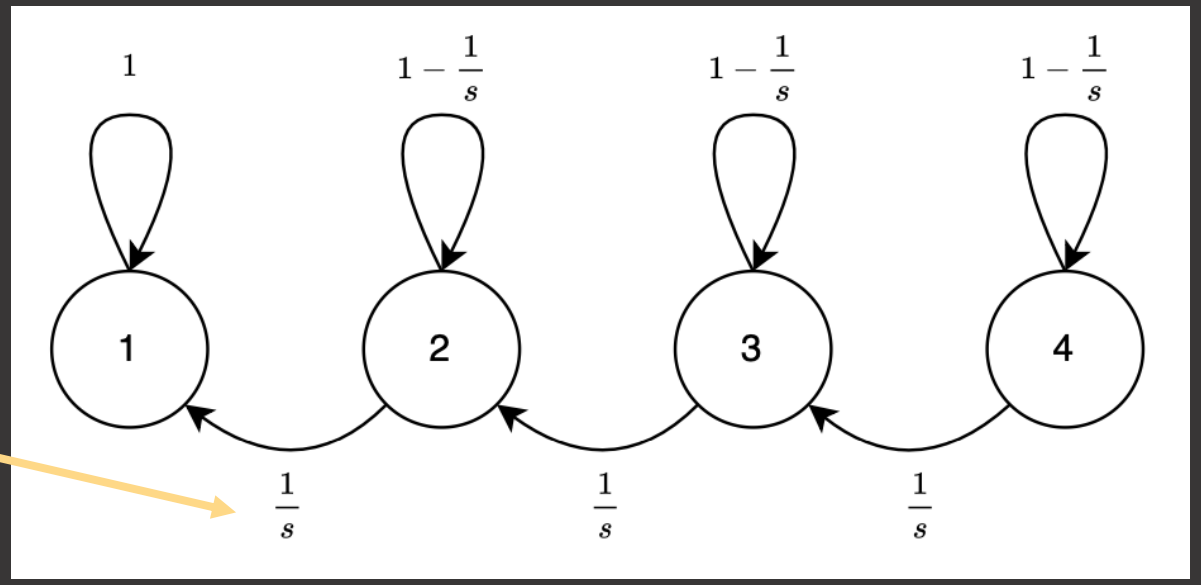
Forget false literals by pushing them towards Maximally Forgotten

Forget Value



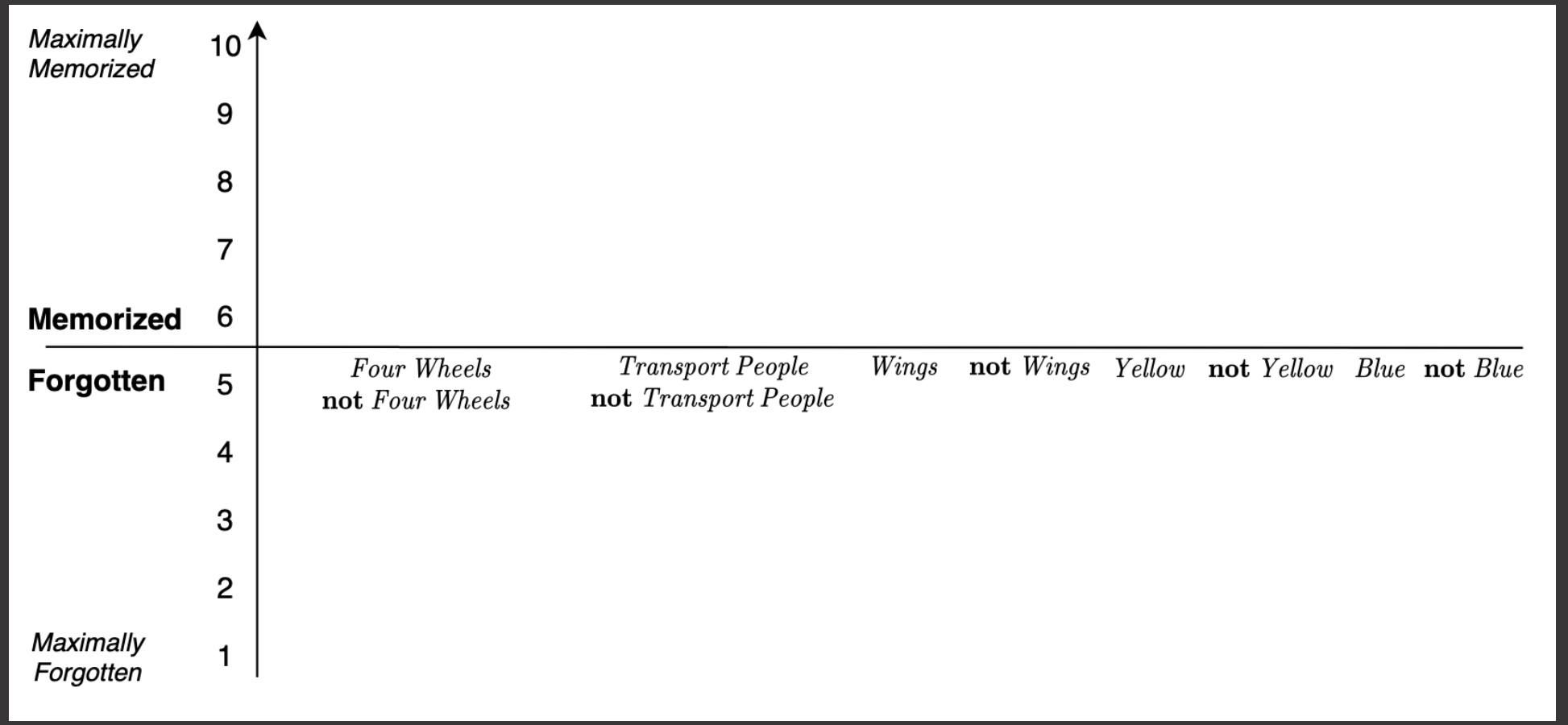
Erase feedback (rule's condition is *False*)

Forget Value



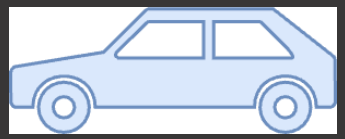
Forget all literals by pushing them towards Maximally Forgotten

Example of recognize feedback



if *True* **then** Car

Observation:



