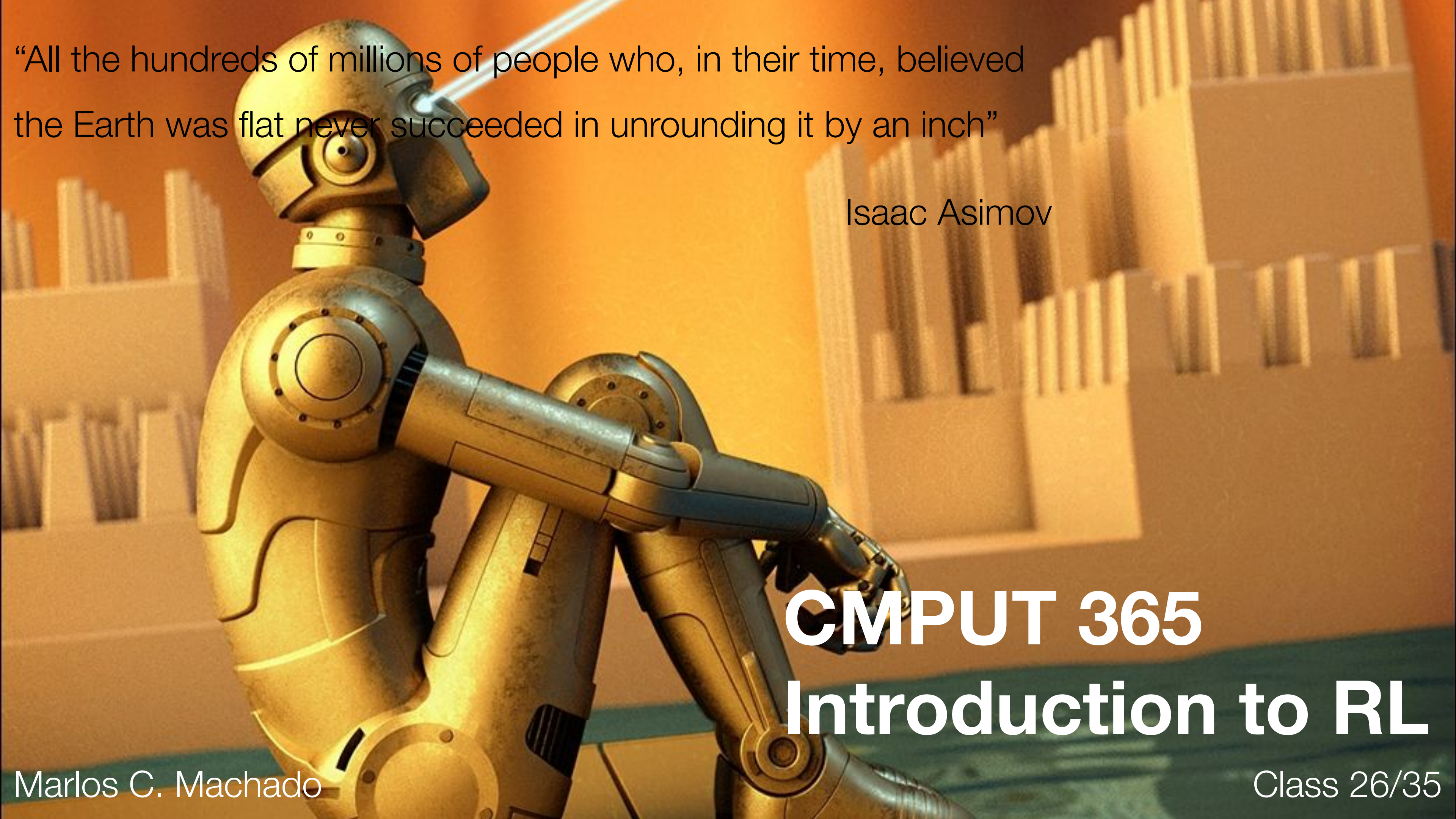


"All the hundreds of millions of people who, in their time, believed the Earth was flat never succeeded in unrounding it by an inch"

Isaac Asimov

CMPUT 365 Introduction to RL

Marlos C. Machado

Class 26/35

Reminder I

You **should be enrolled in the private session** we created in Coursera for CMPUT 365.

