

# **CMPUT 267**

# **Machine Learning I**

## **Fall 2025**

**Instructor: Vlad Tkachuk**

Lectures will be streamed on  
Google Meet

# Lectures Will be Recorded and Made Public

- Lectures will be recorded
- Recordings will be posted publicly to this [YouTube Playlist](#)
- If you speak (ex: ask a question) in class your voice will be recorded
- If you speak on Google Meet your voice and Google profile picture (or video if you have it on) will be recorded

**Last 2 lectures (Dec 2, 4) will be fully virtual.**  
I will be away for a conference



# There is a course website

(vladtkachuk4.github.io/machinelearning1)

These slides (and future) are posted there in the schedule tab

Course is based on the  
course notes

They will be slightly updated throughout the term

# Course Details

- Lectures: Tue & Thu 12:30pm - 1:50pm (CCIS 1-440 & Virtual)
- Instructor: Vlad Tkachuk (email: [vtkachuk@ualberta.ca](mailto:vtkachuk@ualberta.ca))
- Office Hours: Thu 2:00pm - 3:00pm (UCOMM 3-480)

# TAs

**TA email: [cmput267@ualberta.ca](mailto:cmput267@ualberta.ca)**

Name	Day and Time	Location
Het Patel	Monday 9:00am - 10:00am	<a href="#">Virtual</a>
Kiarash Aghakasiri	Monday 2:00pm - 3:00pm	UCOMM 3-003
Alireza Kazemipour	Monday 3:00pm - 4:00pm	UCOMM 6-125
Ian Vyse	Tuesday 9:00am - 10:00am	<a href="#">Virtual</a>
Guoqing Luo	Tuesday 10:00am - 11:00am	<a href="#">Virtual</a>
Kushagra Chandak	Wednesday 9:00am - 10:00am	<a href="#">Virtual</a>
Kaining Yang	Wednesday 10:00am - 11:00am	<a href="#">Virtual</a>
Usaid Ahmed	Wednesday 12:00pm - 1:00pm	<a href="#">Virtual</a>
Layne Pitman	Thursday 11:00am - 12:00pm	TBD
Vlad Tkachuk (Instructor)	Thursday 2:00pm - 3:00pm	UCOMM 3-480
Avani Tiwari	Friday 9:00am - 10:00am	<a href="#">Virtual</a>
Sara Shehada	Friday 11:00am - 12:00pm	<a href="#">Virtual</a>
Saksham Anand	Friday 1:00pm - 2:00pm	<a href="#">Virtual</a>

# Asking Questions and Getting Help

1. **Ask an LLM** (ex: ChatGPT). Fast responses and familiarizes you with LLMs.
  - **IMPORTANT:** LLM outputs should not be blindly trusted; students must verify information if unsure of its accuracy.
2. **Ask on Piazza** (Note: you can ask questions anonymously)
  - Any questions that don't reveal assignment solutions
3. **Email the TAs** (cmput267@ualberta.ca)
  - For private assignment questions
4. **Email the instructor** (vtkachuk@ualberta.ca)
  - Missing exams or personal issues

Join Piazza  
(Link also on the course website)

# Grading

Assessment	Weight	Date
Assignments (8, top 7 counted):	28% (4% each)	See the <a href="#">schedule tab on the course website</a>
Midterm exam 1:	21%	Oct 7, 2025 in class (12:35pm - 1:45pm in CCIS 1-440)
Midterm exam 2:	21%	Nov 18, 2025 in class (12:35pm - 1:45pm in CCIS 1-440)
Final exam	30%	Dec 18, 2024 (8:30am), <a href="#">date and time are tentative</a>

- At least 3 of the assignments will be coding assignments. We will be using Python in [Google Colab](#).
- To do the assignments you will need: An internet connection, and a modern web browser (Chrome, Firefox, or Safari recommended).

# Course Policies

- We will not accept late assignments
- If you are granted an excused absence for a midterm exam its weight will be transferred to the final exam
- All assignments must be done by you
- You can use AI (ex: LLMs) to **help you** with assignments
- No cheating, plagiarism, harassment, physical assault, etc.
  - Can result in suspension or expulsion from the University
  - Familiarize yourself with the new Student Academic Integrity Policy



Refer to the syllabus for  
detailed official course policies

**Advice:** If you can do the assignment questions and examples in the course notes, then you are likely to succeed on the midterms and final exam

**Disclaimer: This course is math heavy,  
and we do not cover “Deep Learning”**

**Disclaimer:** This course is math heavy,  
and we do not cover “Deep Learning”

**However:** Course notes are mostly self contained and  
I will try to motivate things as much as possible

**Disclaimer:** This course is math heavy,  
and we do not cover “Deep Learning”

**However:** Course notes are mostly self contained and  
I will try to motivate things as much as possible

**Necessary background:** Can take derivatives (Calculus)

**Useful background:** Probability and statistics, familiar with vectors and matrices (Lin Alg)

Please ask questions!

Especially: “Why are we doing this?”

# Course Outline

1. Math and probability review
2. Define supervised learning formally (splitting it into regression or classification)
3. Design some learning programs to solve regression problems

## **Midterm Exam 1**

4. Evaluate our learning programs
5. Present some new ways to design learning programs for regression

## **Midterm Exam 2**

6. Repeat the above for classification problems
7. Brief intro to neural networks and language models

## **Final Exam (Cumulative)**

Raise your hand if you're in  
1st year



Raise your hand if you're in  
2nd year

Raise your hand if you're in  
3rd year

Raise your hand if you're in  
4th year

Raise your hand if you're a  
Grad student

# What is Machine Learning?

# What is Machine Learning?

Remaining slides are inspired by: Shai Ben-David (Lecture 1 - CS 485/685)

Raise your hand if you think you  
know what Machine Learning is

Raise your hand if you learned  
about machine learning before  
(ex: taken a course, watched videos, etc.)



Raise your hand if you've  
heard of:



Raise your hand if you use:



Raise your hand if you think  
Machine Learning is exciting

# What is Machine Learning?

# **What is Machine Learning?**

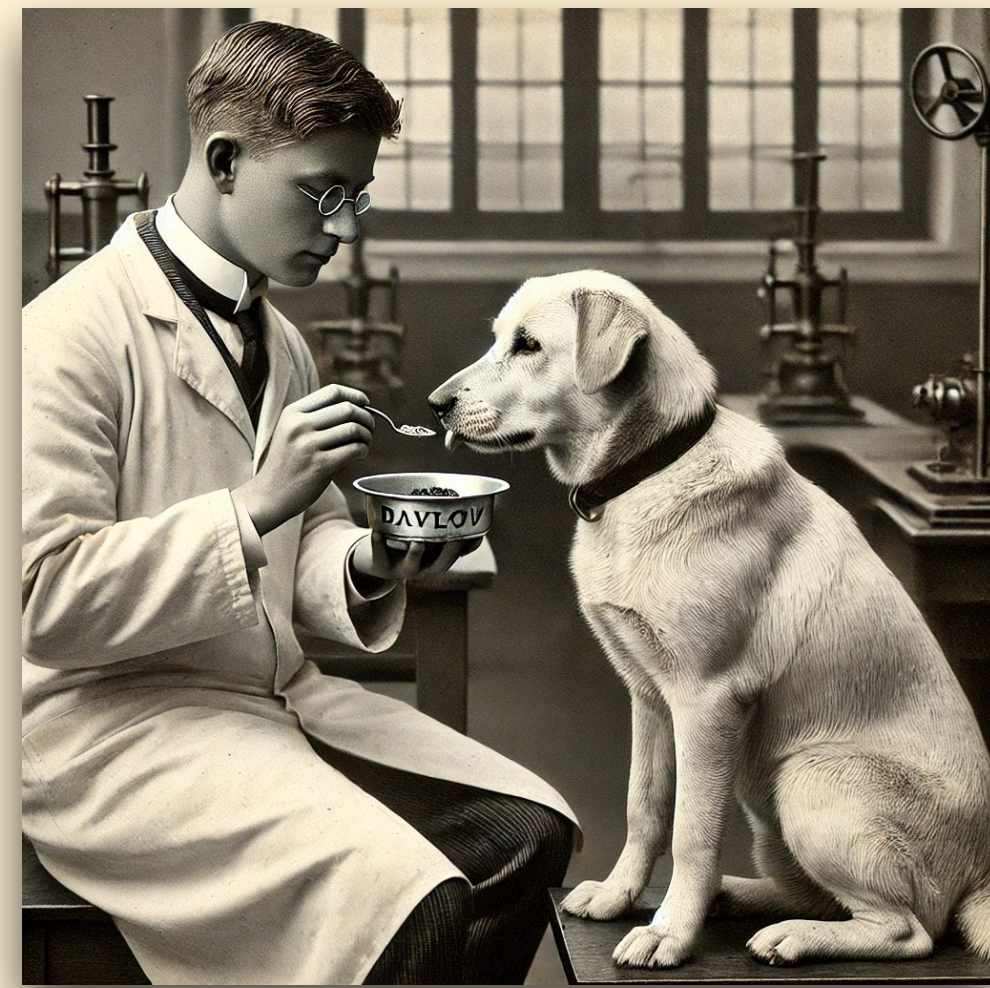
**Answer: A program/algorithm  
that is learning**

# What is Learning?





# Example: Pavlov's Dog



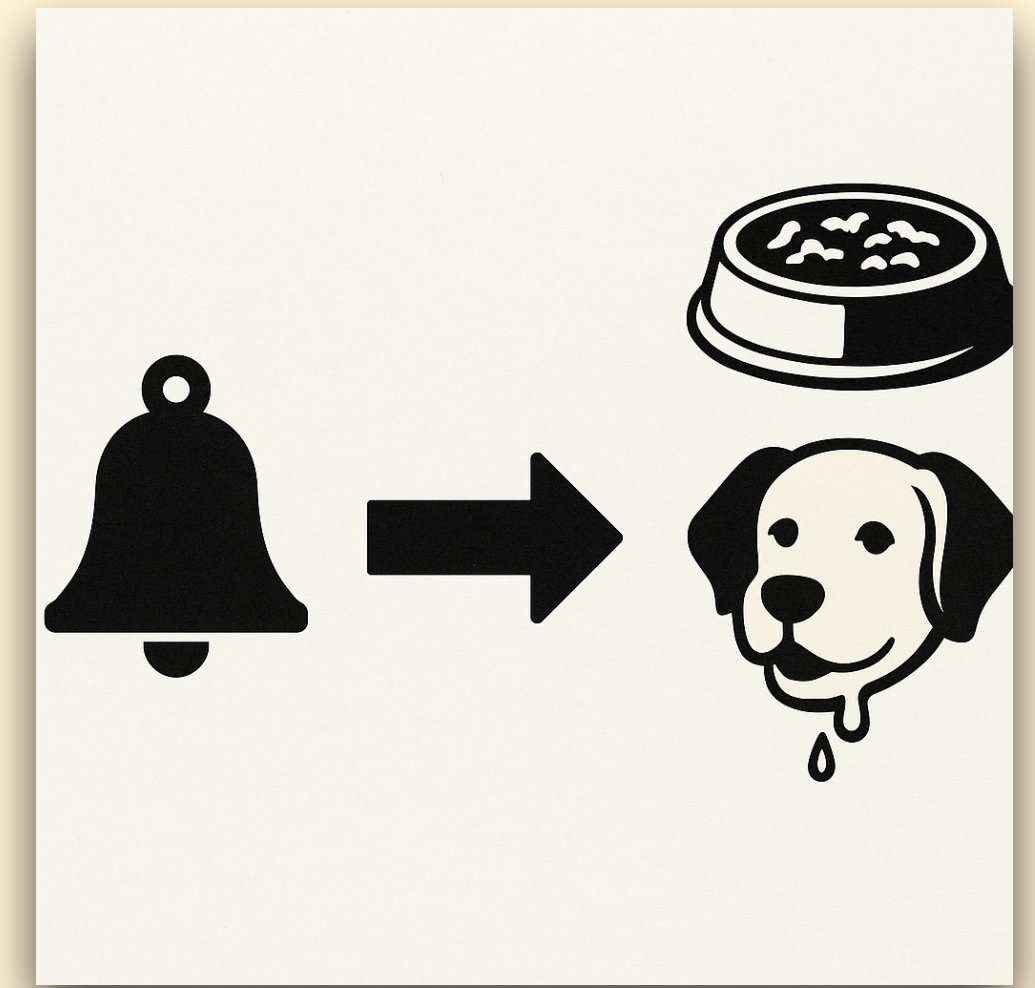
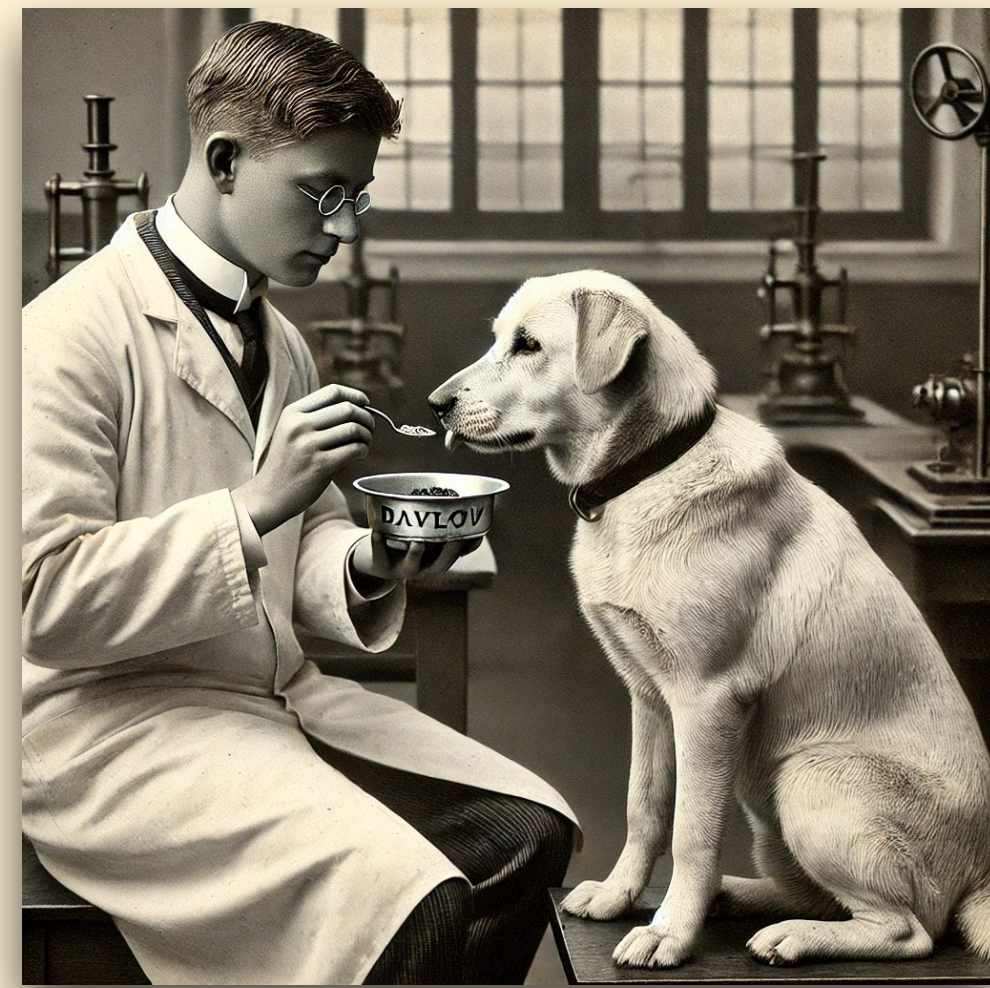
Experience