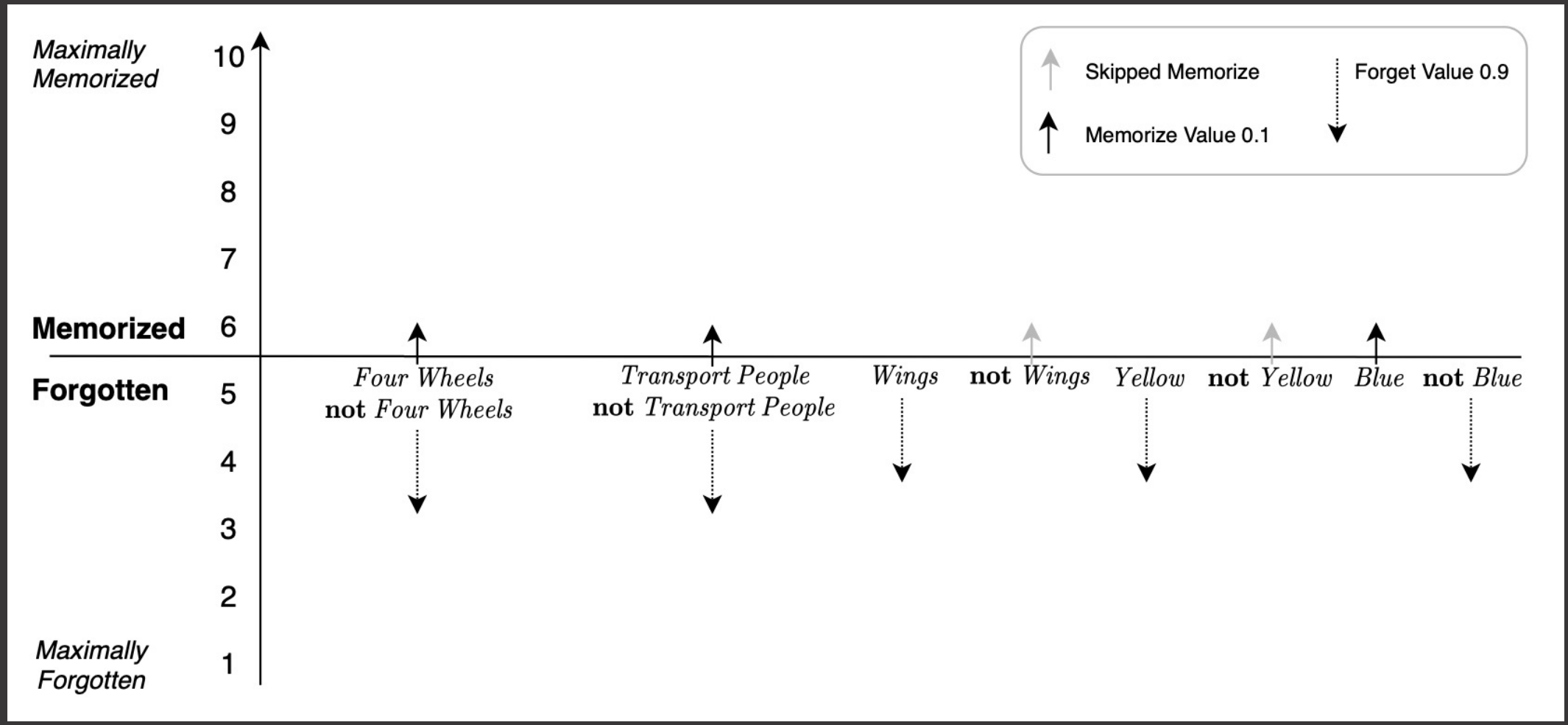
Car: Four Wheels, Transports People, **not** Wings, **not** Yellow, Blue