I **cannot** use marks from the public repository for your course marks.

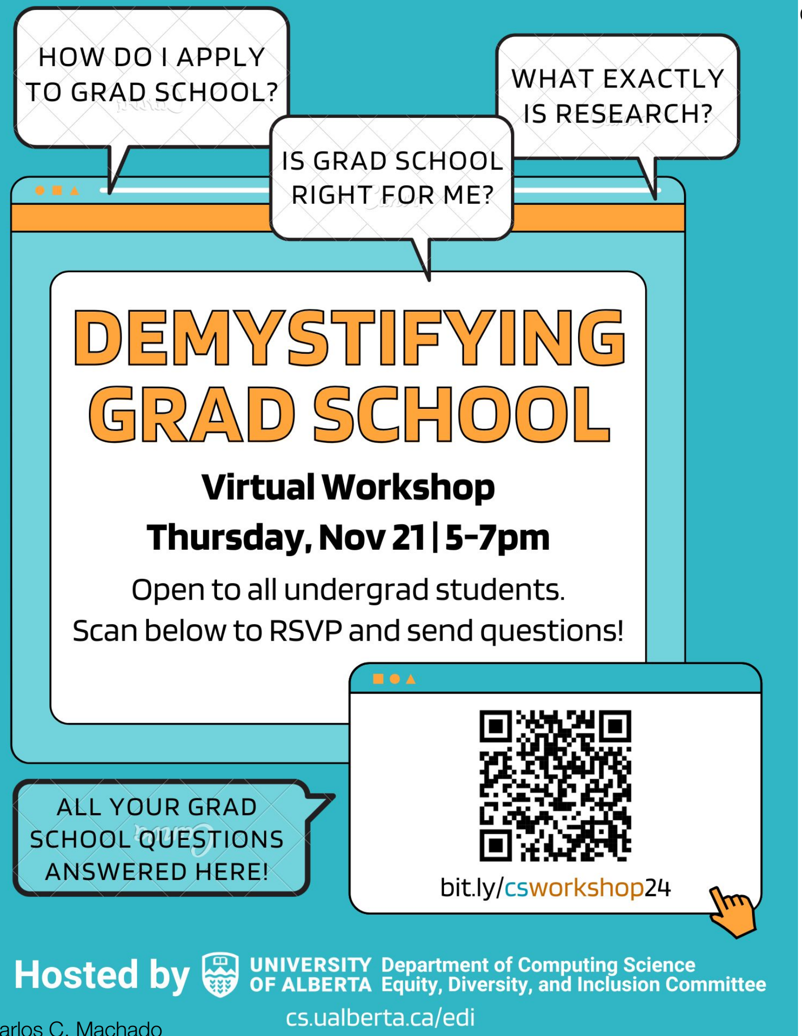
You **need** to **check**, **every time**, if you are in the private session and if you are submitting quizzes and assignments to the private section.

The deadlines in the public session **do not align** with the deadlines in Coursera.

If you have any questions or concerns, **talk with the TAs** or email us `cmput365@ualberta.ca`.

Reminders and Notes

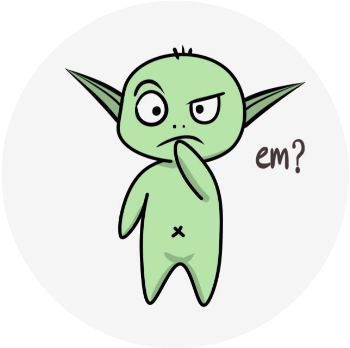
- The practice quiz for “Constructing features for prediction” is due today.
- The programming assignment is due on Wednesday.
- Rich Sutton will give a guest lecture Dec 9th, Monday. Spread the word.
- A note on the final exam:
    - The required reading from the syllabus does not mean that’s what will be covered in the final exam. There are some mismatches. Anything we discussed in class is fair game, including Maximization Bias and Double Learning (Section 6.7), and Nonlinear Function Approximation: Artificial Neural Networks (Section 9.7).
- I have a meeting at 2pm today that I cannot be late for.