Learning

Knowledge



# Example: Pavlov's Dog



Experience

Learning

Knowledge



# Example: Learning to Play Tennis

Experience

Learning

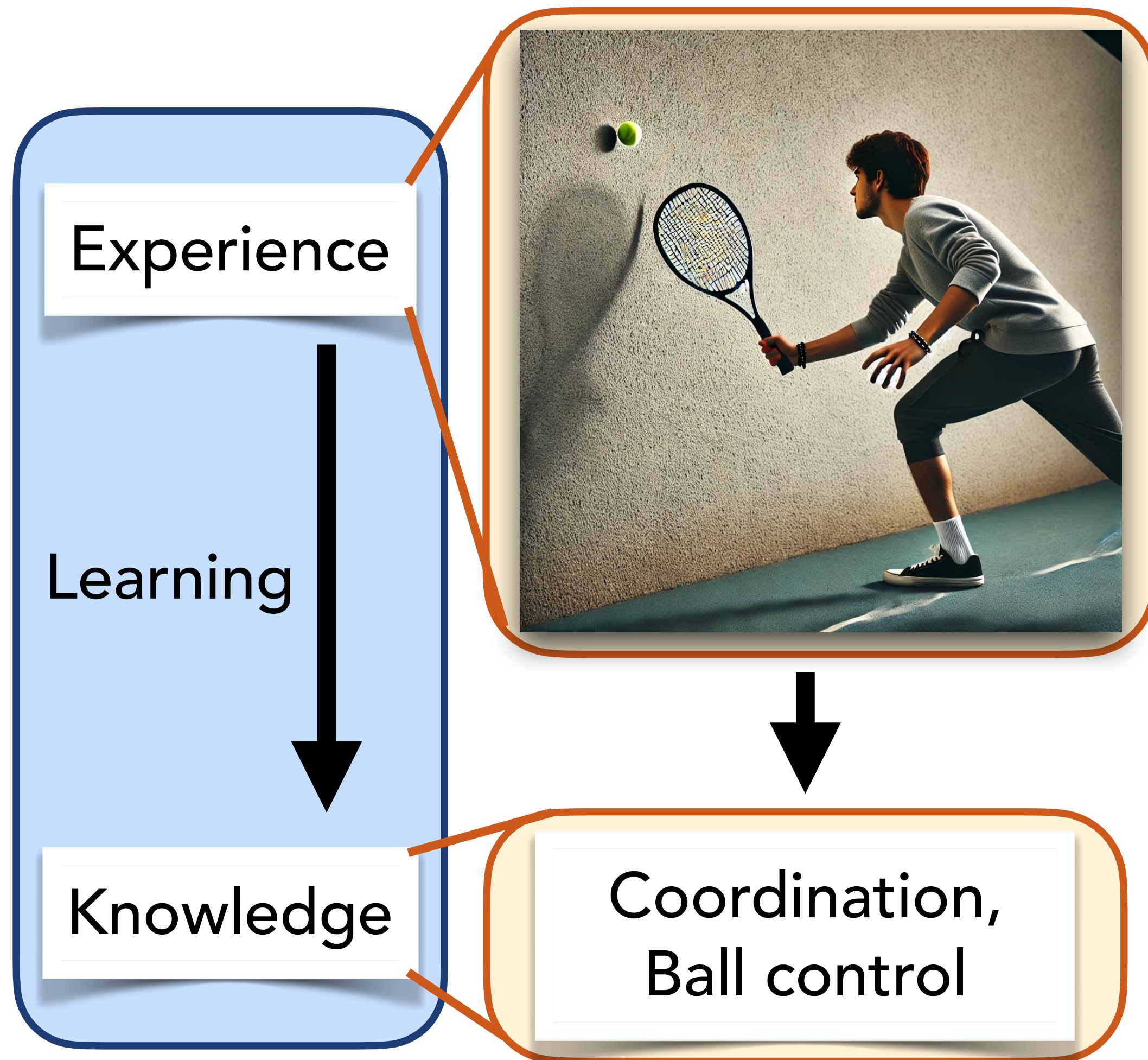


Knowledge





# Example: Learning to Play Tennis





# Example: Learning to Play Tennis

Unsupervised Learning

Experience

Learning

Knowledge



Coordination,  
Ball control



# Example: Learning to Play Tennis

Unsupervised Learning

Experience

Learning

Knowledge



Coordination,  
Ball control



# Example: Learning to Play Tennis

Unsupervised Learning

Experience

Learning

Knowledge



Coordination,  
Ball control



How to play



# Example: Learning to Play Tennis

Unsupervised Learning



Coordination,  
Ball control

Supervised Learning



How to play

Experience

Learning

Knowledge



# Example: Learning to Play Tennis

Unsupervised Learning

Supervised Learning

Experience

Learning

Knowledge



Coordination,  
Ball control

How to play



# Example: Learning to Play Tennis

Unsupervised Learning



Coordination,  
Ball control

Supervised Learning



How to play



How to play

Experience

Learning

Knowledge



# Example: Learning to Play Tennis

Unsupervised Learning



Coordination,  
Ball control

Supervised Learning



How to play

Reinforcement Learning



How to play

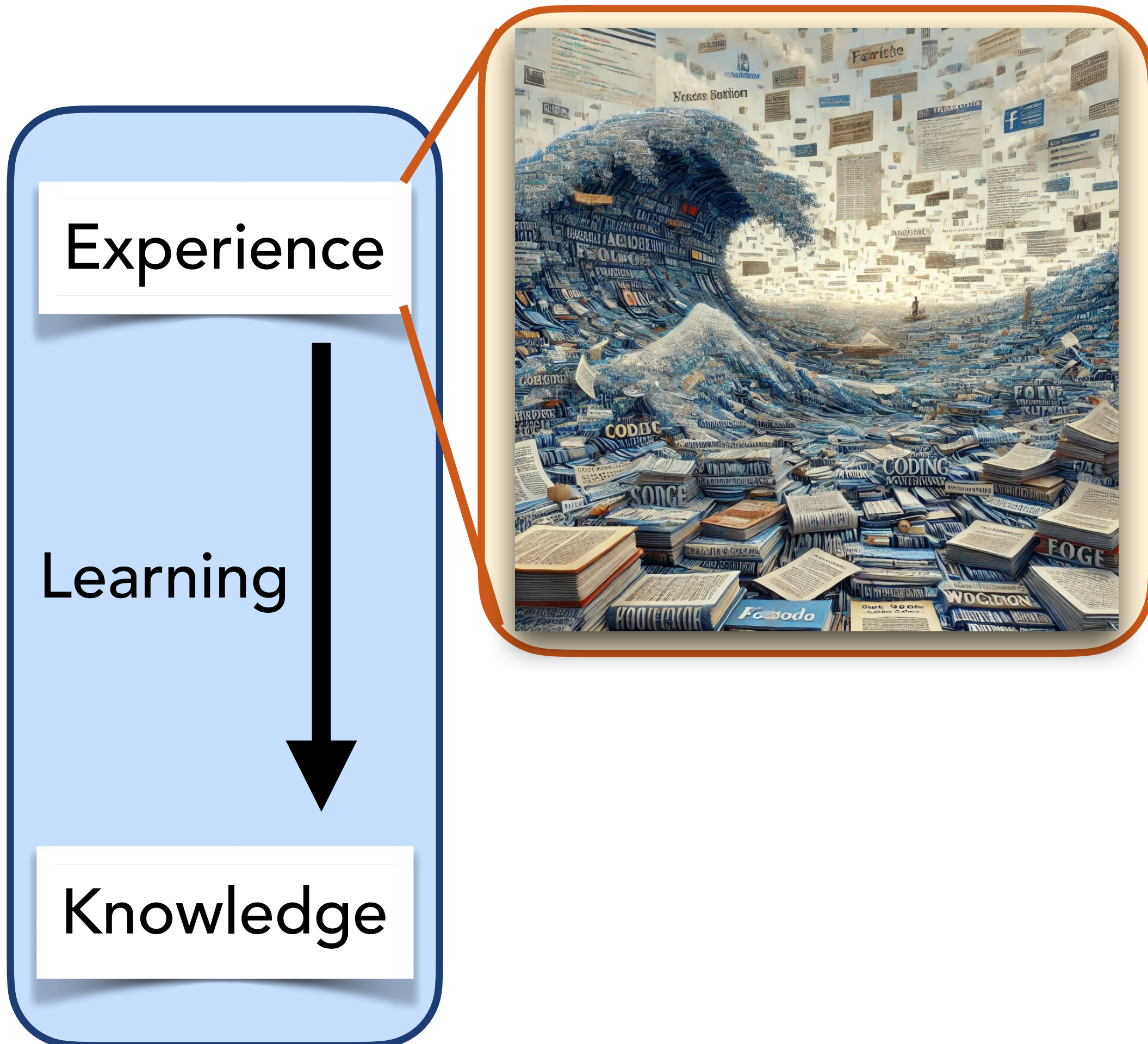
Experience

Learning

Knowledge

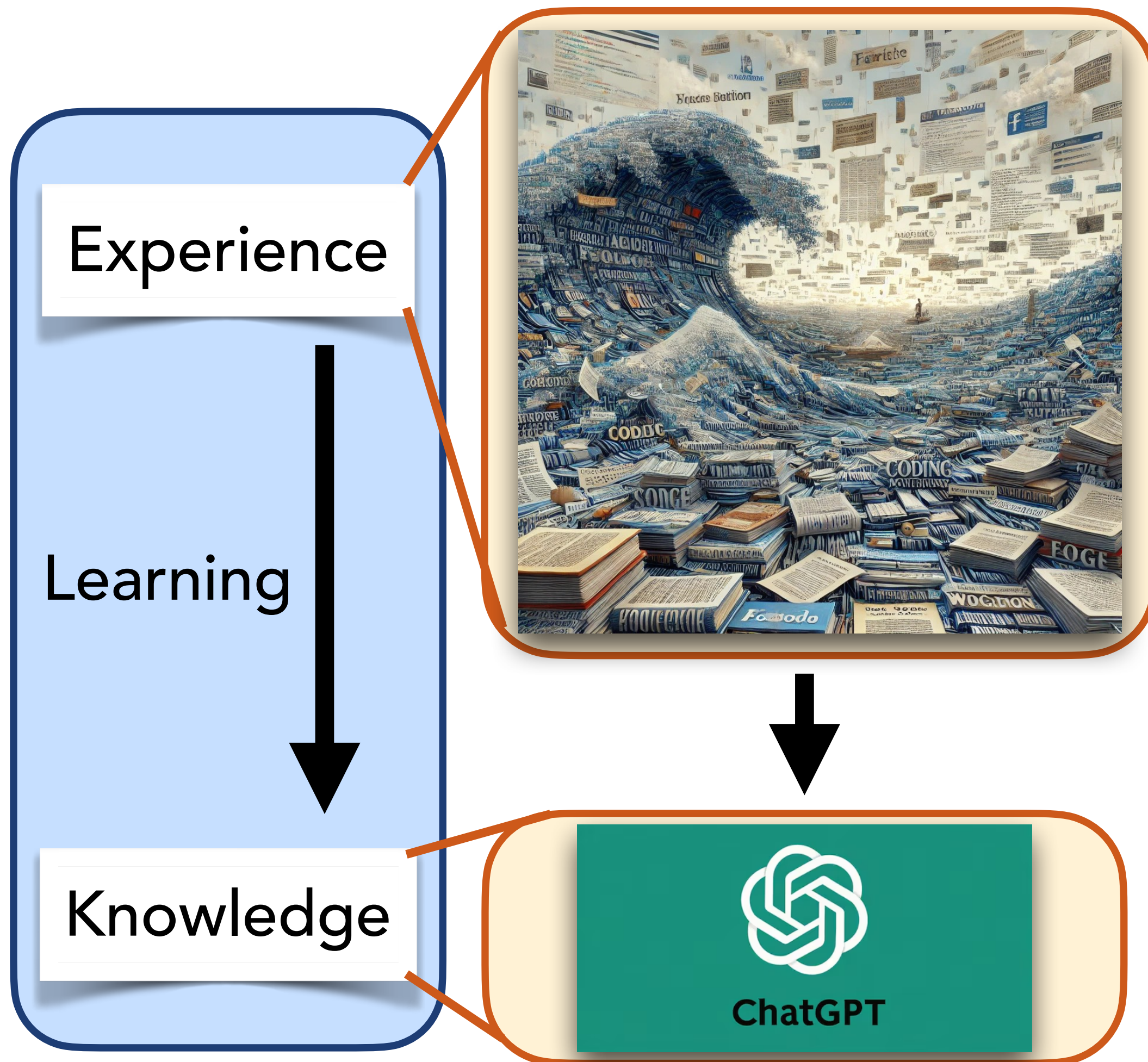


# Example: Learning Language





# Example: Learning Language





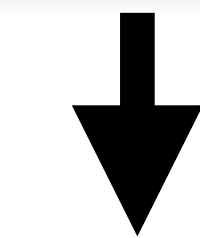
# Example: Learning Language

Offline (Batch) Learning

Experience

Learning

Knowledge



ChatGPT



# Example: Learning Language

Offline (Batch) Learning

Experience

Learning

Knowledge

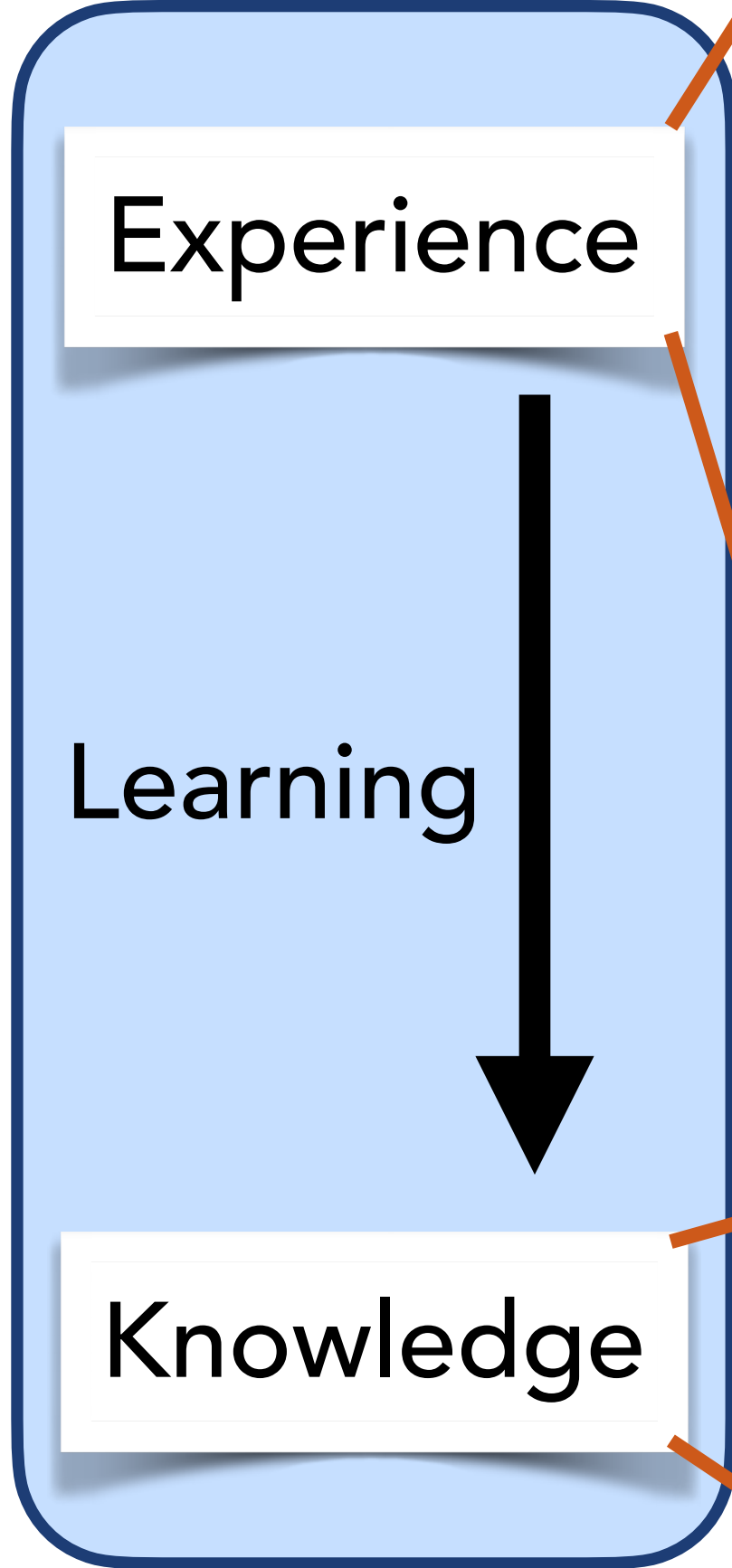


ChatGPT



# Example: Learning Language

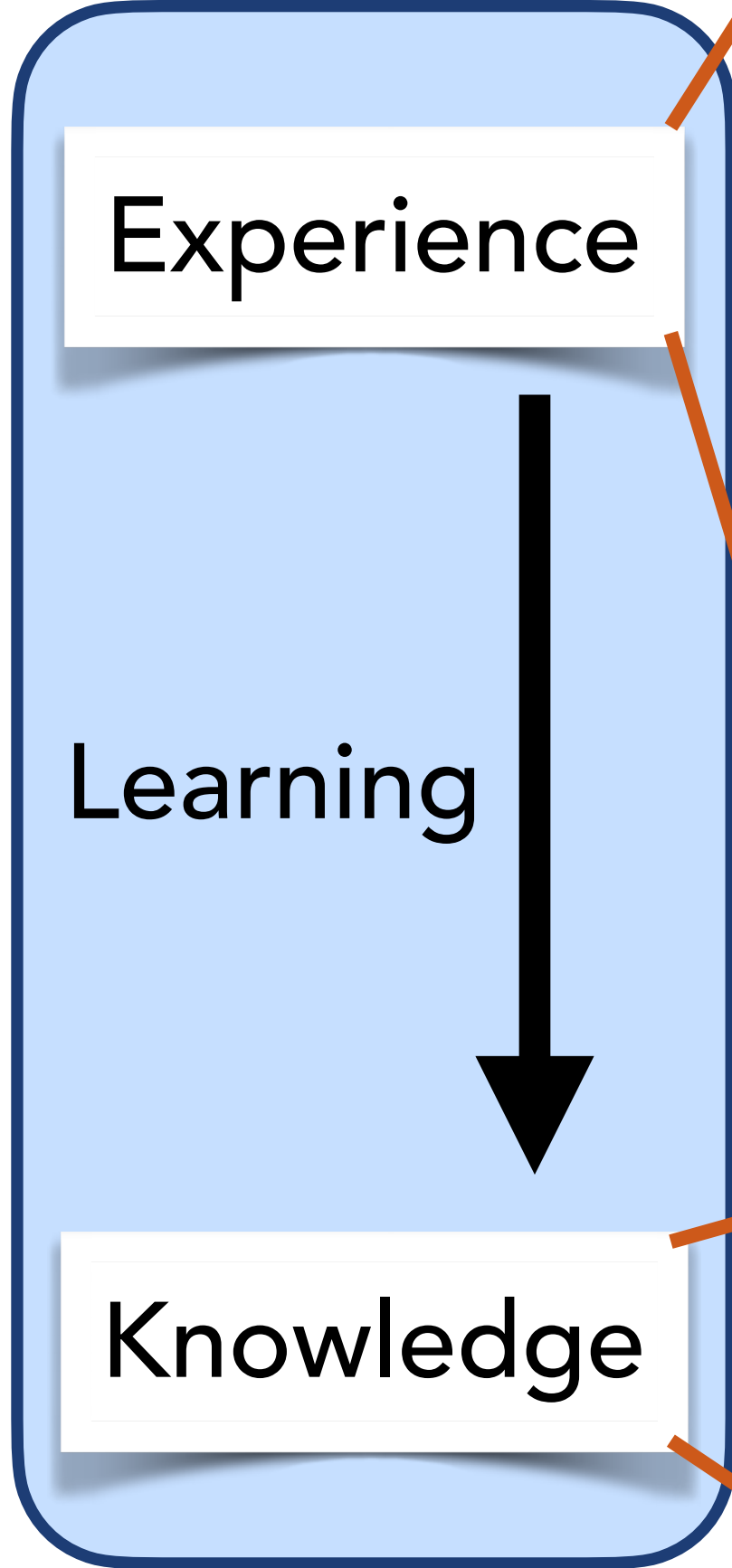
Offline (Batch) Learning





# Example: Learning Language

Offline (Batch) Learning





# Example: Learning Language

Offline (Batch) Learning

Online Learning

Experience

Learning

Knowledge





# Example: Games



Experience

Learning



Knowledge



# Example: Games

Experience

Learning



Knowledge





# Example: Games

Stochastic Learning

Experience

Learning



Knowledge





# Example: Games

Stochastic Learning

Experience

Learning



Knowledge