Rule: **if** *True* **then** Car

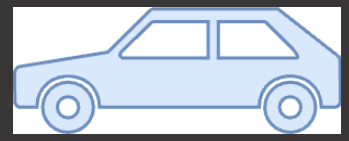
Evaluation: *True*

⇒ Recognize feedback

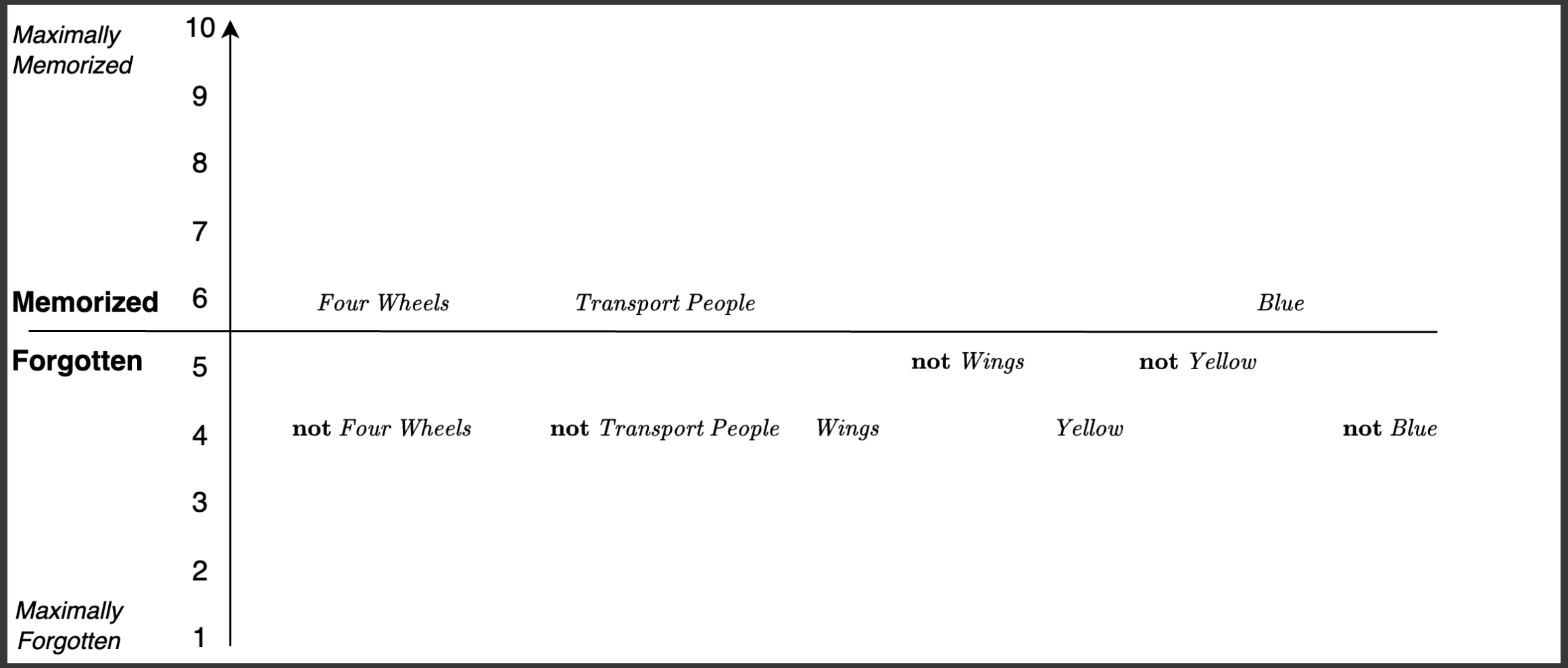
Recognize feedback



Car: Four Wheels, Transports People, not Wings, not Yellow, Blue



Updated memory



if *Four Wheels* **and** *Transport People* **and** *Blue* **then** *Car*

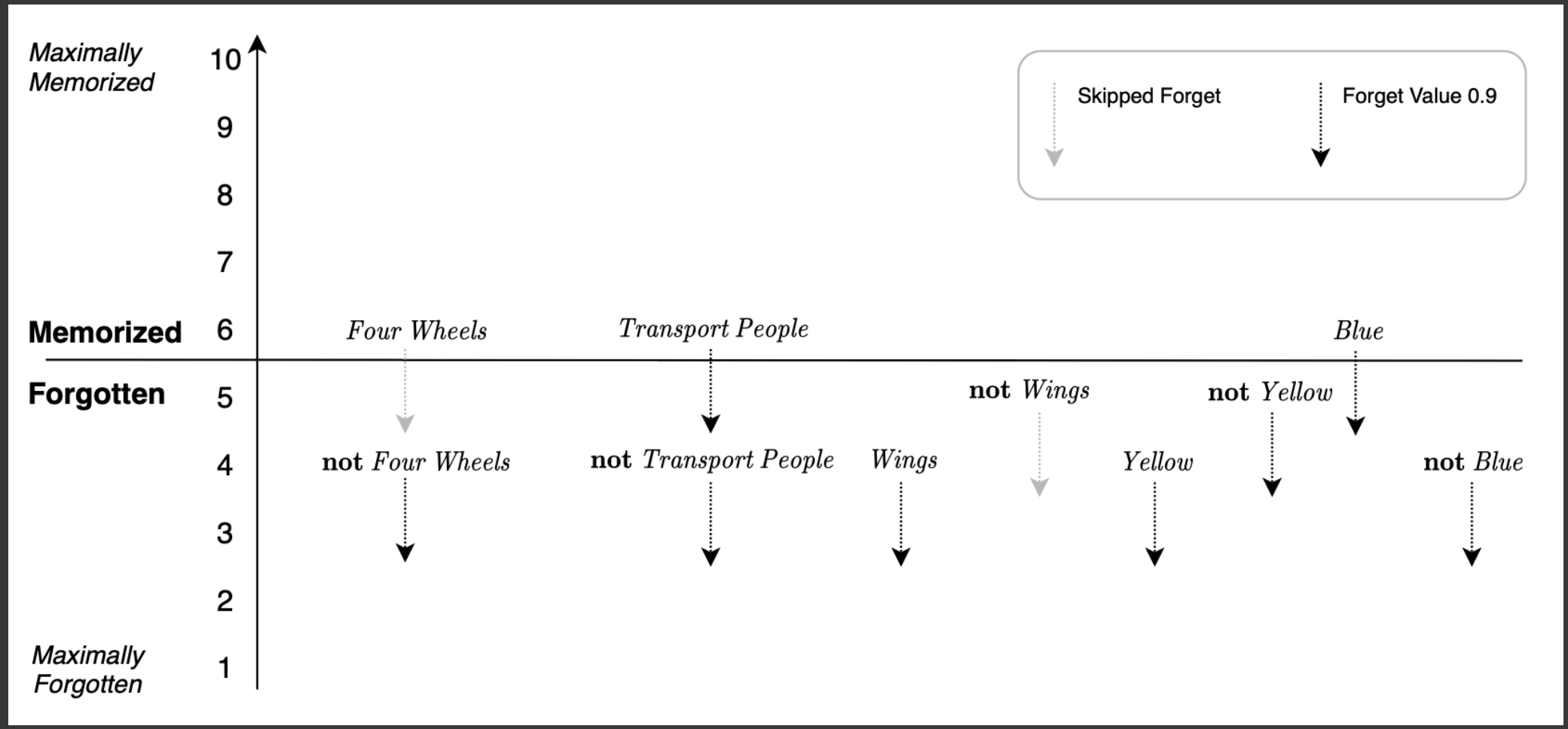
Observation:

Car: Four Wheels, Transports People, **not** Wings, Yellow, **not** Blue

Rule: **if** *Four Wheels* **and** *Transport People* **and** *Blue* **then** *Car*

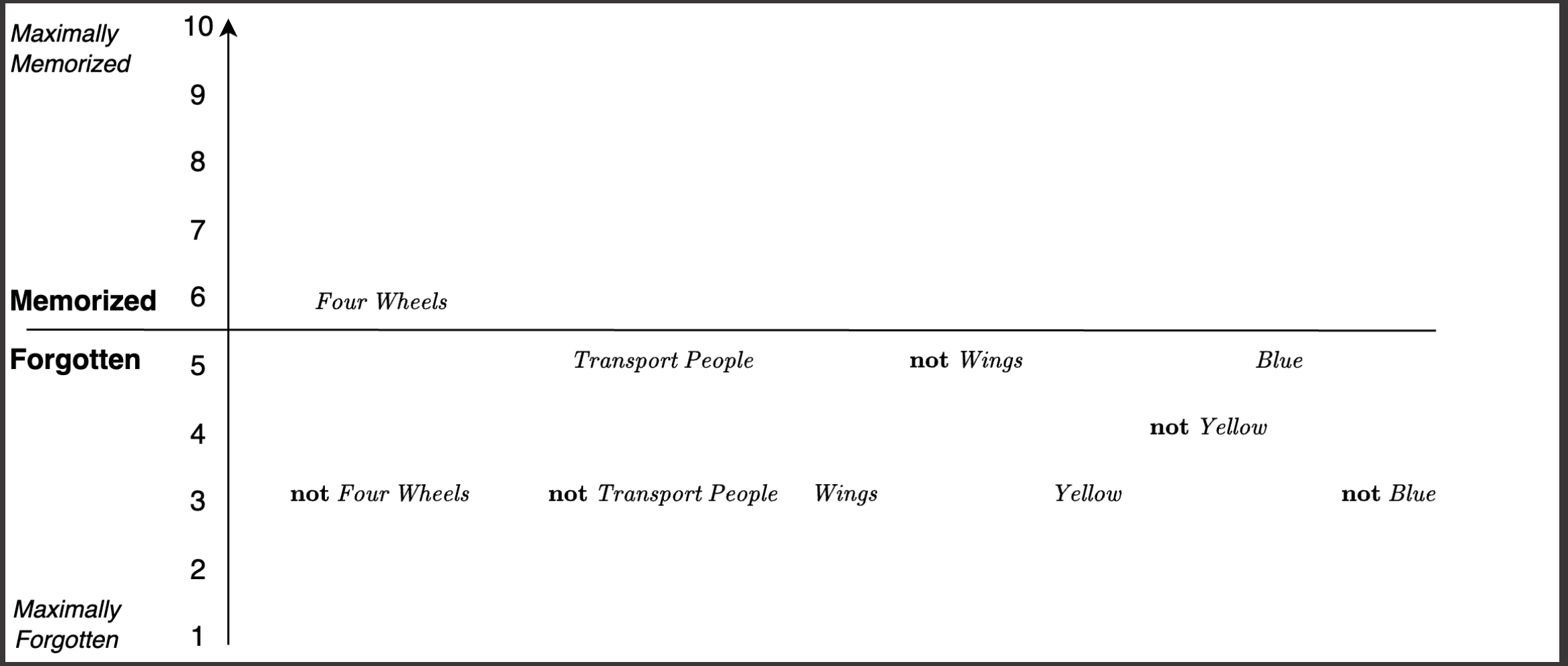
Evaluation: *True* **and** *True* **and** *False* = *False*