RSVP form (not required, but appreciated):
https://docs.google.com/forms/d/1lodJJgO3kgJ_XFDq9vnEz4FABjlNKx7-iL6AkCJ67ZQ/edit

Direct link to the zoom:
https://ualberta-ca.zoom.us/j/93282952849?pwd=eqE7hm46hwMJS02EZoqjw5GOngtWkK.1

Please, interrupt me at any time!



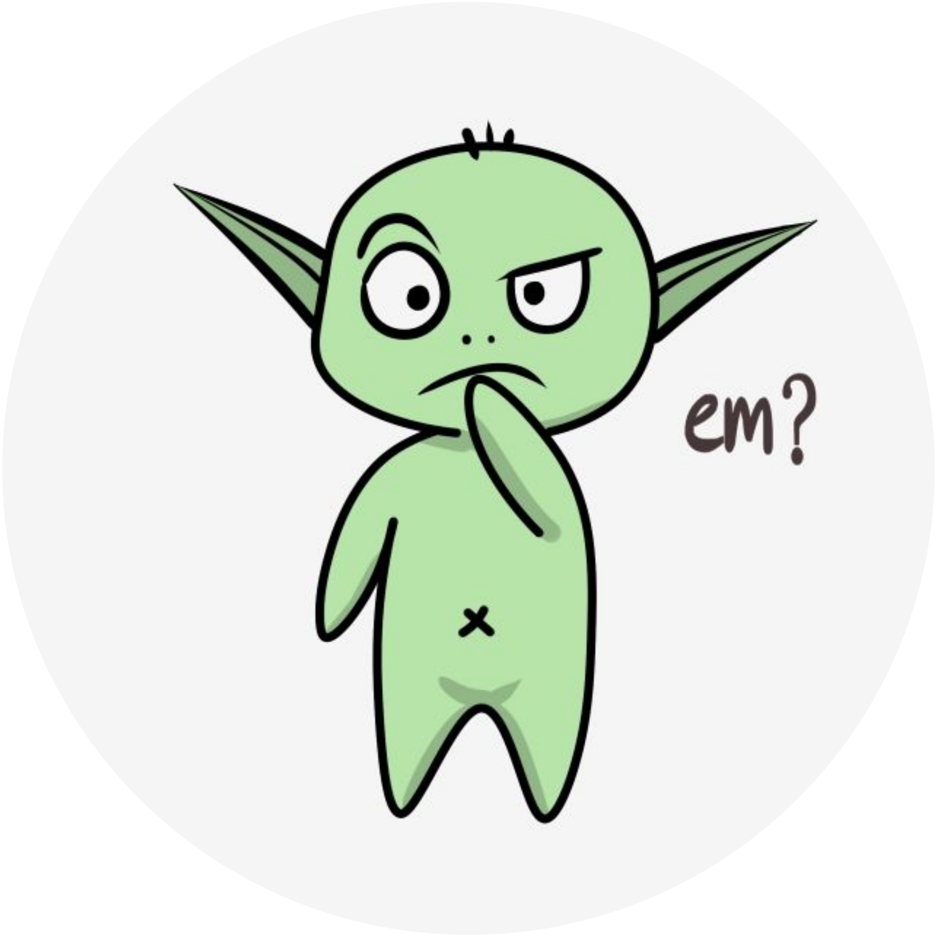
https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

Two weeks ago: Semi-gradient TD with Function Approximation

Semi-gradient TD(0) for estimating $\hat{v} \approx v_\pi$

Input: the policy π to be evaluated **What approximation should we use?!**
Input: a differentiable function $\boxed{\hat{v}}: \mathcal{S}^+ \times \mathbb{R}^d \to \mathbb{R}$ such that $\hat{v}(\text{terminal},\cdot) = 0$
Algorithm parameter: step size $\alpha > 0$
Initialize value-function weights $\mathbf{w} \in \mathbb{R}^d$ arbitrarily (e.g., $\mathbf{w} = \mathbf{0}$)