# Example: Games

Stochastic Learning

Experience



Learning



Knowledge





# Example: Games

Stochastic Learning



Adversarial Learning



Experience

Learning

Knowledge





# Different Kinds of Learning

Unsupervised Learning

Supervised Learning

Reinforcement Learning

Offline (Batch) Learning

Online Learning

Stochastic Learning

Adversarial Learning



# What we Will Cover

Unsupervised Learning

Supervised Learning

Reinforcement Learning

Offline (Batch) Learning

Online Learning

Stochastic Learning

Adversarial Learning

Supervised, Offline, Stochastic Learning

# What we Will Cover

Unsupervised Learning

Supervised Learning

Reinforcement Learning

Offline (Batch) Learning

Online Learning

Stochastic Learning

Adversarial Learning

**Supervised Learning = Supervised, Offline, Stochastic Learning**

# What we Will Cover

Unsupervised Learning

Supervised Learning

Reinforcement Learning

Offline (Batch) Learning

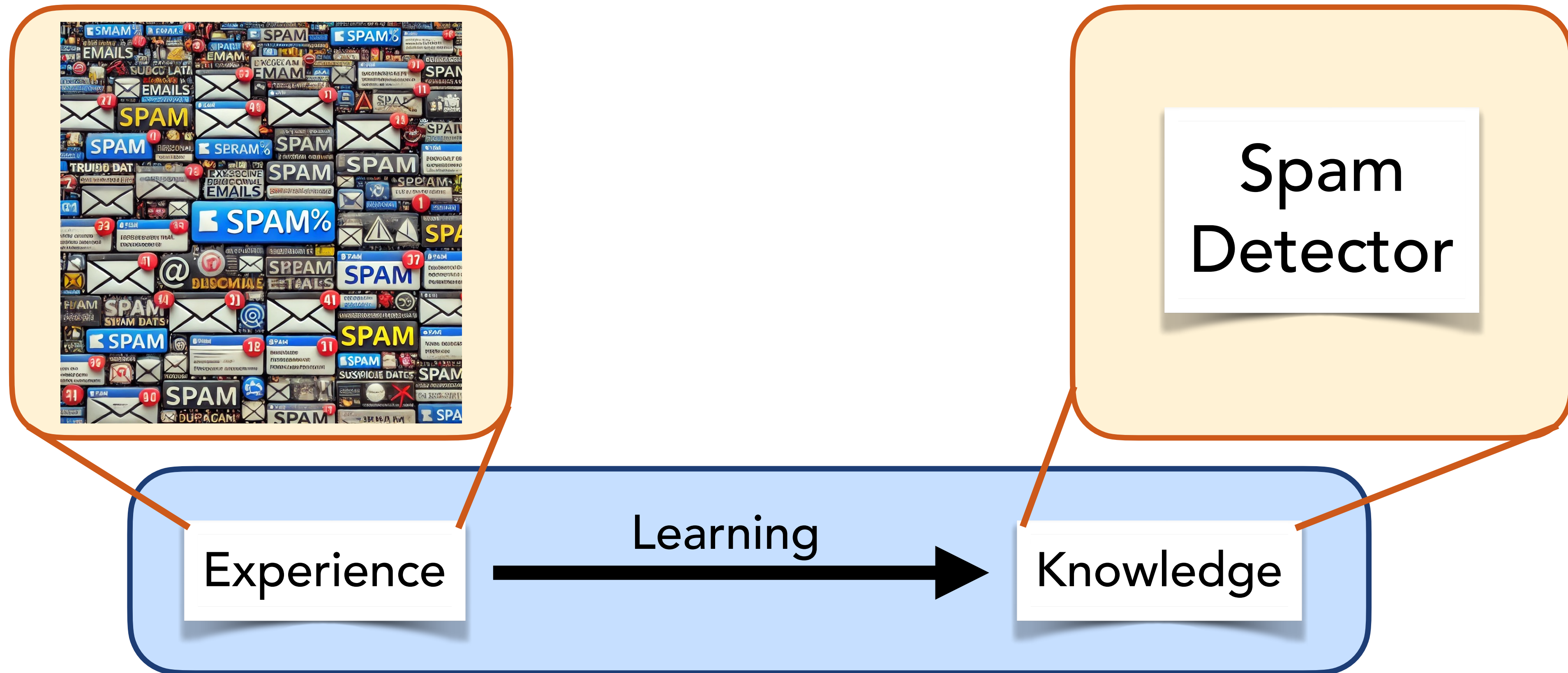
Online Learning

Stochastic Learning

Adversarial Learning

**Supervised Learning** = Learning from a **batch** of **labeled** **randomly selected** experience

# Example: Spam Detector



**Supervised Learning** = Learning from a **batch** of **labeled randomly selected** experience

**Why are machines  
(programs) that learn useful?  
(instead of just having humans)**

# Why are Programs Useful?

**Programs can perform computations much more efficiently than humans**



# Why are Programs Useful?

Programs can perform computations much more efficiently than humans

*Examples:*

- A calculator can do math faster than humans

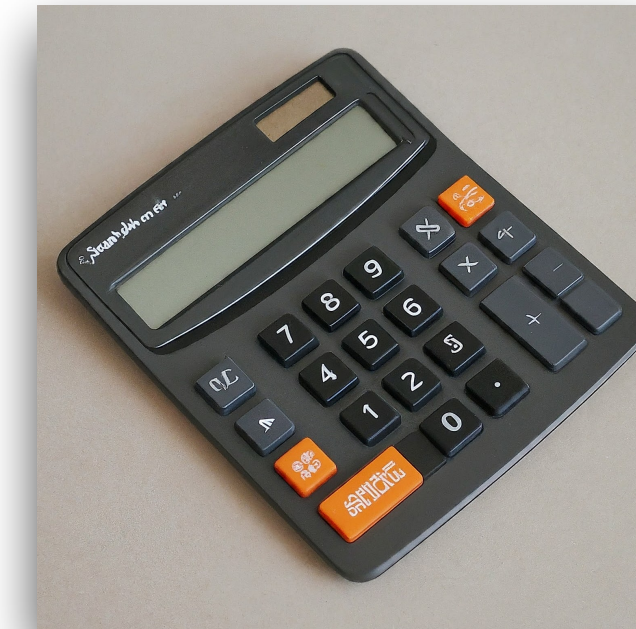


# Why are Programs Useful?

Programs can perform computations much more efficiently than humans

*Examples:*

- A calculator can do math faster than humans
- Excel can plot some data faster than a human



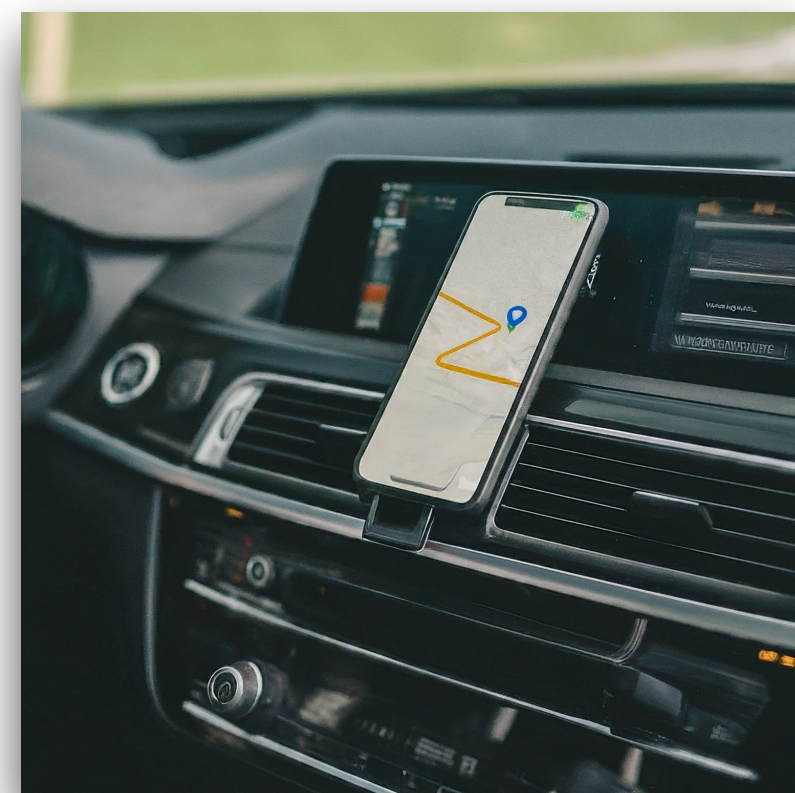
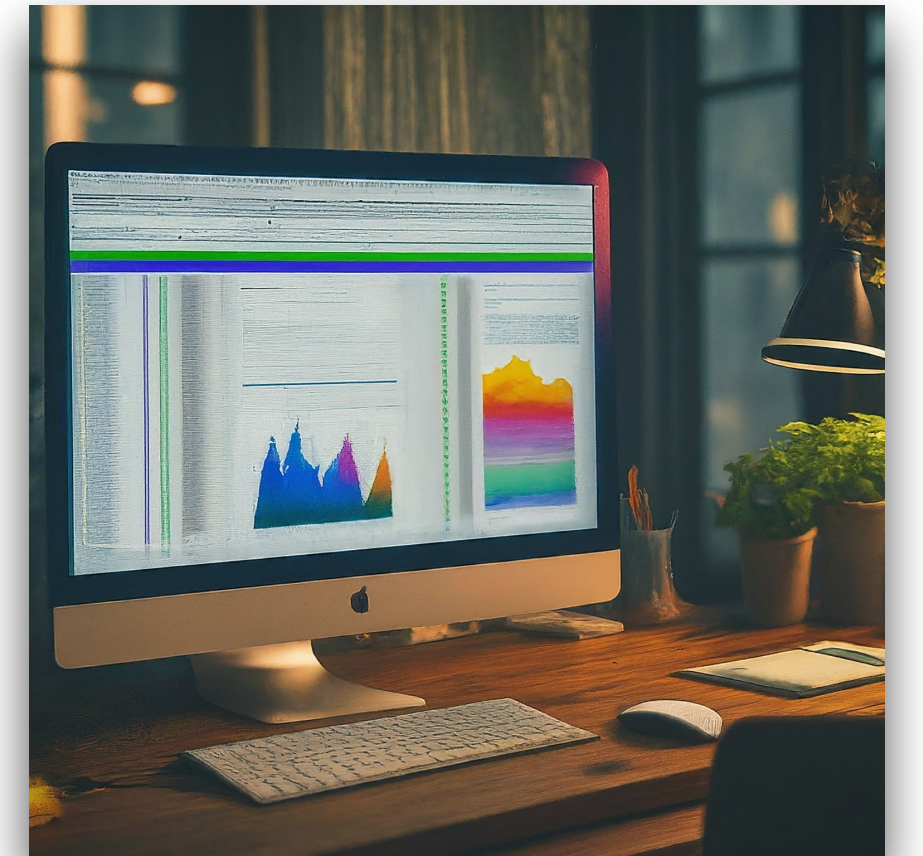


# Why are Programs Useful?

Programs can perform computations much more efficiently than humans

*Examples:*

- A calculator can do math faster than humans
- Excel can plot some data faster than a human
- Google Maps can plan a driving route faster than a human



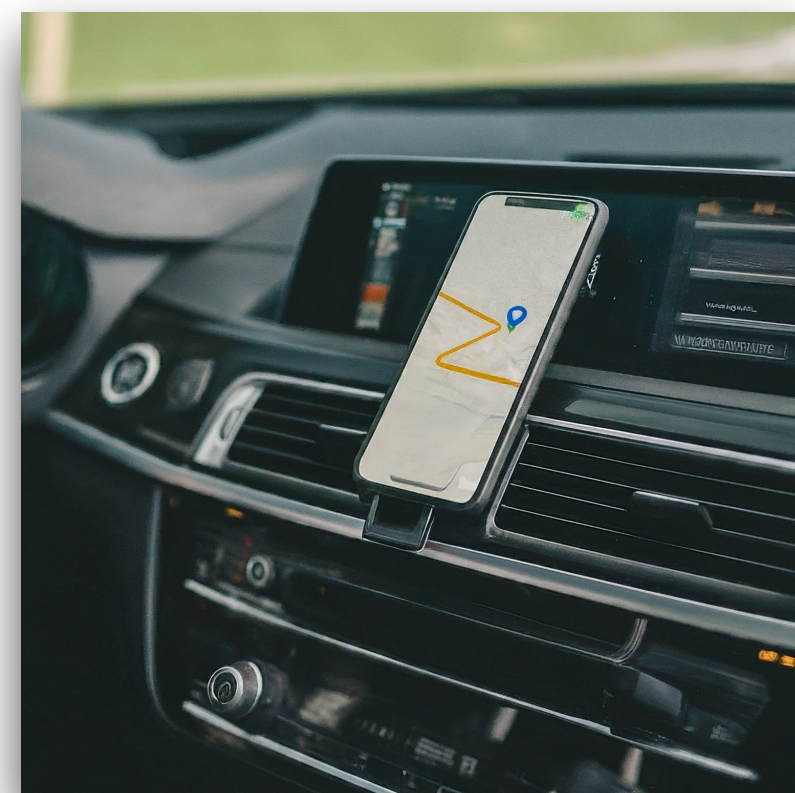
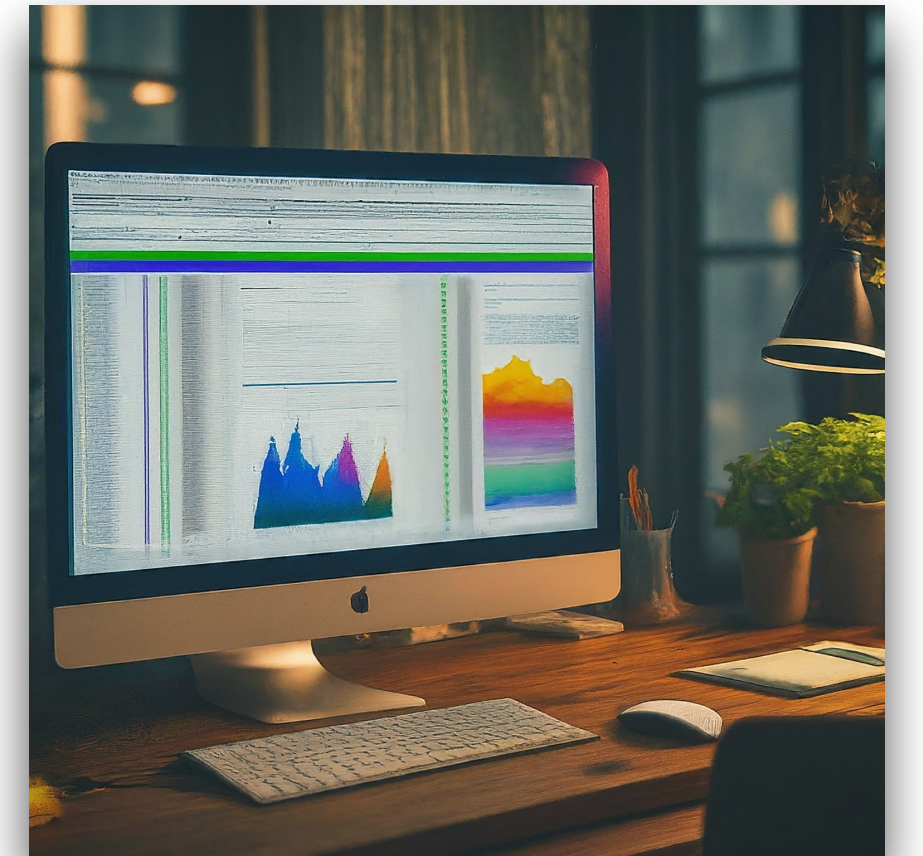
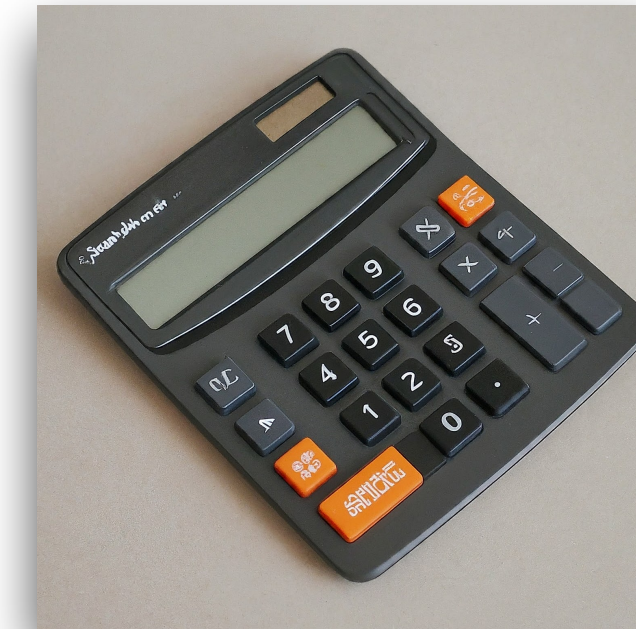