⇒ Erase feedback



Car: Four Wheels, Transports People, not Wings, Yellow, not Blue

Updated memory



if *Four Wheels* **then** *Car*

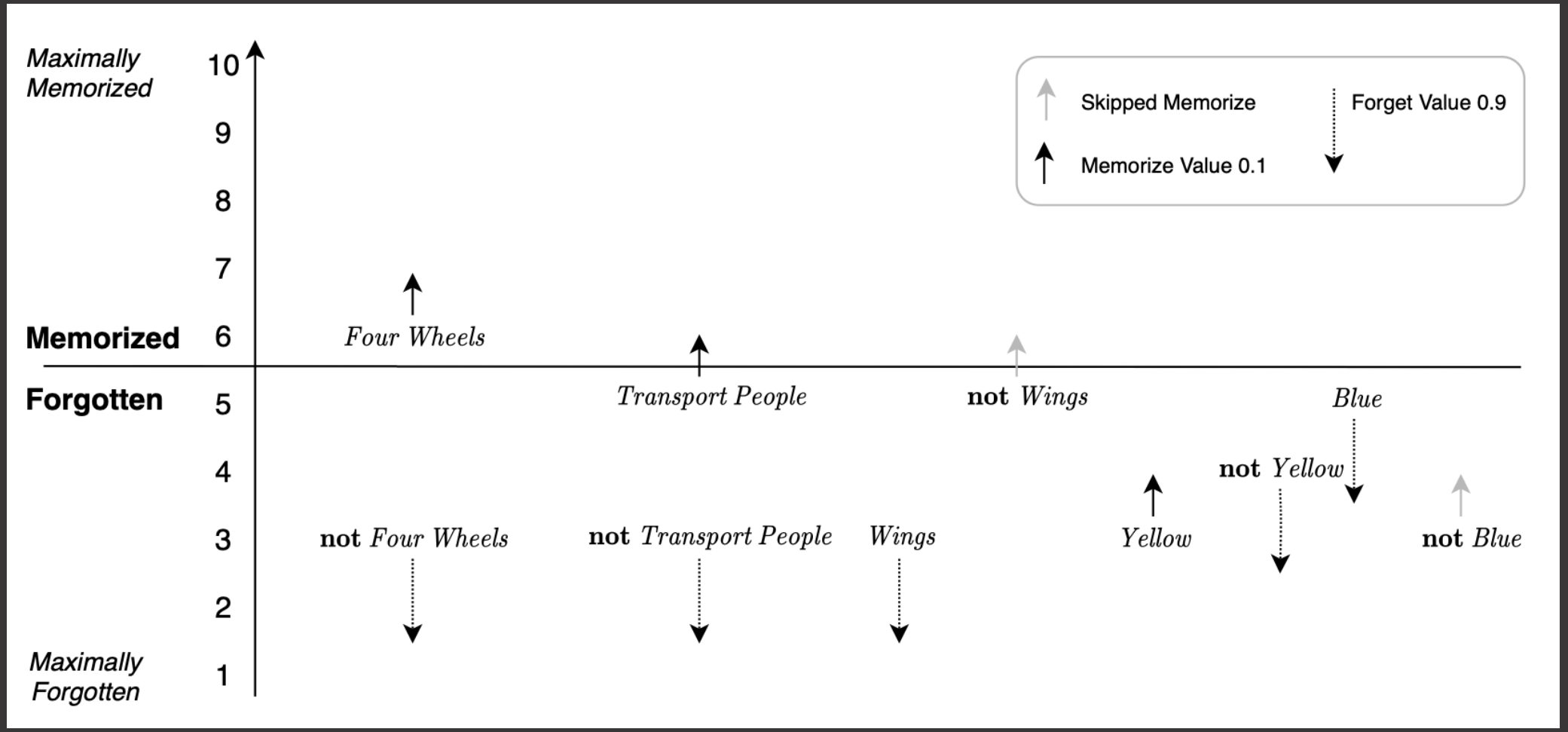
Observation:

Car: Four Wheels, Transports People, **not** Wings, Yellow, **not** Blue

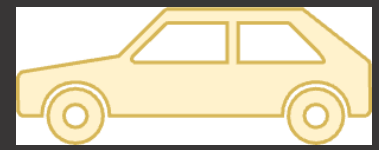
Rule: **if** *Four Wheels* **then** *Car*