Loop for each episode:
  Initialize S
  Loop for each step of episode:
    Choose $A \sim \pi(\cdot|S)$
    Take action A, observe R, S'
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha\big[R + \gamma\hat{v}(S',\mathbf{w}) - \hat{v}(S,\mathbf{w})\big]\nabla\hat{v}(S,\mathbf{w})$
    $S \leftarrow S'$
  until S is terminal



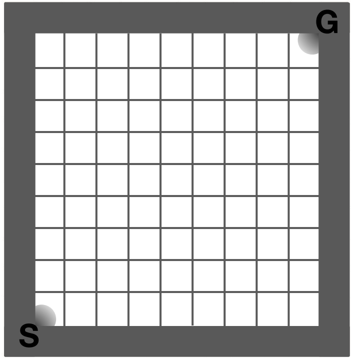
https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

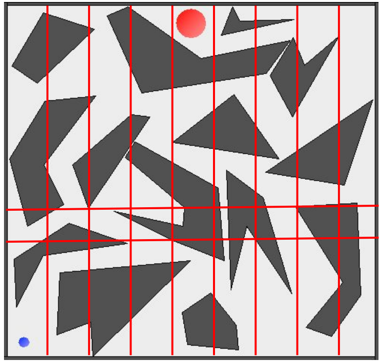
Feature Construction for Linear Methods

- Linear methods can be effective, but they heavily rely on how states are represented in terms of features.
- Feature construction is a way of adding domain knowledge; but at the same time, it went out of fashion because of *deep reinforcement learning*.
- Naïve linear function approximation methods do not take into consideration the interaction between features.

State Aggregation

- Simplest form of representation
- States are grouped together (one component of the vector **w**) for each group.
- State aggregation is a special case of SGD in which the gradient, $\nabla \hat{v}(S_t, \mathbf{w}_t)$, is 1 for S_t's group's component and 0 for the other components.





http://irl.cs.brown.edu/pinball/

Polynomials

- Doesn’t work so well, but they are one of the simplest families of features.

Polynomials

- Doesn’t work so well, but they are one of the simplest families of features.
- Suppose an RL problem has states with two numerical dimensions.

$$\mathbf{x}(s) = (s_1, s_2)^\top$$

Polynomials

- Doesn’t work so well, but they are one of the simplest families of features.
- Suppose an RL problem has states with two numerical dimensions.

$$\mathbf{x}(s) = (s_1, s_2)^\top$$

But what about interactions? What if both features were zero?

$$\mathbf{x}(s) = (1, s_1, s_2, s_1 s_2)^\top$$

Polynomials

- Doesn’t work so well, but they are one of the simplest families of features.
- Suppose an RL problem has states with two numerical dimensions.

$$\mathbf{x}(s) = (s_1, s_2)^\top$$

But what about interactions? What if both features were zero?