# Why are Programs Useful?

Programs can perform computations much more efficiently than humans

*Examples:*

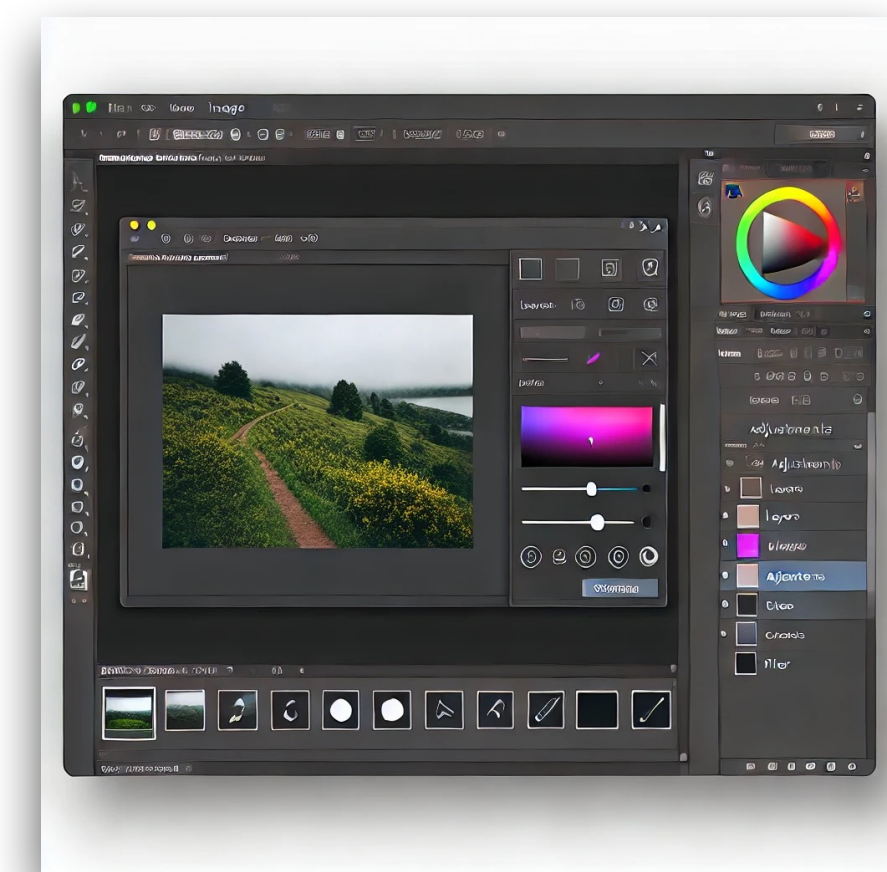
- A calculator can do math faster than humans
- Excel can plot some data faster than a human
- Google Maps can plan a driving route faster than a human
- Google Docs can count the number of words in a document faster than a human





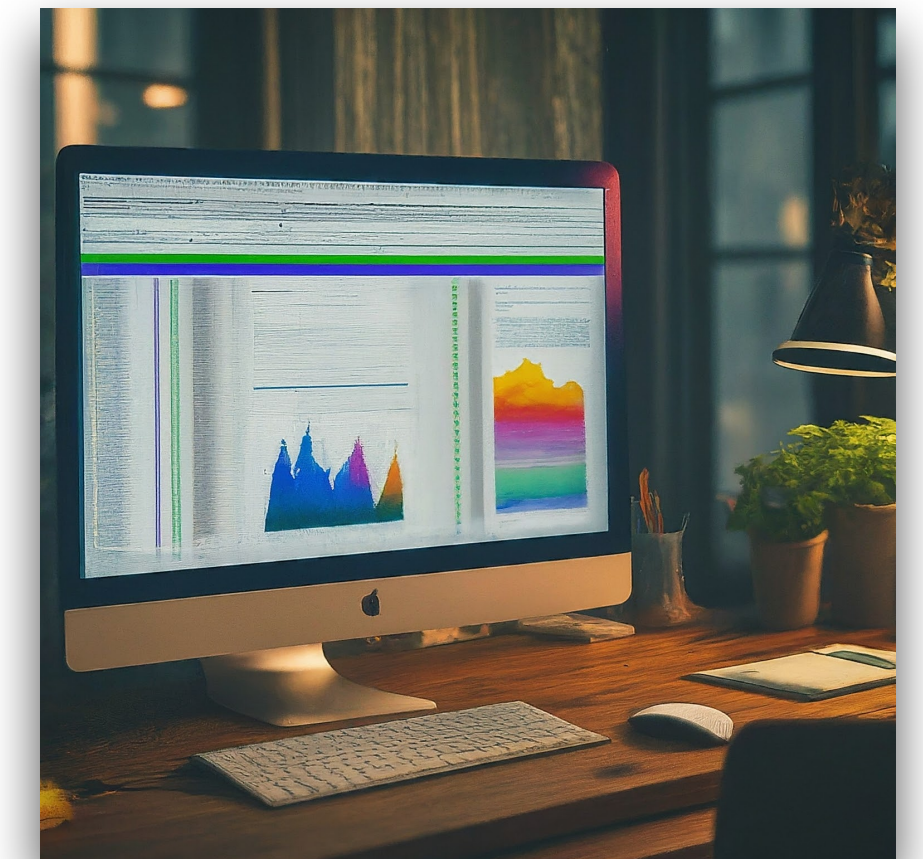
# Why are Programs Useful?

Programs can perform computations much more efficiently than humans



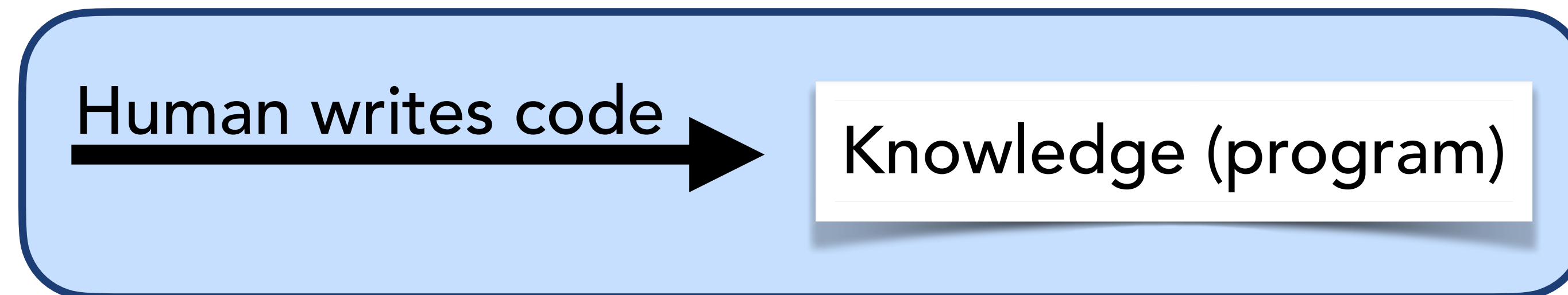
*Examples:*

- A calculator can do math faster than humans
- Excel can plot some data faster than a human
- Google Maps can plan a driving route faster than a human
- Google Docs can count the number of words in a document faster than a human
- etc.





# Classic Programs

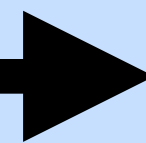


# Classic Programs

*Example: A human writes a calculator program.*

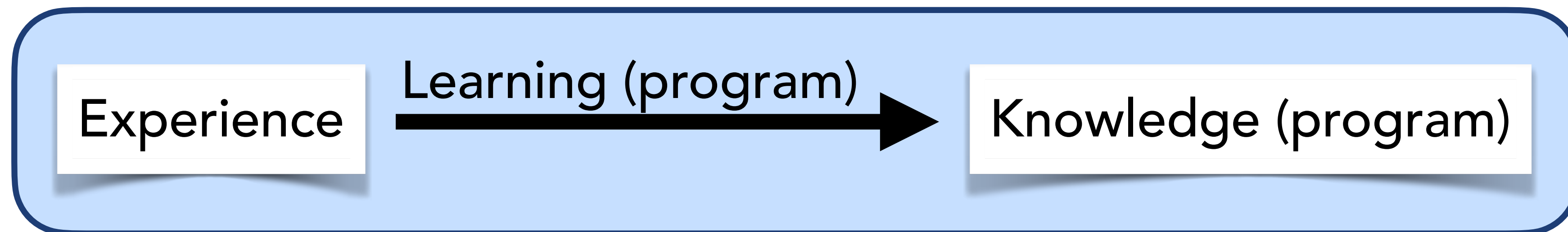


Human writes code



Knowledge (program)

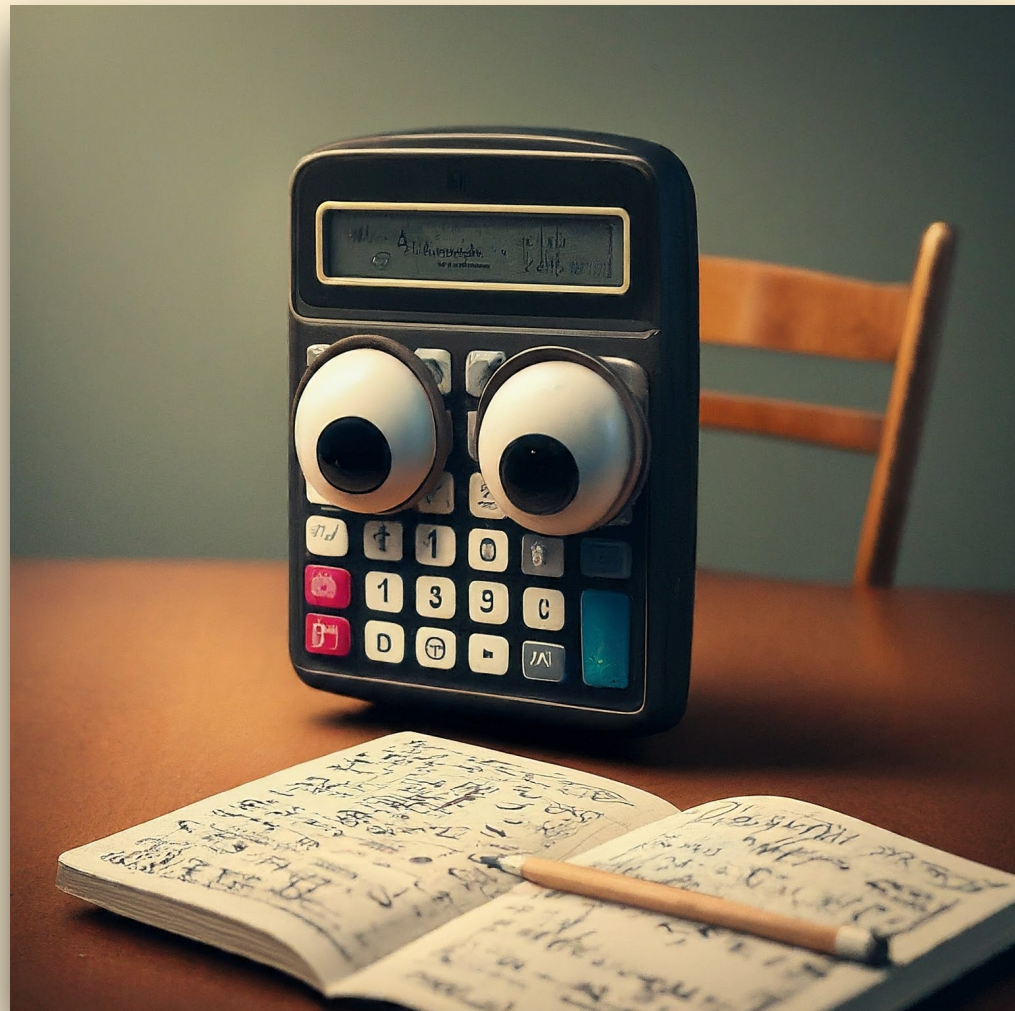
# Programs that Learn





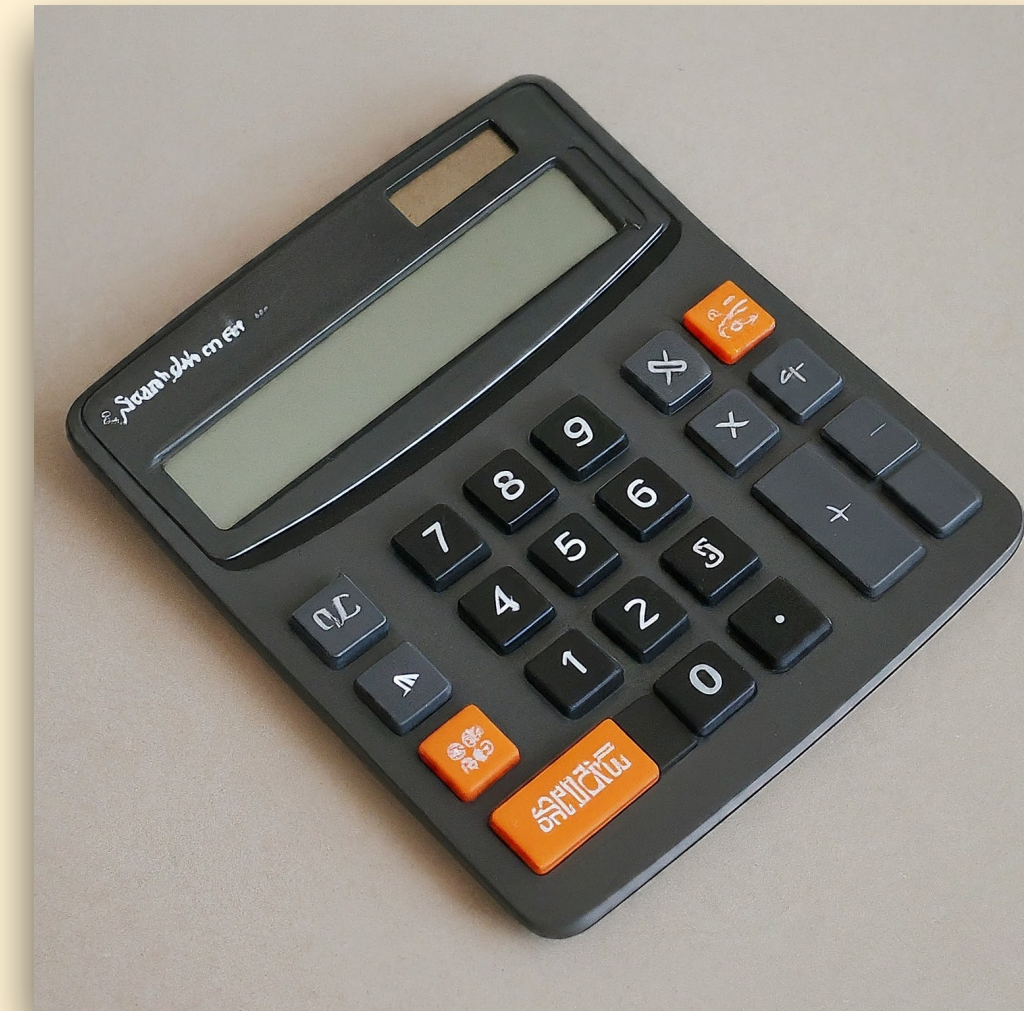
# Programs that Learn

*Example: A calculator learns addition by seeing examples of numbers being added together.*



Experience

Learning (program) →



Knowledge (program)

# Why are Programs that Learn Useful?

1. We don't know how to write the code for certain types of knowledge



# Why are Programs that Learn Useful?

1. We don't know how to write the code for certain types of knowledge

*Examples:*

- Creating an image of something

# Why are Programs that Learn Useful?

1. We don't know how to write the code for certain types of knowledge

*Examples:*

- Creating an image of something  
"Generate an image of a cat"

# Why are Programs that Learn Useful?

1. We don't know how to write the code for certain types of knowledge

*Examples:*

- Creating an image of something
  - “Generate an image of a person that can not explain the steps to draw a cat”



# Why are Programs that Learn Useful?

1. We don't know how to write the code for certain types of knowledge

*Examples:*

- Creating an image of something  
“Generate an image of a person that can not explain the steps to draw a cat”





# Why are Programs that Learn Useful?

1. We don't know how to write the code for certain types of knowledge

*Examples:*

- Creating an image of something  
"Generate an image of a person that can not explain the steps to draw a cat"

**All of the images in this presentation were generated by**



ChatGPT





# What Learn Useful?

1.