Evaluation: *True*

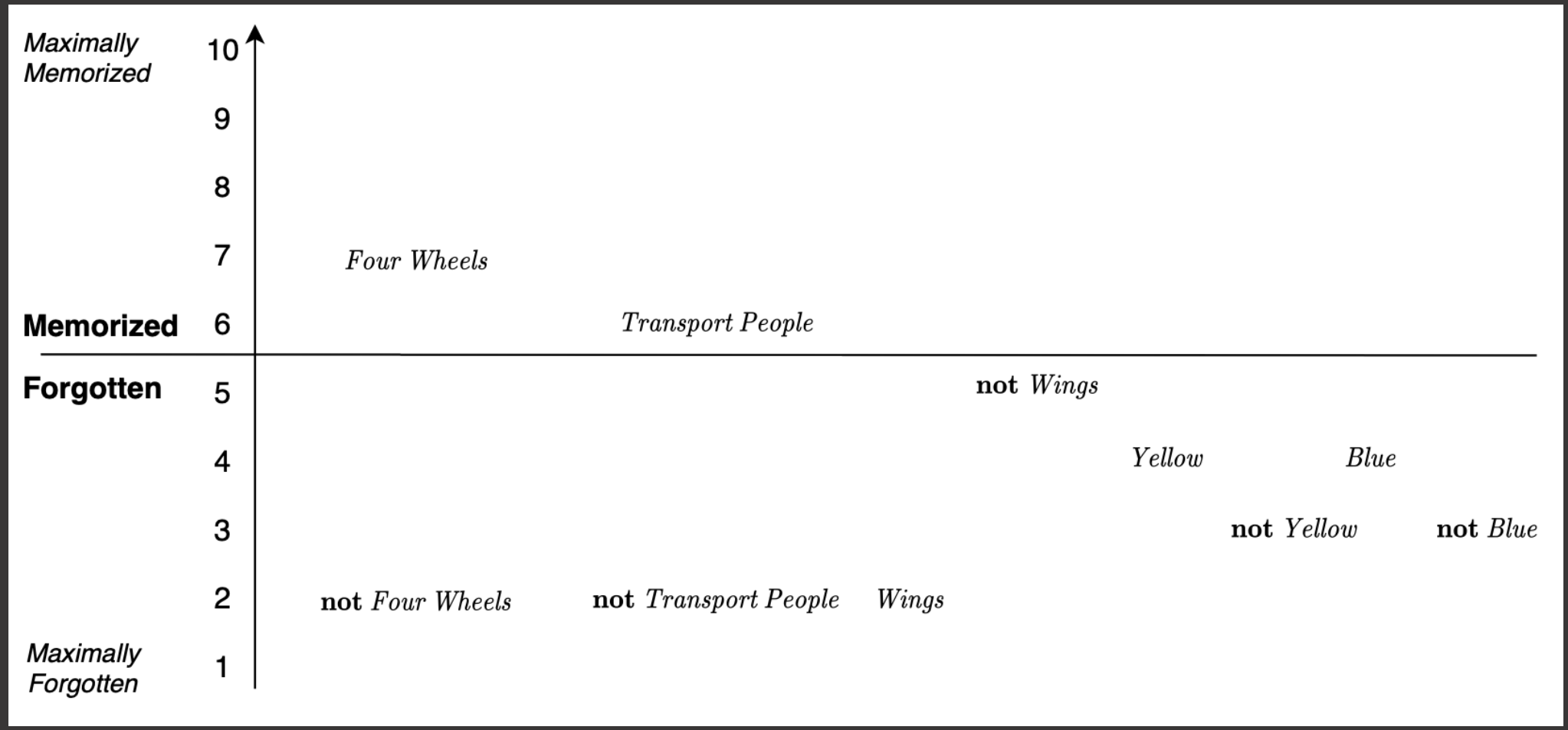
⇒ Recognize feedback



Car: Four Wheels, Transports People, not Wings, Yellow, not Blue



Updated memory



if *Four Wheels* **and** *Transport People* **then** *Car*

"When you have eliminated all which is impossible, then whatever remains, however improbable, must be the truth."

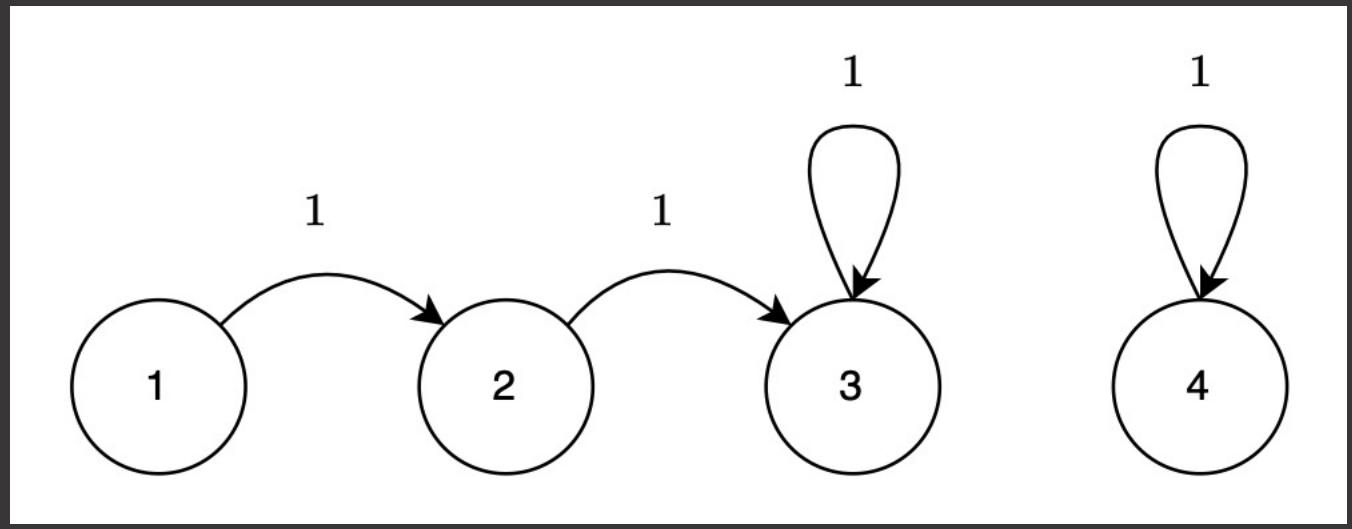
Sherlock Holmes

A **or** B
not A

B

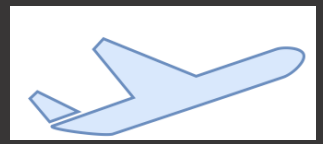
Increasing discrimination power with reject feedback (reasoning by elimination)

Occurs when the rule's condition is True



Memorize false literals by pushing them towards Maximally Memorized

Observation:

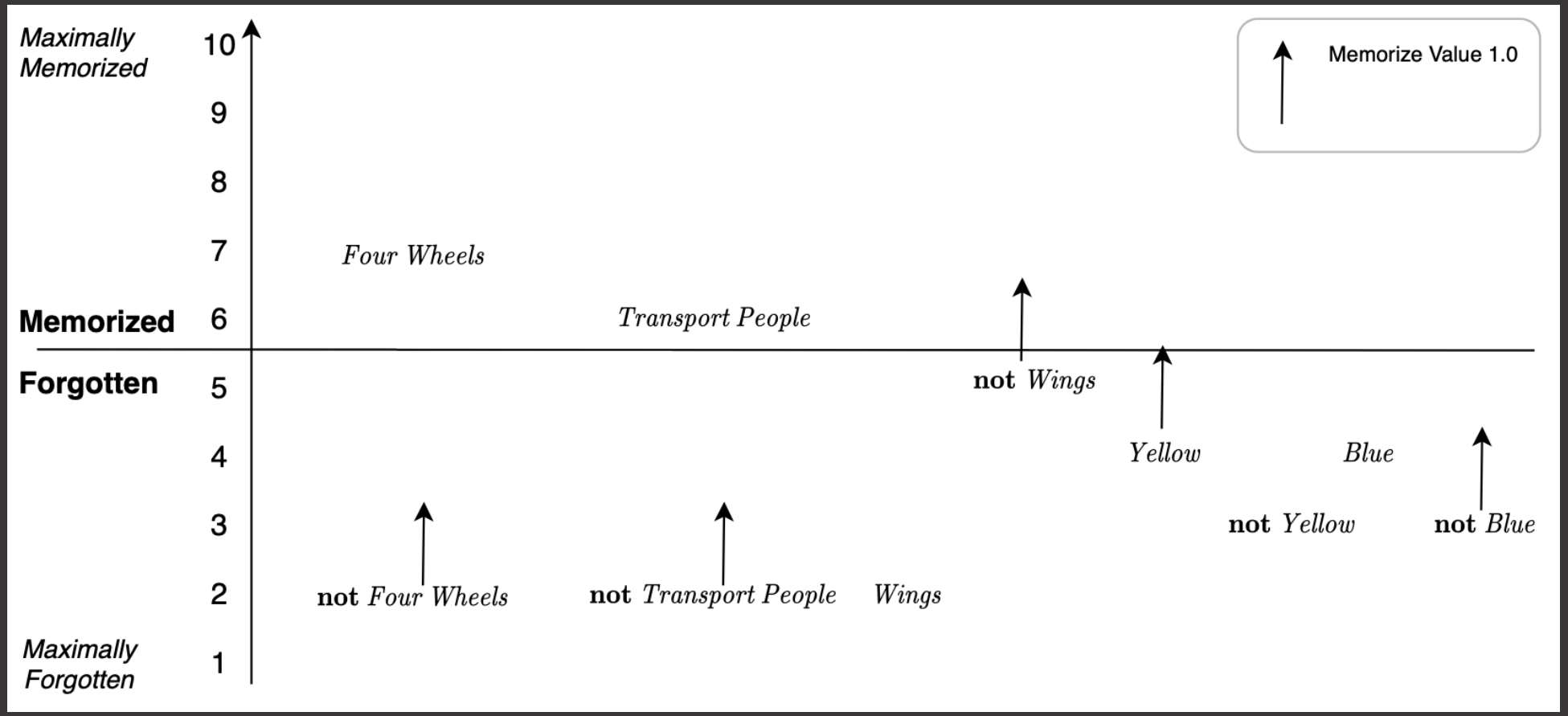


Plane: Four Wheels, Transports People, Wings,
not Yellow, Blue

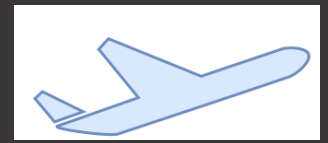
Rule: **if** *Four Wheels* **and** *Transport People* **then** *Car*

Evaluation: *True* **and** *True* = *True*

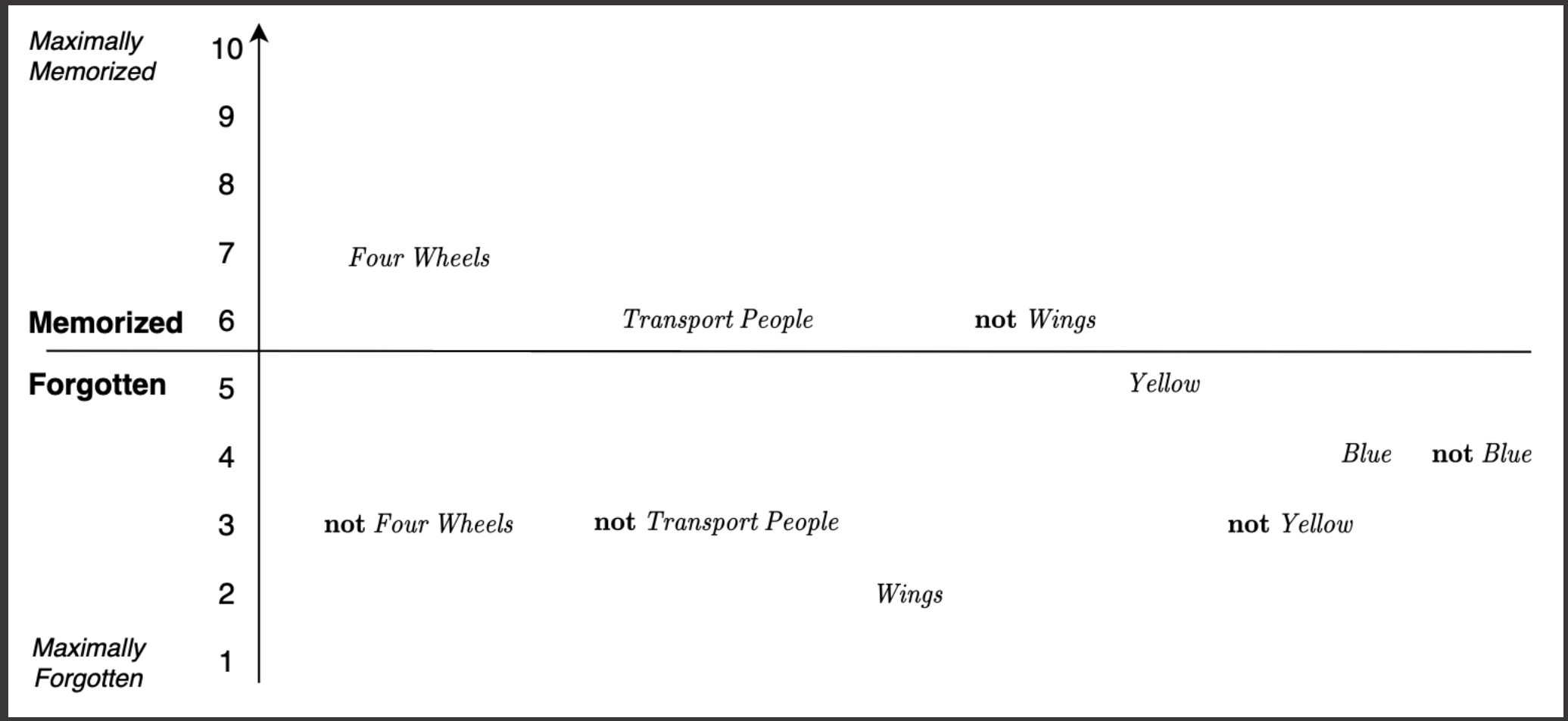
⇒ Reject feedback



Plane: Four Wheels, Transports People, Wings, not Yellow, Blue



Updated memory



if *Four Wheels* **and** *Transport People* **and not** *Wings* **then** *Car*

Complete learning algorithm

1. *Observe a new object and its class. The observation consists of the object's literals (see Section 1.1).*