$$\mathbf{x}(s) = (1, s_1, s_2, s_1 s_2)^\top$$

And we can keep going…

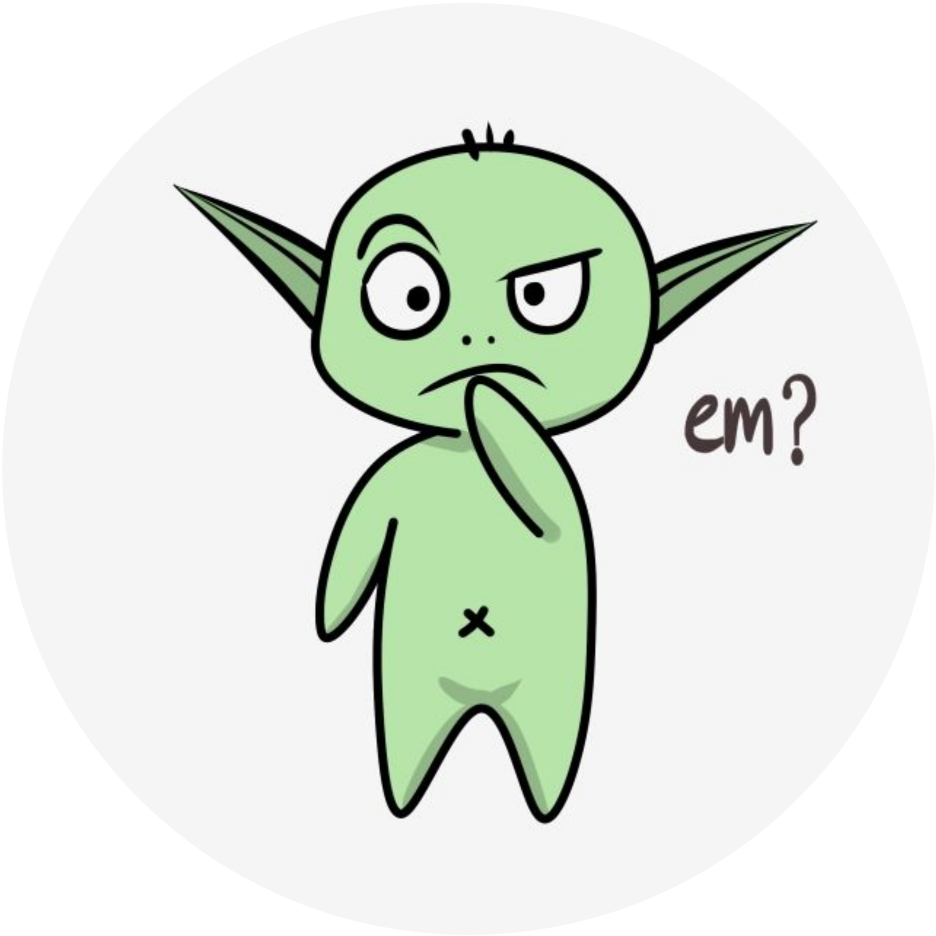
$$\mathbf{x}(s) = (1, s_1, s_2, s_1 s_2, s_1^2, s_2^2, s_1 s_2^2, s_1^2 s_2, s_1^2 s_2^2)^\top$$

Polynomials

Suppose each state s corresponds to k numbers, s_1, s_2, ..., s_k, with each $s_i \in \mathbb{R}$. For this k-dimensional state space, each order-n polynomial-basis feature x_i can be written as

$$x_i(s) = \Pi_{j=1}^{k} s_j^{c_{i,j}}, \tag{9.17}$$

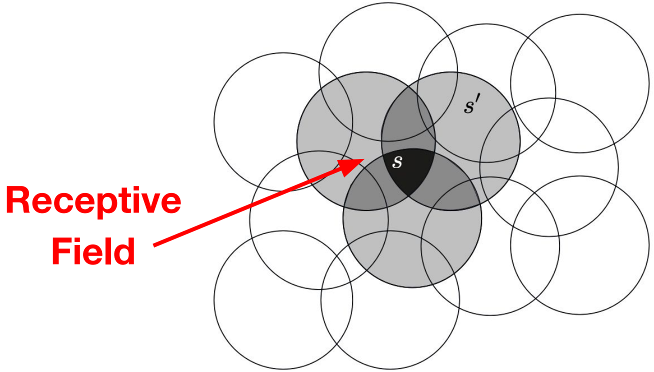
where each $c_{i,j}$ is an integer in the set $\{0, 1, \ldots, n\}$ for an integer $n \geq 0$. These features make up the order-n polynomial basis for dimension k, which contains $(n+1)^k$ different features.