- Object detection: stop sign, pedestrian, red light, green light, etc.





# Why are Programs that Learn Useful?

*More examples:*

- Chatbot (LLMs: ChatGPT, Claude, Gemini, etc.)





# Why are Programs that Learn Useful?

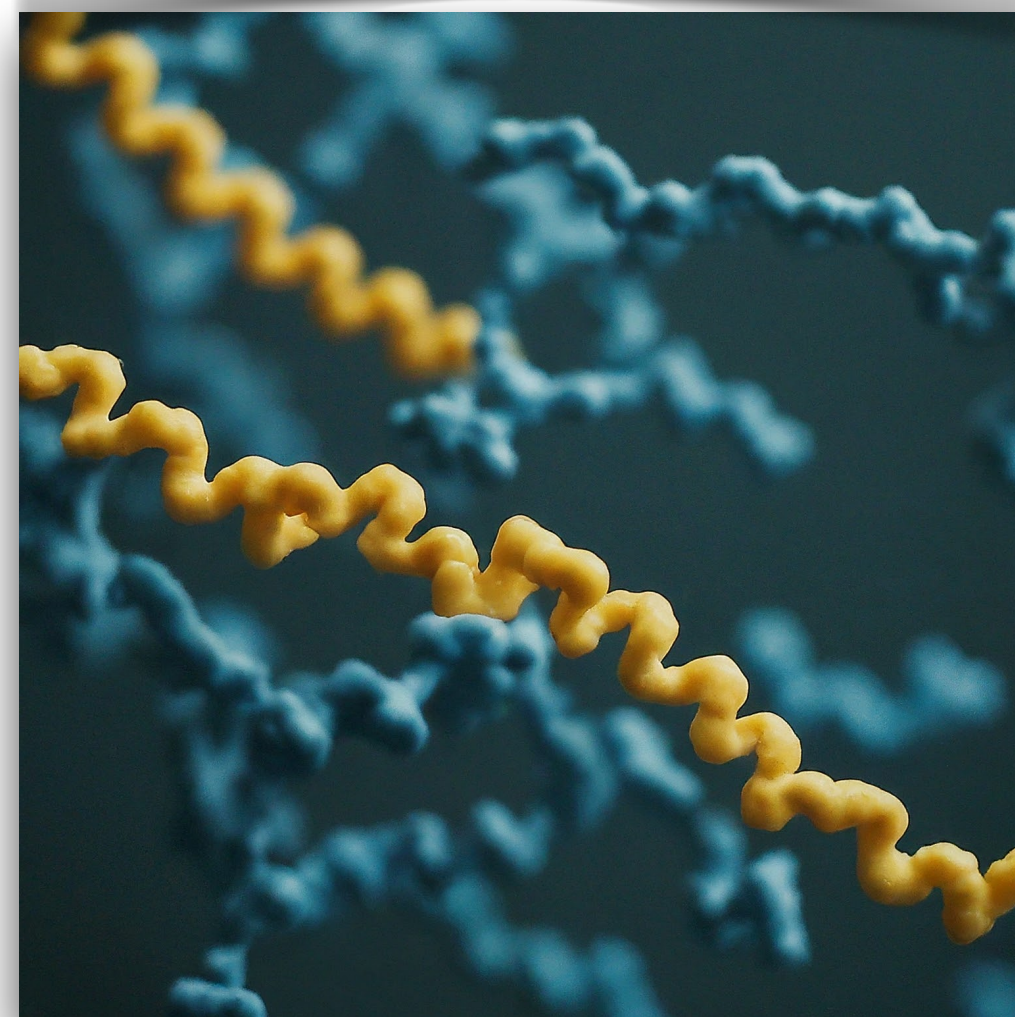
*More examples:*

- Chatbot (LLMs: ChatGPT, Claude, Gemini, etc.)
- Discovery: Predicting protein folding (Deepmind's AlphaFold).

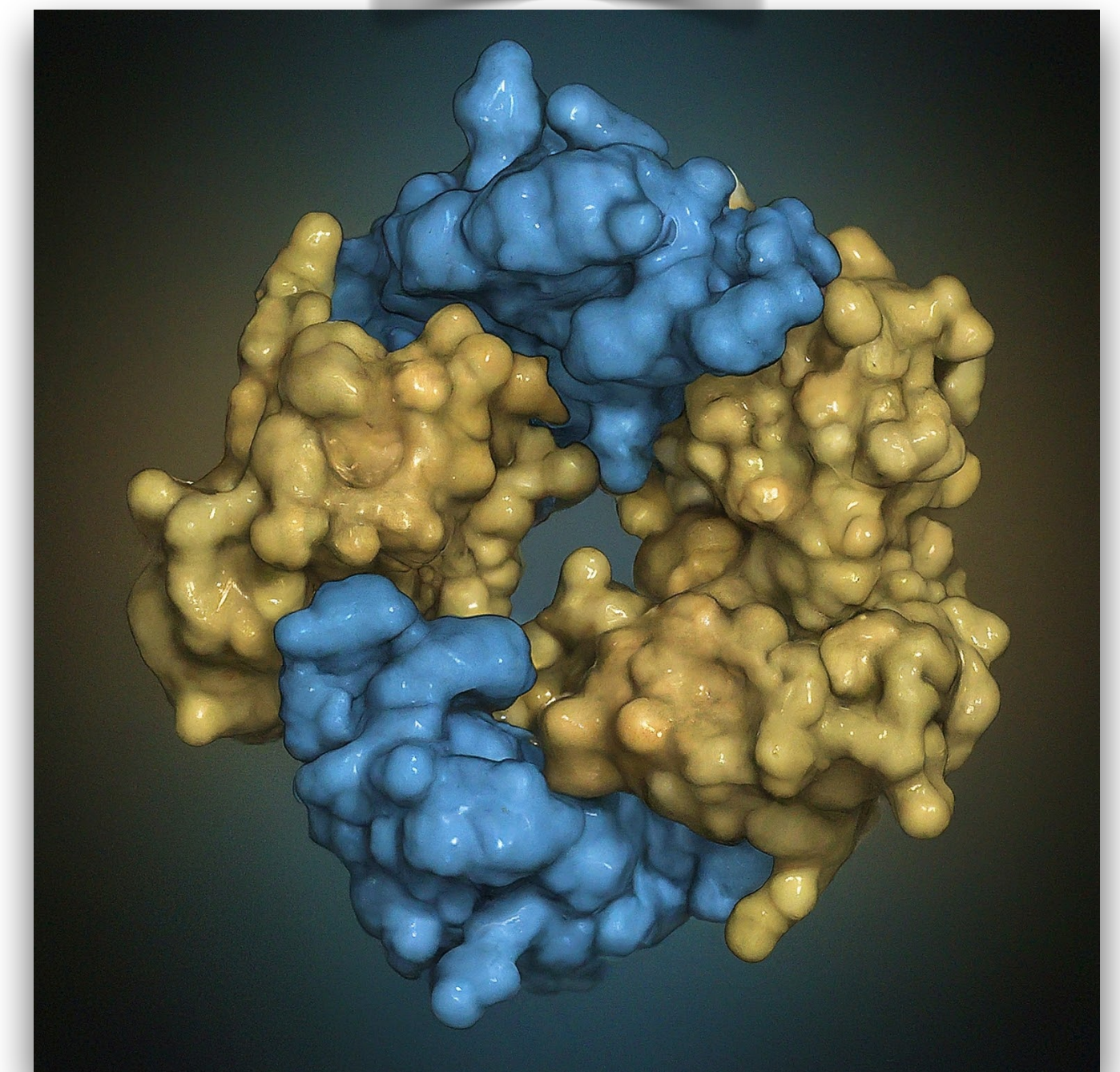


Protein

Amino acid chain



Folding





# Why are Programs that Learn Useful?

2. Can adapt to changing environments

# Why are Programs that Learn Useful?

## 2. Can adapt to changing environments

*Example:*

- Object detection, but at night time

Day time





# Why are Programs that Learn Useful?

## 2. Can adapt to changing environments

*Example:*

- Object detection, but at night time

Day time



Night time





**What will you learn in this  
course?**



# What we Will Cover

Unsupervised Learning

Supervised Learning

Reinforcement Learning

Offline (Batch) Learning

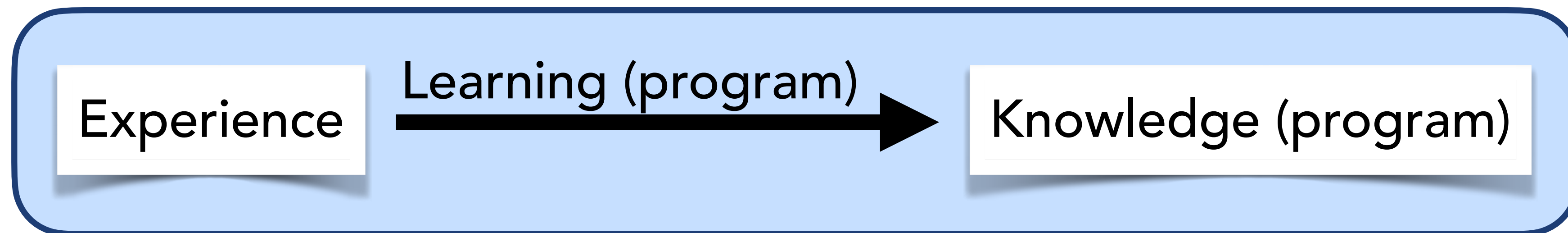
Online Learning

Stochastic Learning

Adversarial Learning

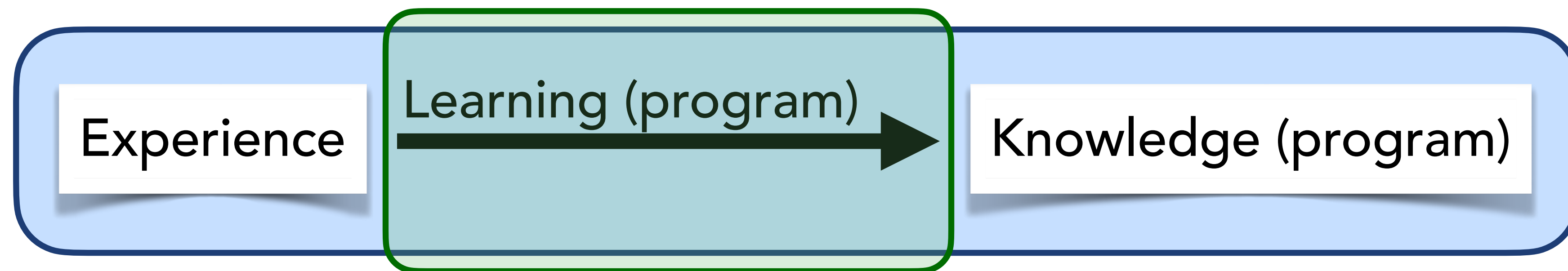
**Supervised Learning** = Learning from a **batch** of **labeled randomly selected** experience

# You will learn to write a program that learns

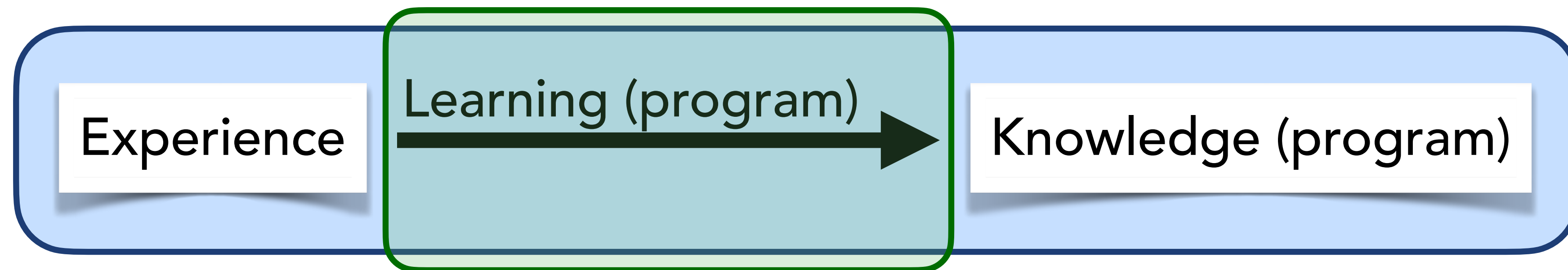
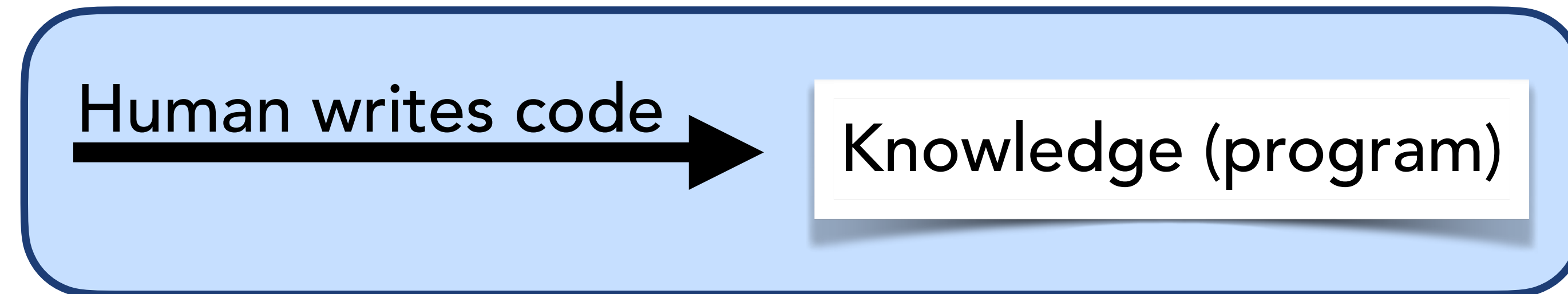




# You will learn to write a program that learns

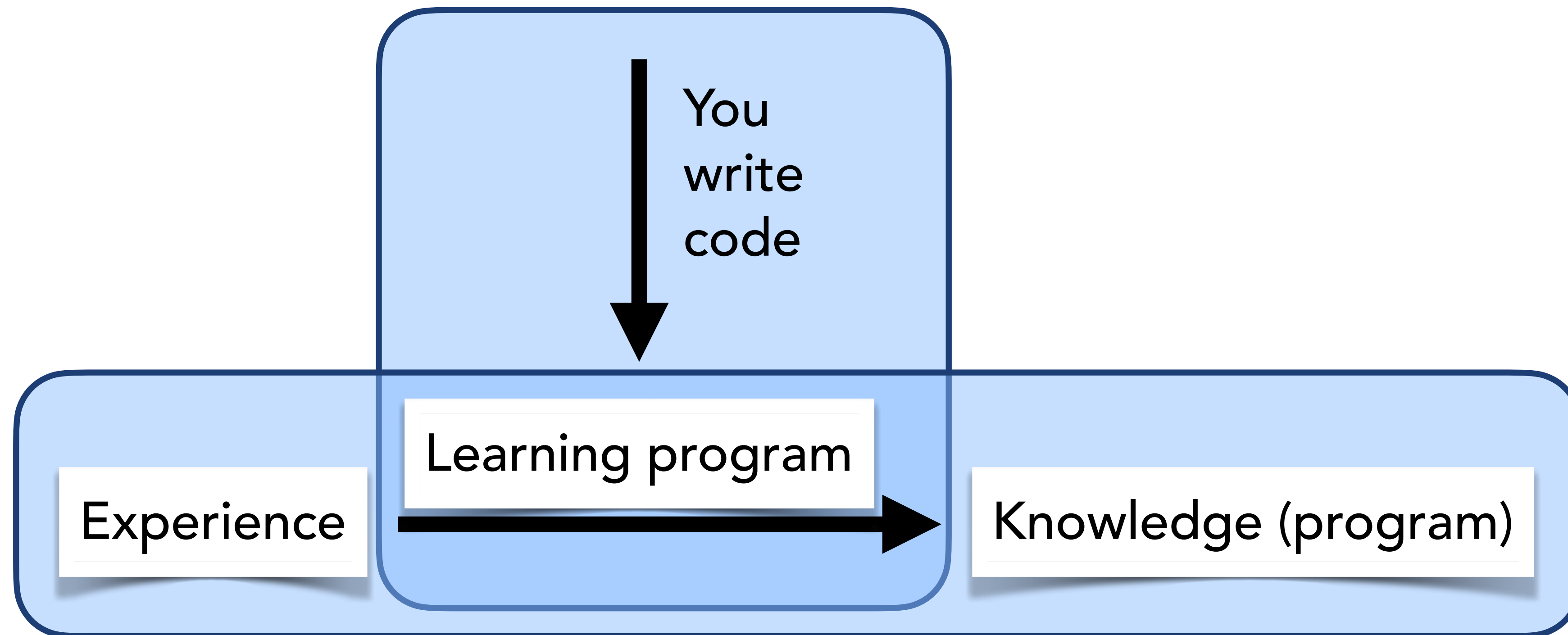


# You will learn to write a program that learns





# You will learn to write a program that learns



**Examples of what that looks  
like**



# Example: Predicting House Prices Based on # of Rooms

# Of Rooms	Price
2	\$200k
4	\$590k
3	\$350k
7	\$970k

# Example: Predicting House Prices Based on # of Rooms

# Of Rooms	Price
2	\$200k
4	\$590k
3	\$350k
7	\$970k

Experience

Learning (program) →

Knowledge (program)