2. *Evaluate each rule's condition using the truth values of the literals (see Section 1.2).*

3. *Calculate the vote sum (see Section 1.5):*

   a) *Identify the rules whose condition is* True. *Use these for voting.*

   b) *Add up the votes in favour of the object's class.*

   c) *Subtract the votes in favour of the other class.*

   d) *Refer to the summation outcome as* v.

   e) *Set* v *to the Vote Margin* 2 *if larger than* 2 *and to* -2 *if smaller than* -2.

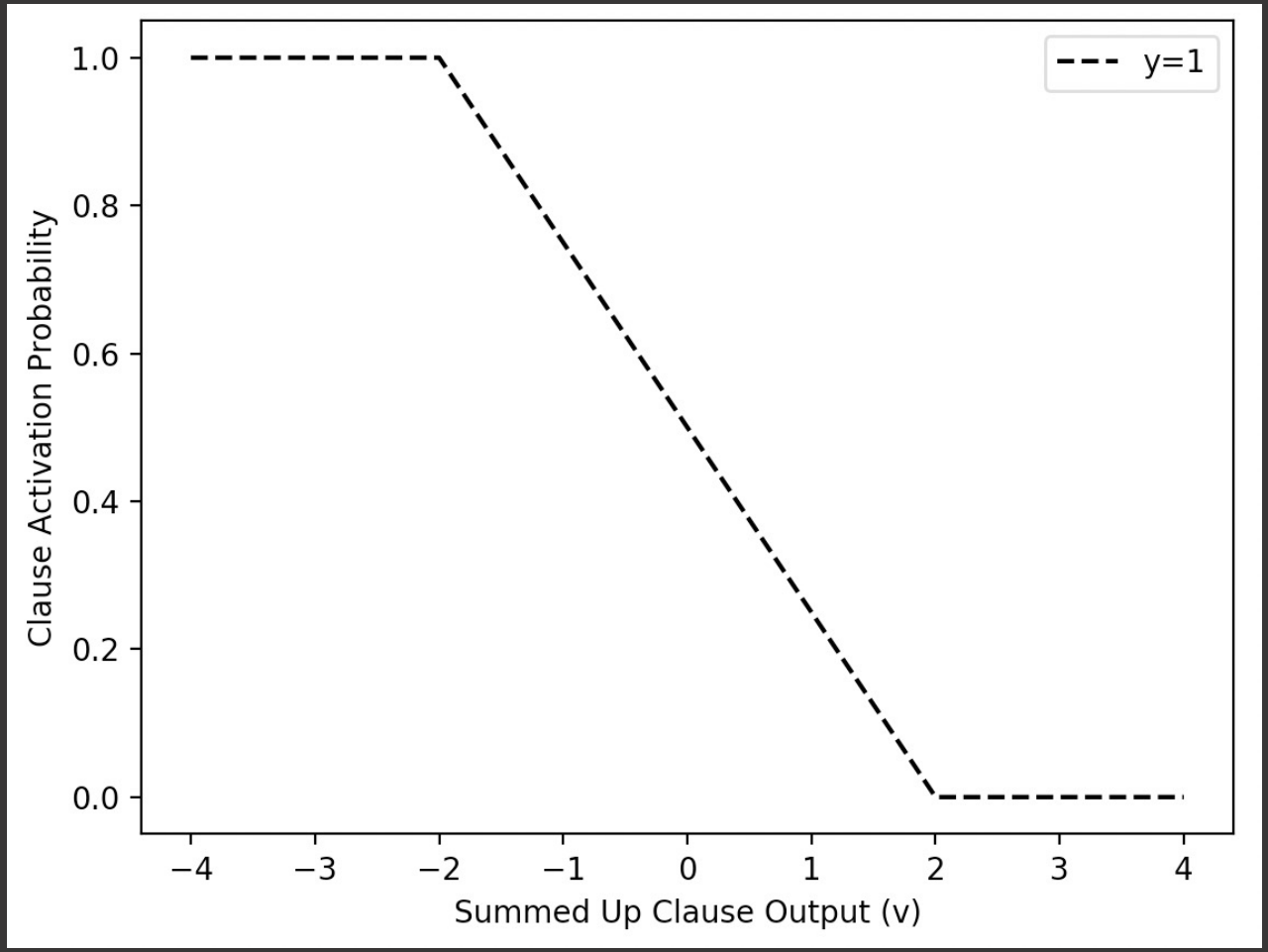
4. *Go through each rule and give it feedback if Rand*$() \leq \frac{2-v}{4}$, *drawn randomly per rule:*

   a) *Give the rule Recognize or Erase Feedback if the rule belongs to the object's class (see Section 1.3).*

   b) *Give the rule Reject Feedback if it belongs to another class (see Section 1.4).*

5. *Goto 1.*

Vote margin


y=1
1.0
0.8
0.6
0.4
0.2
0.0
Clause Activation Probability
−4
−3
−2
−1
0
1
2
3
4
Summed Up Clause Output (v)

Prioritization example

#	Menop.	Inv-nodes	Deg-malig	*Recur.*	R1	R2	R3	v
1.	ge40	3-5	3	*yes*	●	●	·	+2
2.	lt40	0-2	3	*no*	·	●	●	+0
3.	ge40	6-8	3	*yes*	●	●	·	+2
4.	ge40	0-2	2	*no*	·	·	●	−1
5.	premeno	0-2	3	*yes*	●	●	●	+1
6.	premeno	0-2	1	*no*	·	·	●	−1

R1 **if** *Deg-malign 3* **and not** *Menopause lt40* **then** *Recurrence*,

R2 **if** *Deg-malign 3* **then** *Recurrence*,

R3 **if** *Inv-nodes 0-2* **then** *Non-Recurrence*.