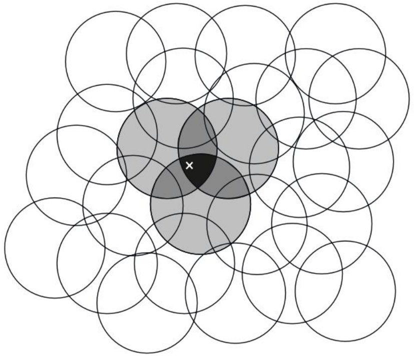
https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht

Coarse Coding

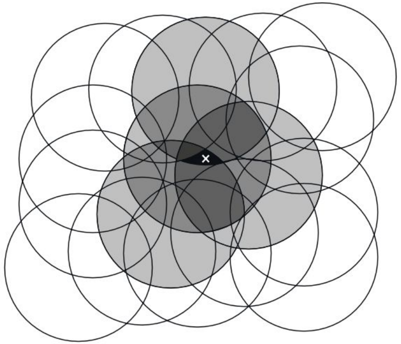
- Consider a task in which the natural representation of the state set is a continuous two- dimensional space.
- We define binary features indicating whether a state is present or not in a specific circle.



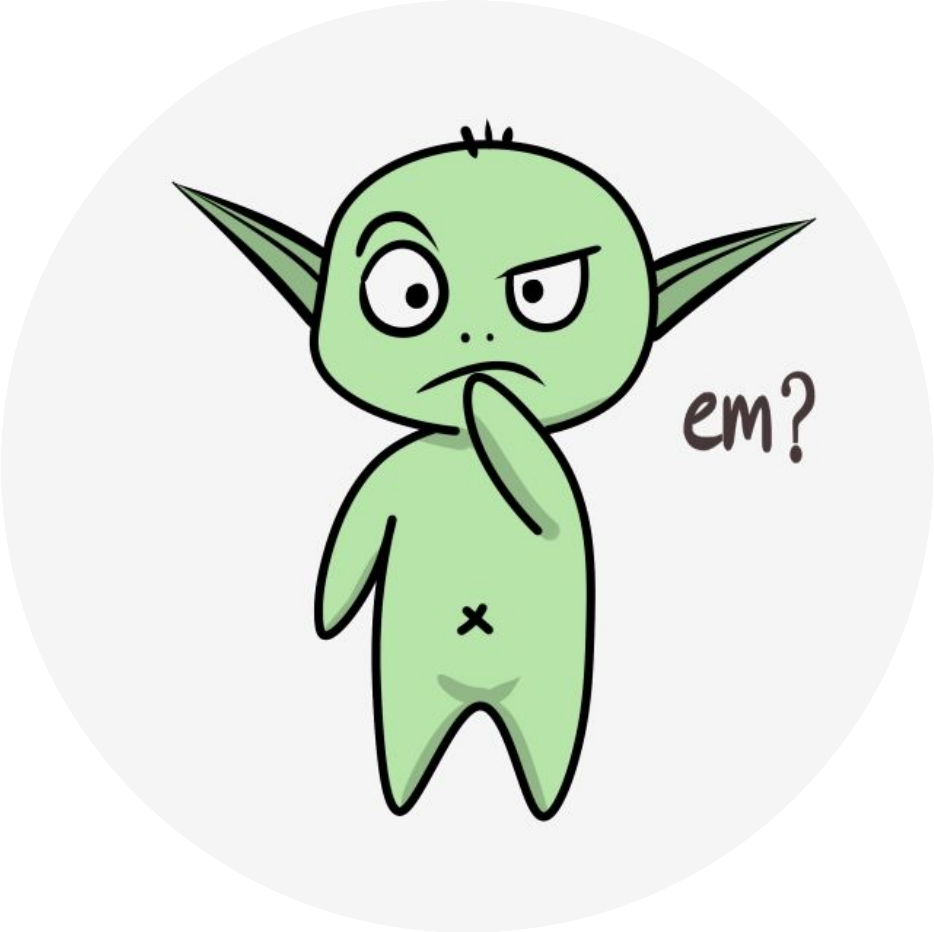
The shape defines generalization



Narrow generalization



Broad generalization



https://pngtree.com/freepng/question-expression-cartoon-illustration_4545209.ht