# Example: Predicting House Prices Based on # of Rooms

# Of Rooms	Price
2	\$200k
4	\$590k
3	\$350k
7	\$970k

Prediction function  $f$ :

Input: # of rooms

Output: price

Example:  $f(5) = \$700k$

Experience

Learning (program) →

Knowledge (program)

# Example: Predicting House Prices Based on # of Rooms

# Of Rooms	Price
2	\$200k
4	\$590k
3	\$350k
7	\$970k

## Objective:

Write a Learning program that outputs a predictor  $f$ , such that,  $f$  can predict the price of any unseen house

## Prediction function $f$ :

Input: # of rooms

Output: price

Example:  $f(5) = \$700k$

Experience

Learning program

Knowledge (program)



**Supervised Learning** = Learning from a **batch** of **labeled randomly selected** experience

# Of Rooms	Price
2	\$200k
4	\$590k
3	\$350k
7	\$970k

**Objective:**

Write a Learning program that outputs a predictor  $f$ , such that,  $f$  can predict the price of any unseen house

**Prediction function  $f$ :**

Input: # of rooms

Output: price

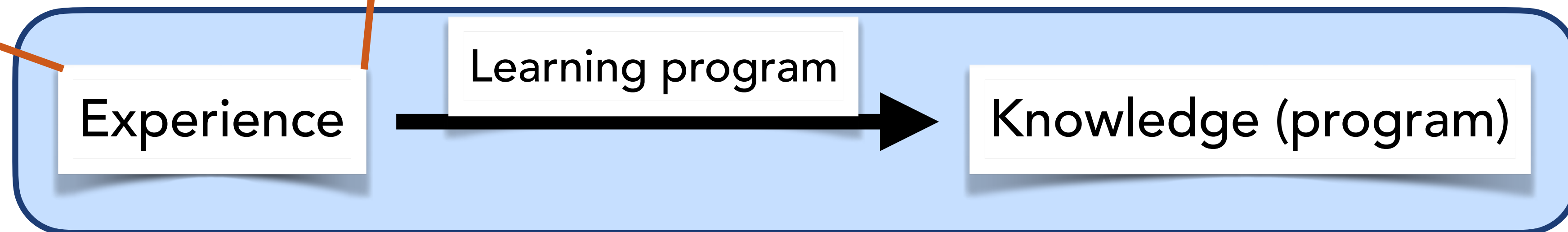
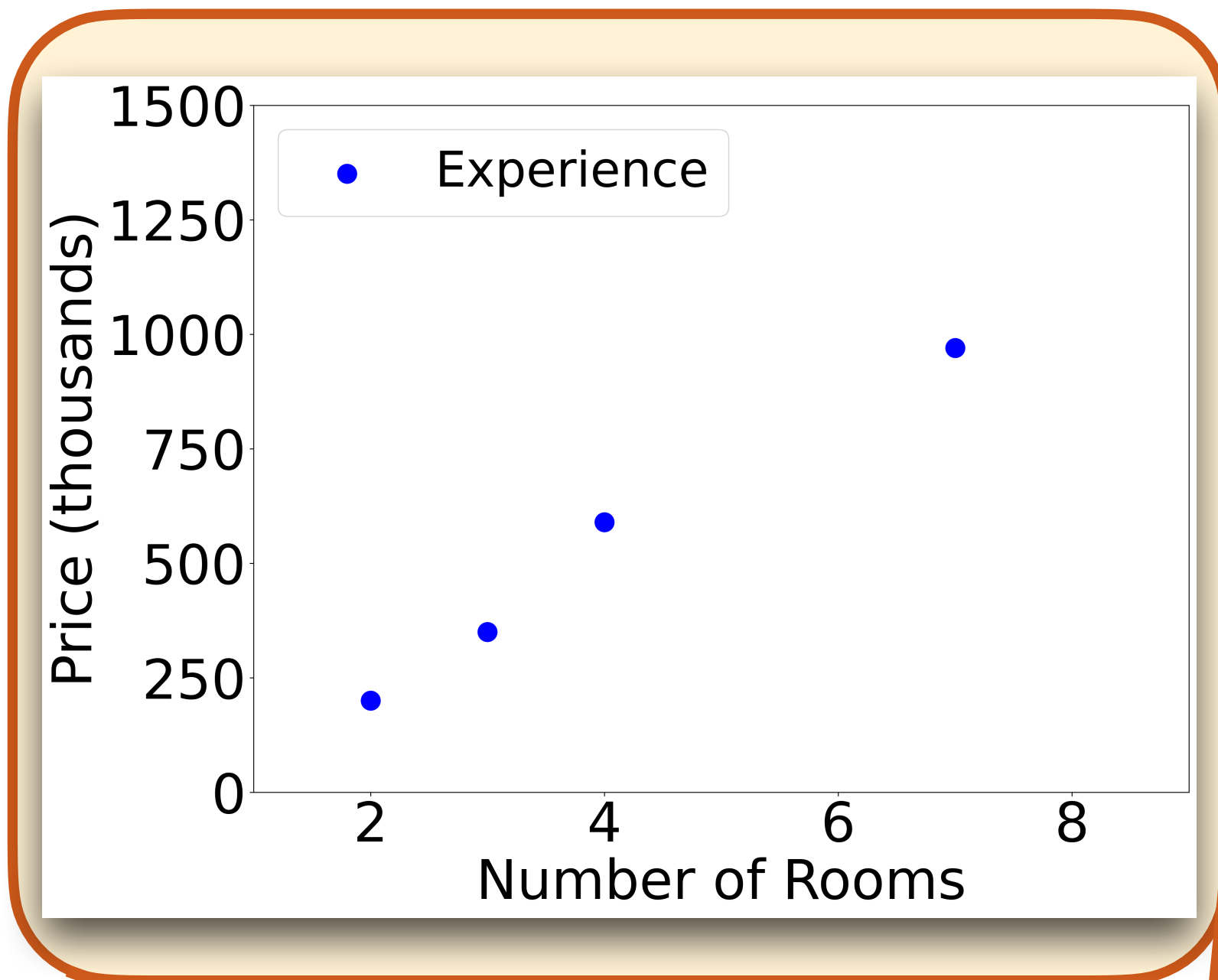
Example:  $f(5) = \$700k$

Experience

Learning program

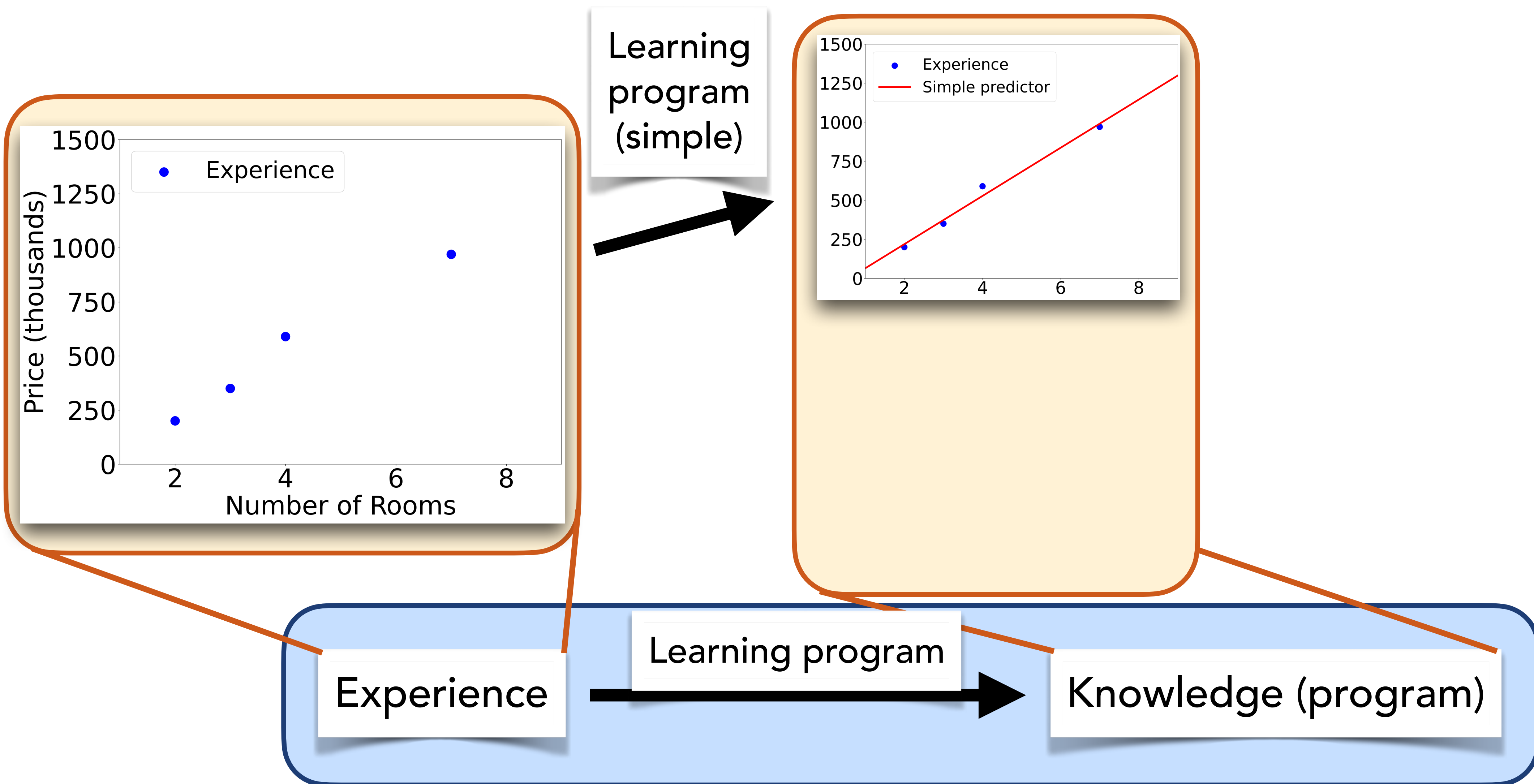
Knowledge (program)