#	Menop.	Inv-nodes	Deg-malig	*Recur.*	R1	R2	R3	v
1.	ge40	3-5	3	*yes*	●	●	·	+2
2.	lt40	0-2	3	*no*	·	●	●	+0
3.	ge40	6-8	3	*yes*	●	●	·	+2
4.	ge40	0-2	2	*no*	·	·	●	−1
5.	premeno	0-2	3	*yes*	●	●	●	+1
6.	premeno	0-2	1	*no*	·	·	●	−1

- Observe how the vote sums of patients #1, #3, #4, #5, and #6 give the correct classification.
- However, patients #4, #5, and #6 do not reach the vote margin of 2.
- As a result, every time these patients appear you update each rule randomly if Rand() <= (2-1)/4, that is, one fourth of the time.
- Accordingly, patients #4, #5, and #6 are prioritized over patients #1 and #3.

#	Menop.	Inv-nodes	Deg-malig	*Recur.*	R1	R2	R3	v
1.	ge40	3-5	3	*yes*	●	●	·	+2
2.	lt40	0-2	3	*no*	·	●	●	+0
3.	ge40	6-8	3	*yes*	●	●	·	+2
4.	ge40	0-2	2	*no*	·	·	●	−1
5.	premeno	0-2	3	*yes*	●	●	●	+1
6.	premeno	0-2	1	*no*	·	·	●	−1

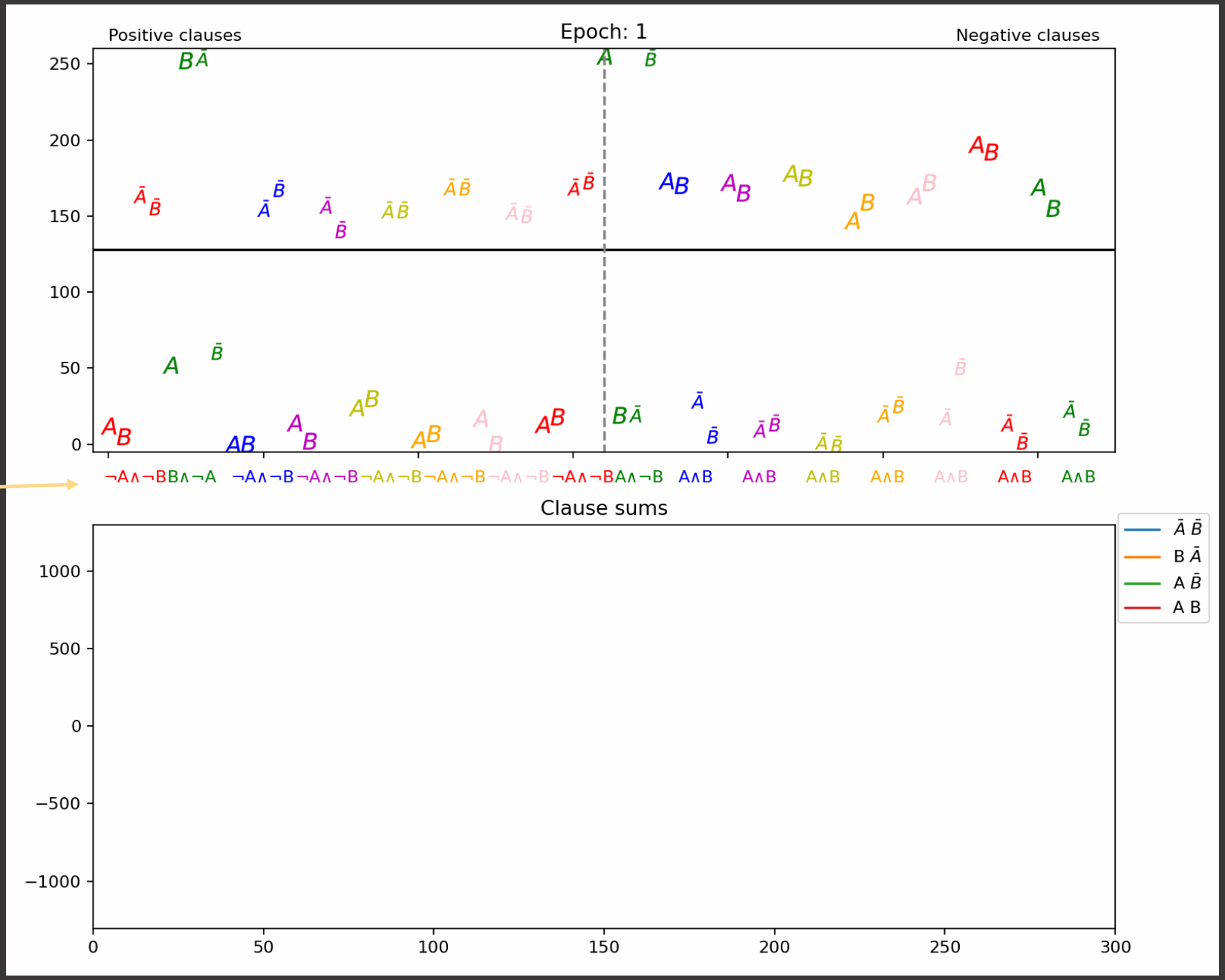
- The vote sum of patient #2 gives the wrong classification and is even further away from closing the margin.
- Every time this patient appears, you update each rule when Rand() is less than *(2-0)/4*, which is half of the time.
- So, patient #2 is prioritized over all the other patients.

UiA University of Agder Norway

CAIR CENTRE FOR ARTIFICIAL INTELLIGENCE RESEARCH

Tsetlin automata

Rules



Vote sums

A	B	P(X)
0	0	1.00
0	1	0.80
1	0	0.10
1	1	0.05

By Runar Helin, 2024

Assignment

1. Use the Jupyter Notebook for Chapter 1: https://github.com/cair/TsetlinMachineBook as a starting point
2. Create a dataset for the 6 breast cancer patients
3. Add the three breast cancer recurrence rules R1, R2, and R3 manually from the previous slide
4. Classify all the 6 patients with the three rules
5. Mix Type I and Type II Feedback with forget value 0.8 and memorize value 0.2 to learn a rule for Recurrence
6. Mix Type I and Type II Feedback with forget value 0.8 and memorize value 0.2 to learn a rule for Non-recurrence
7. Repeat 5. and 6. with forget value 0.5 and memorize value 0.5
8. Repeat 5. and 6. with forget value 0.2 and memorize value 0.8