# Example: Predicting House Prices Based on # of Rooms

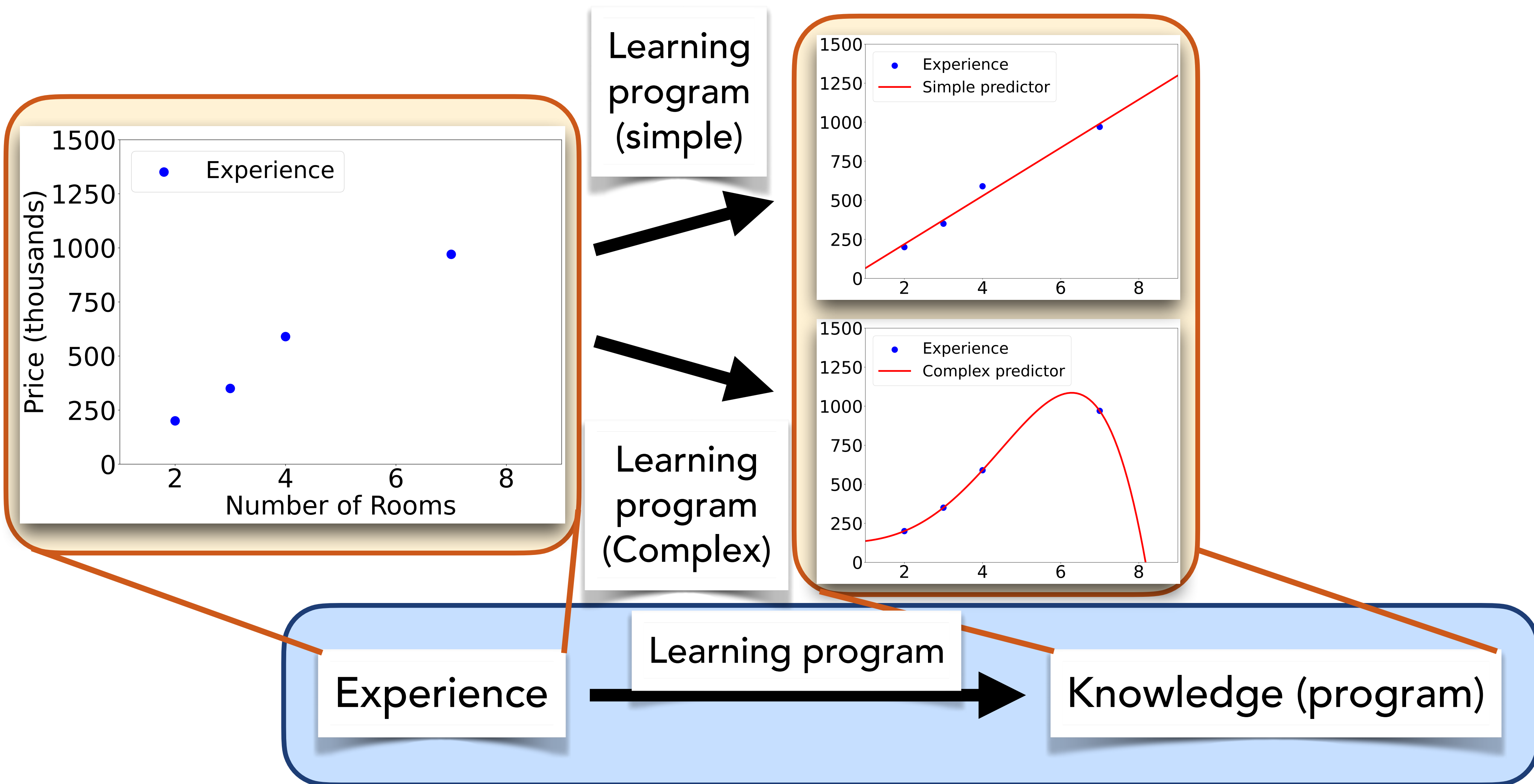




# Example: Predicting House Prices Based on # of Rooms

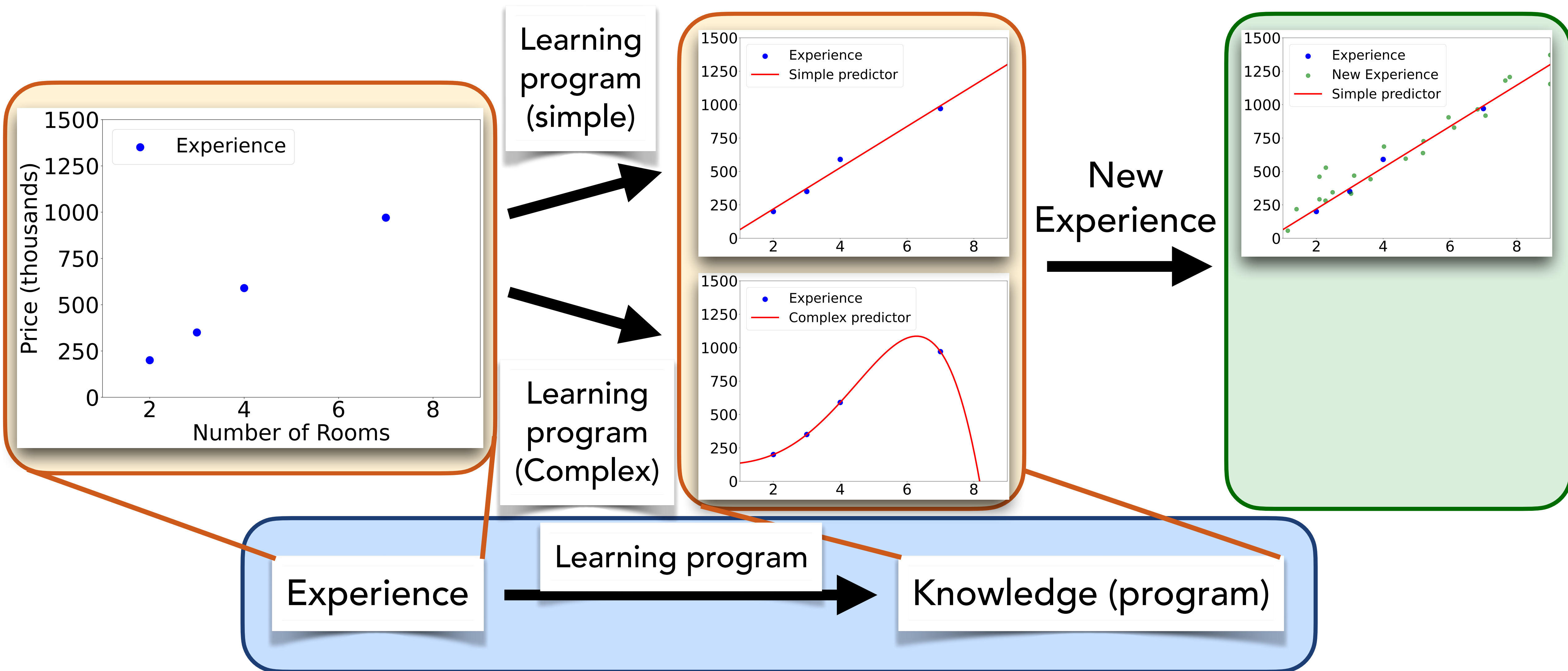


# Example: Predicting House Prices Based on # of Rooms

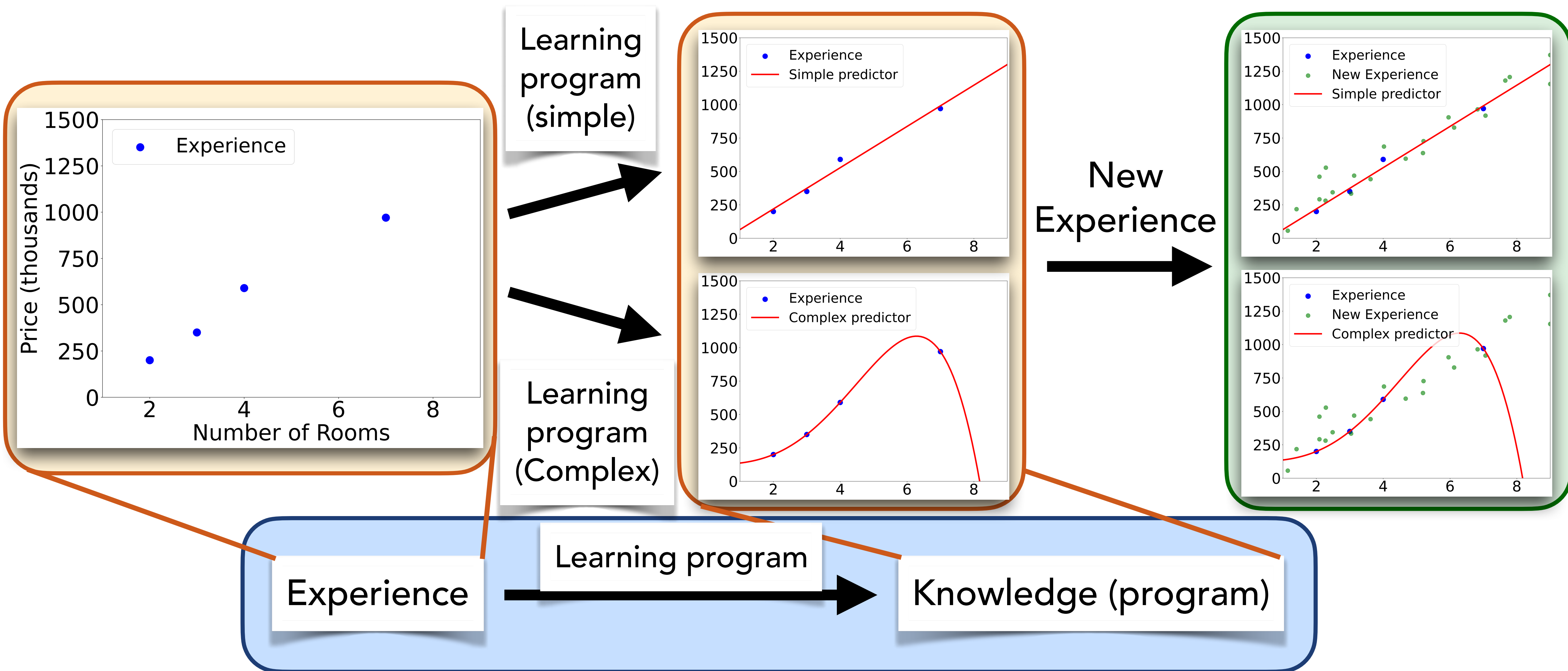




# Example: Predicting House Prices Based on # of Rooms



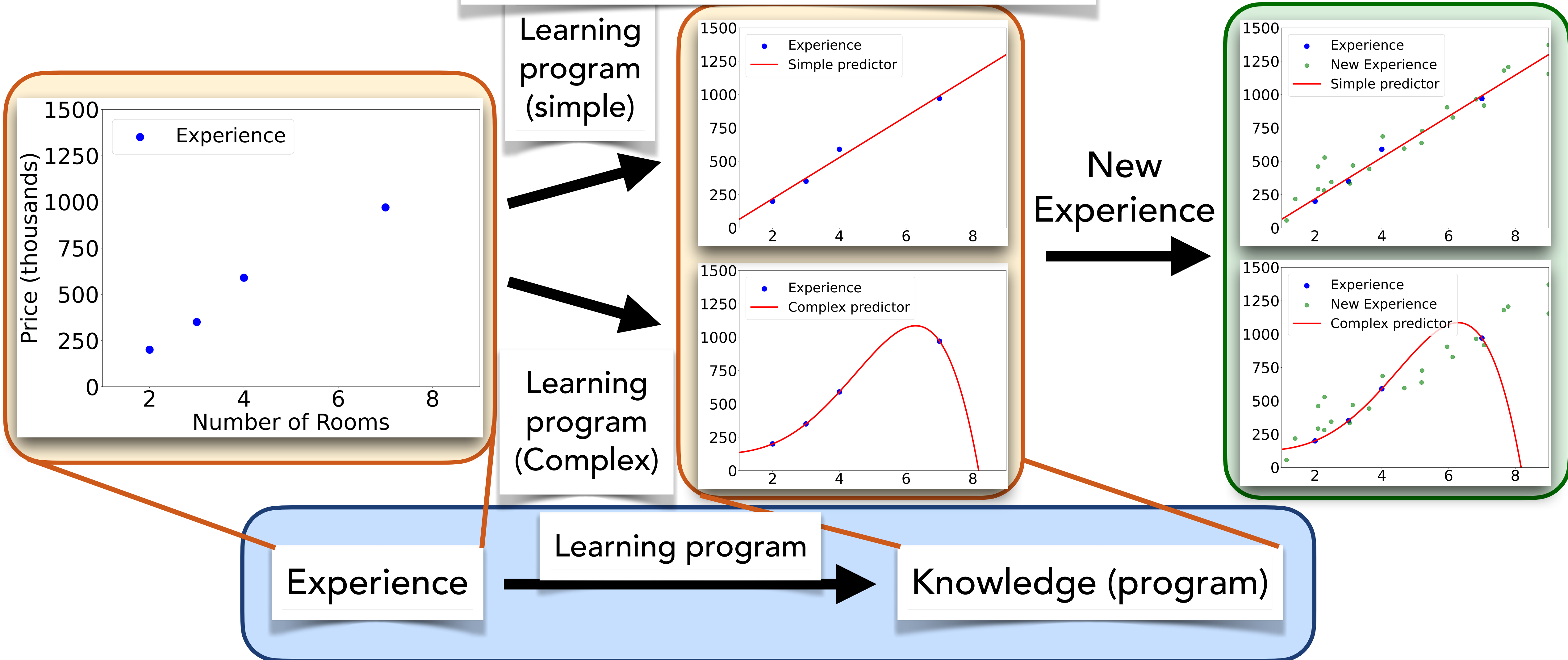
# Example: Predicting House Prices Based on # of Rooms





# Example: Prediction on # of Rooms

## Complex predictor is a Neural Network



# Example: Classifying Wine Based on Chemical Properties

Proline	Flavanoid	Type
2.3	3.4	Barolo
1.6	0.8	Not Barolo
⋮	⋮	⋮
2.8	3.5	Barolo



# Example: Classifying Wine Based on Chemical Properties

Proline	Flavanoid	Type
2.3	3.4	Barolo
1.6	0.8	Not Barolo
⋮	⋮	⋮
2.8	3.5	Barolo

Experience

Learning (program) →

Knowledge (program)

# Example: Classifying Wine Based on Chemical Properties

Proline	Flavanoid	Type
2.3	3.4	Barolo
1.6	0.8	Not Barolo
⋮	⋮	⋮
2.8	3.5	Barolo

**Prediction function  $f$ :**  
Input: Proline, Flavanoid  
Output: Type of wine  
Example:  $f(3,3) = \text{Barolo}$

Experience

Learning (program) →

Knowledge (program)



# Example: Classifying Wine Based on Chemical Properties

Proline	Flavanoid	Type
2.3	3.4	Barolo
1.6	0.8	Not Barolo
⋮	⋮	⋮
2.8	3.5	Barolo

## Objective:

Write a Learning program that outputs a predictor  $f$ , such that,  $f$  can predict the type of any unseen wine

## Prediction function $f$ :

Input: Proline, Flavanoid

Output: Type of wine

Example:  $f(3,3) = \text{Barolo}$

Experience

Learning program

Knowledge (program)

**Supervised Learning** = Learning from a **batch** of **labeled randomly selected** experience

Proline	Flavanoid	Type
2.3	3.4	Barolo
1.6	0.8	Not Barolo
⋮	⋮	⋮
2.8	3.5	Barolo

**Objective:**

Write a Learning program that outputs a predictor  $f$ , such that,  $f$  can predict the type of any unseen wine

**Prediction function  $f$ :**

Input: Proline, Flavanoid

Output: Type of wine

Example:  $f(3,3) = \text{Barolo}$

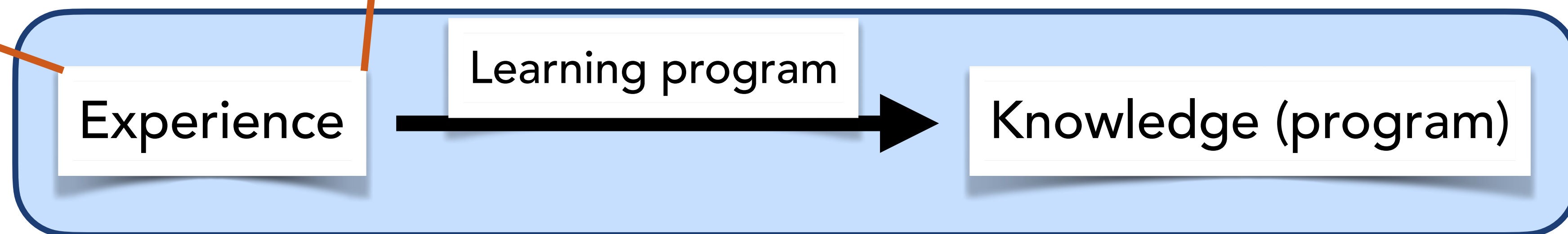
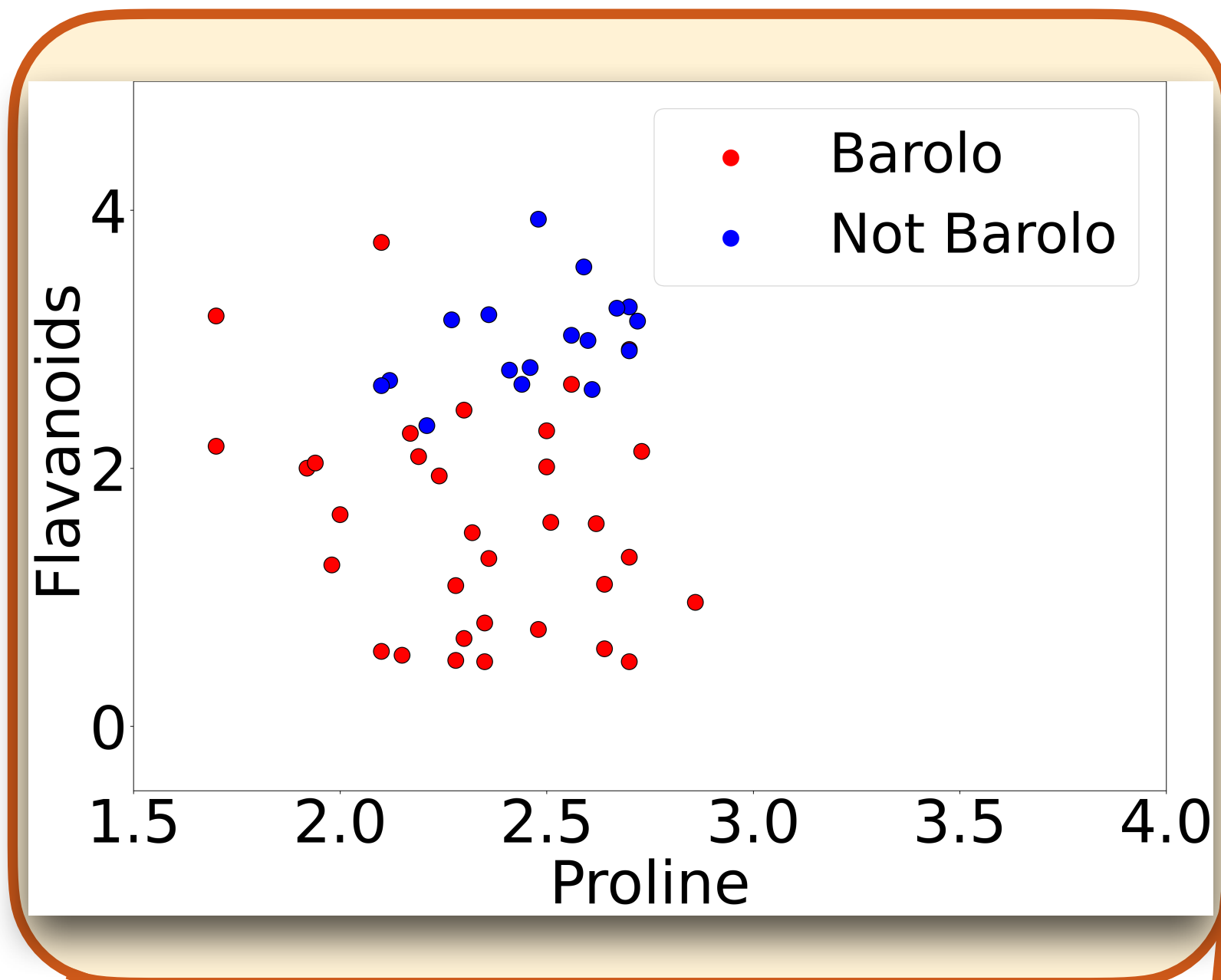
Experience

Learning program

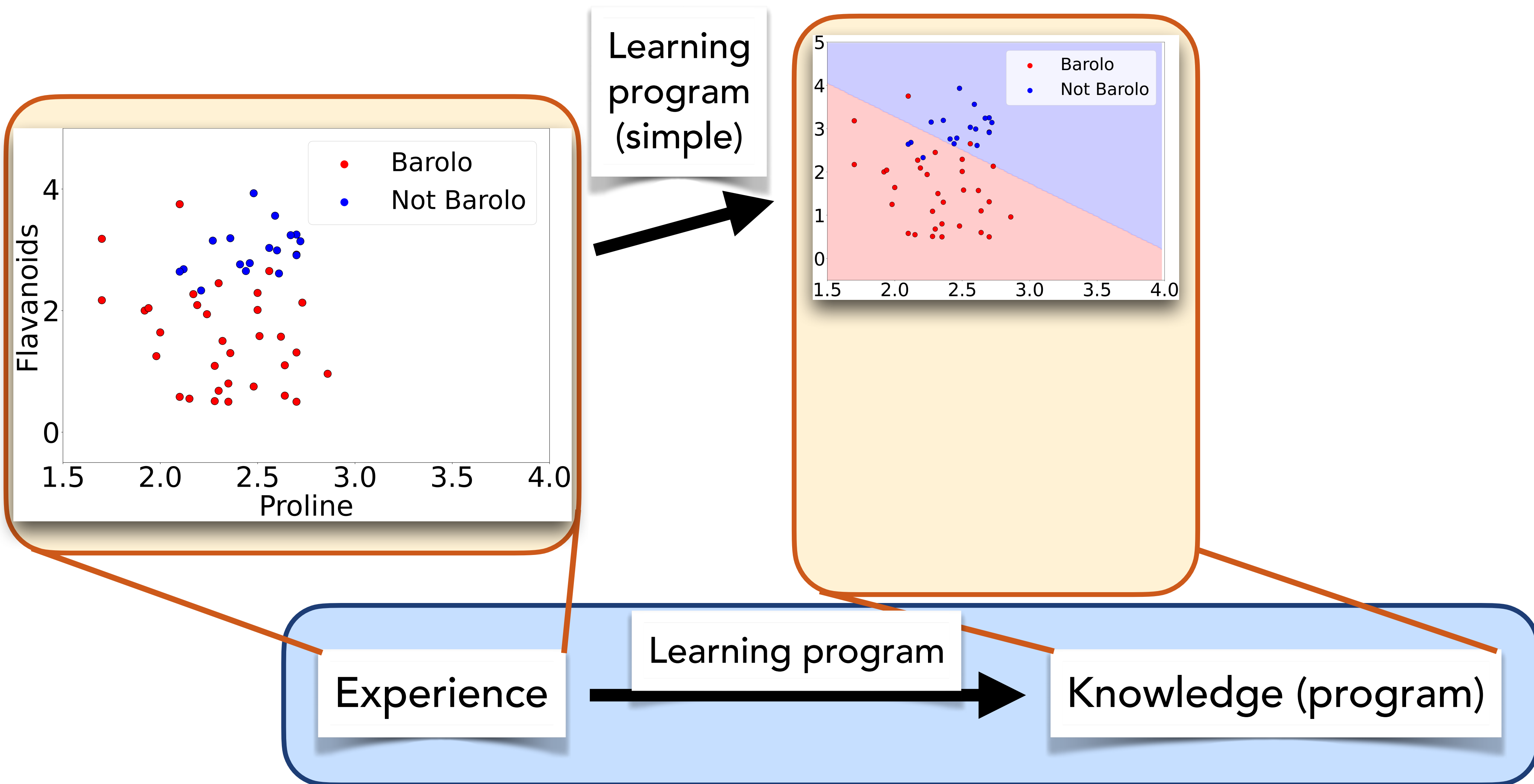
Knowledge (program)



# Example: Classifying Wine Based on Chemical Properties

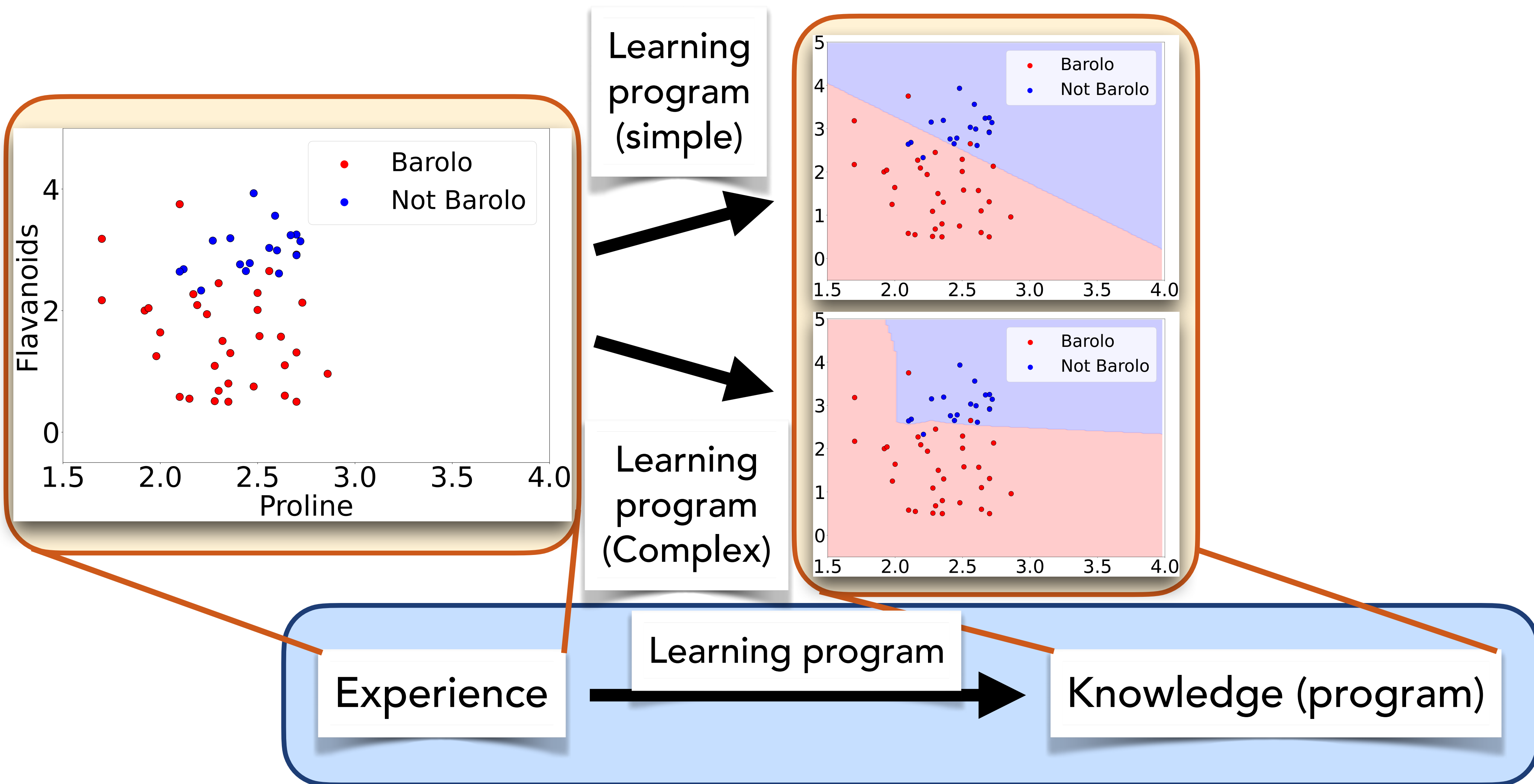


# Example: Classifying Wine Based on Chemical Properties

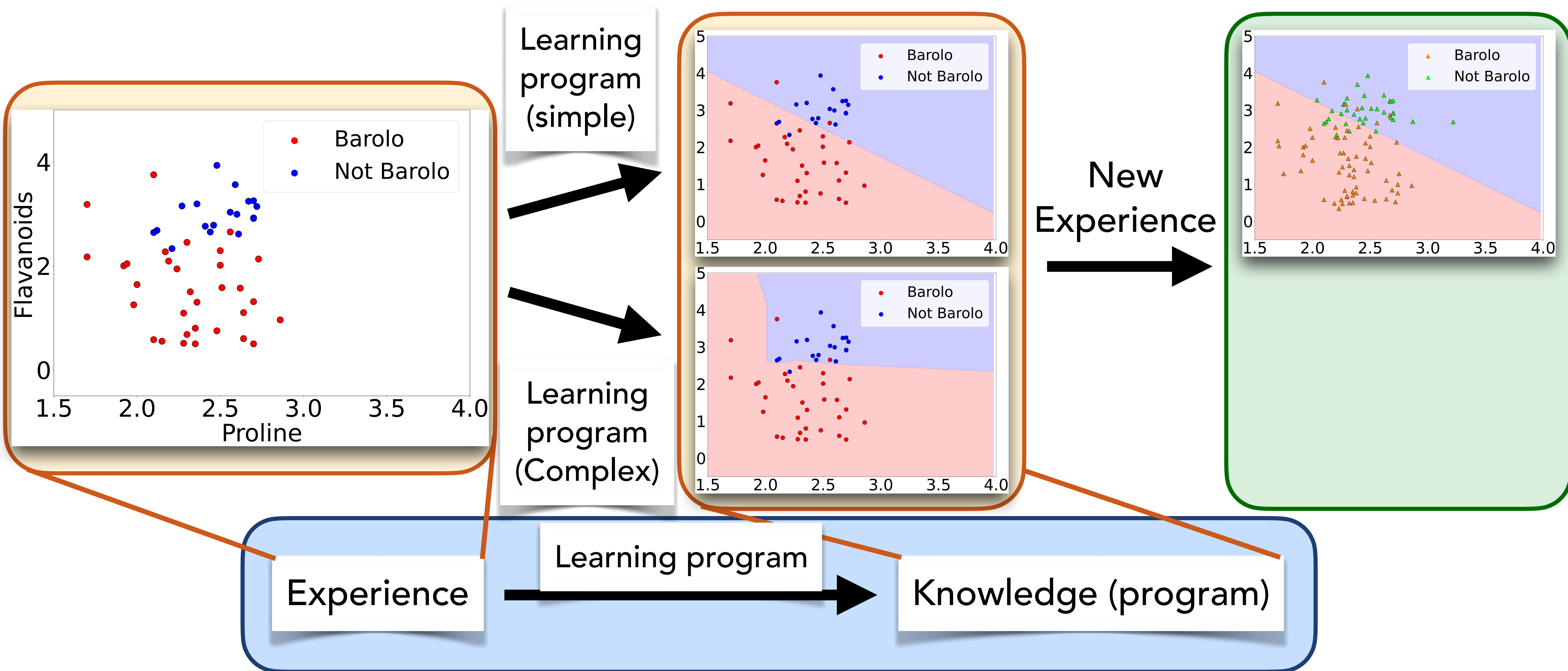




# Example: Classifying Wine Based on Chemical Properties

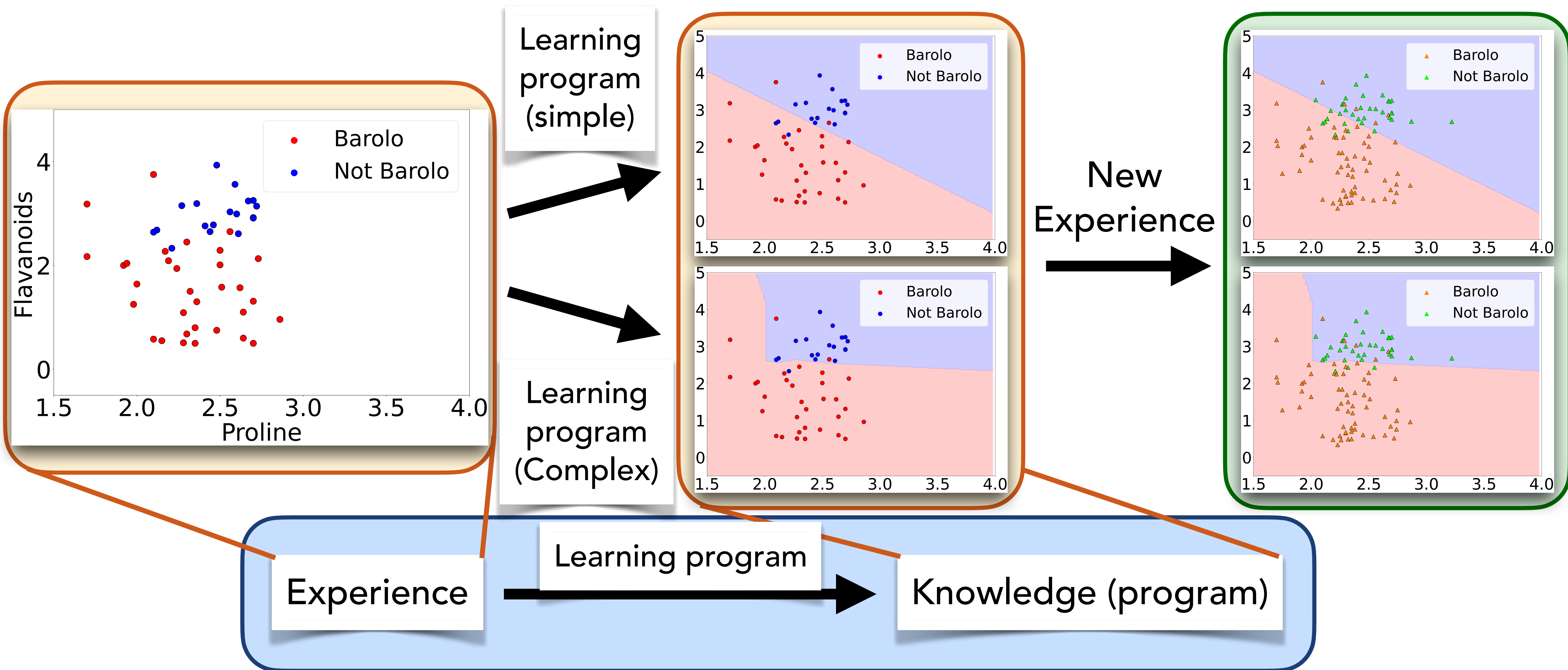


# Example: Classifying Wine Based on Chemical Properties





# Example: Classifying Wine Based on Chemical Properties



# Regression vs Classification

**Regression:** Labels are ordered (usually continuous) values (ex: house prices)

**Classification:** Labels are discrete and unordered (ex: type of wine)



# Course Outline

1. Math and probability review
2. Define supervised learning formally (splitting it into regression or classification)
3. Design some learning programs to solve regression problems

## **Midterm Exam 1**

4. Evaluate our learning programs
5. Present some new ways to design learning programs for regression

## **Midterm Exam 2**

6. Repeat the above for classification problems
7. Brief intro to neural networks and language models

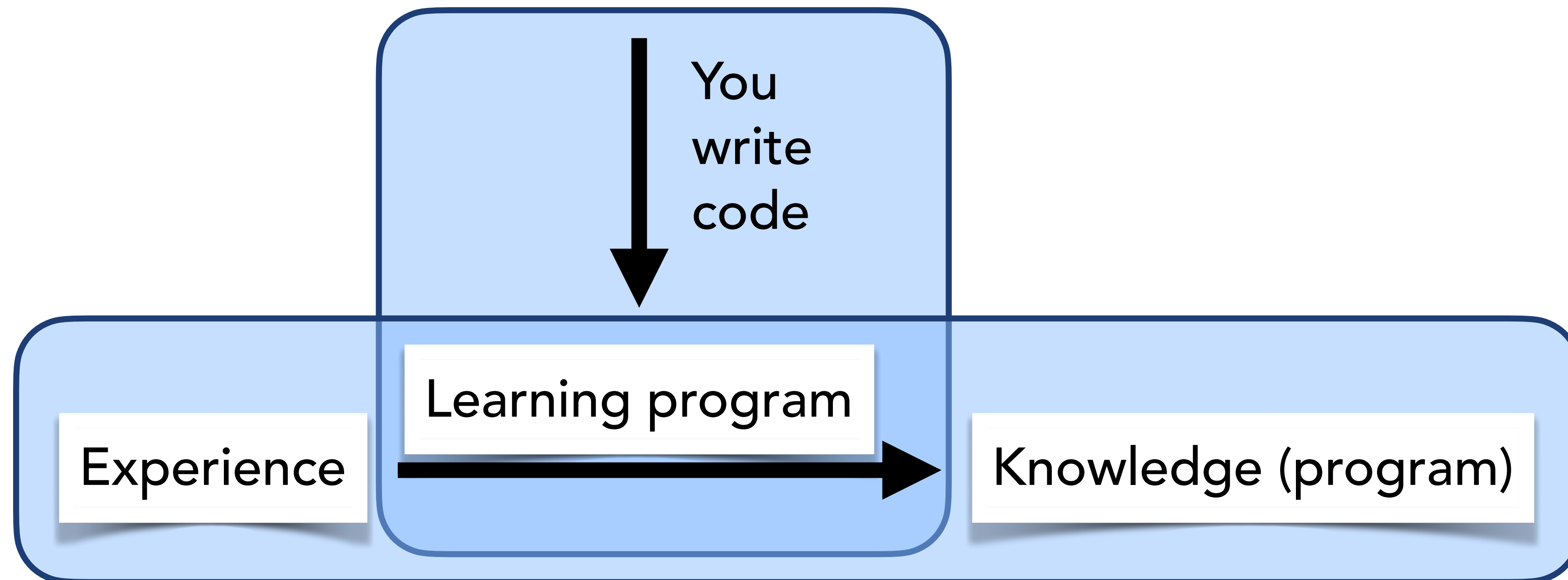
## **Final Exam (Cumulative)**

**The code for all of the plots  
was generated by**

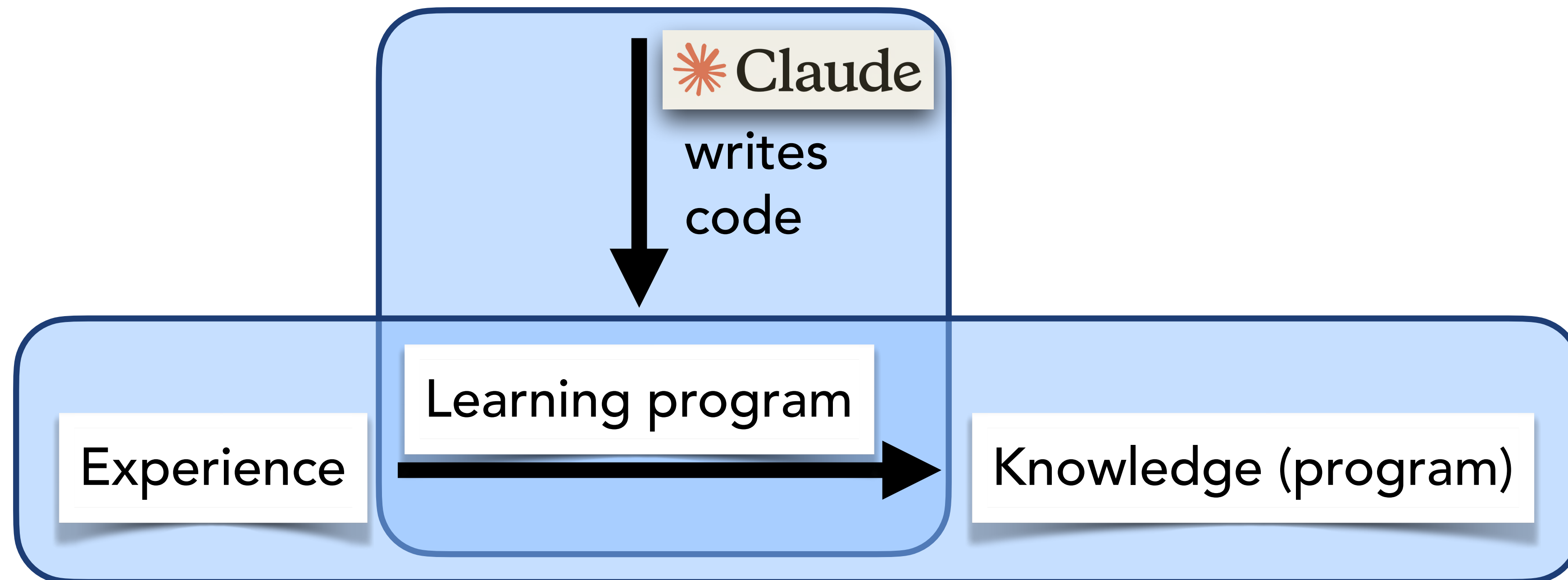




# Wait a Minute...

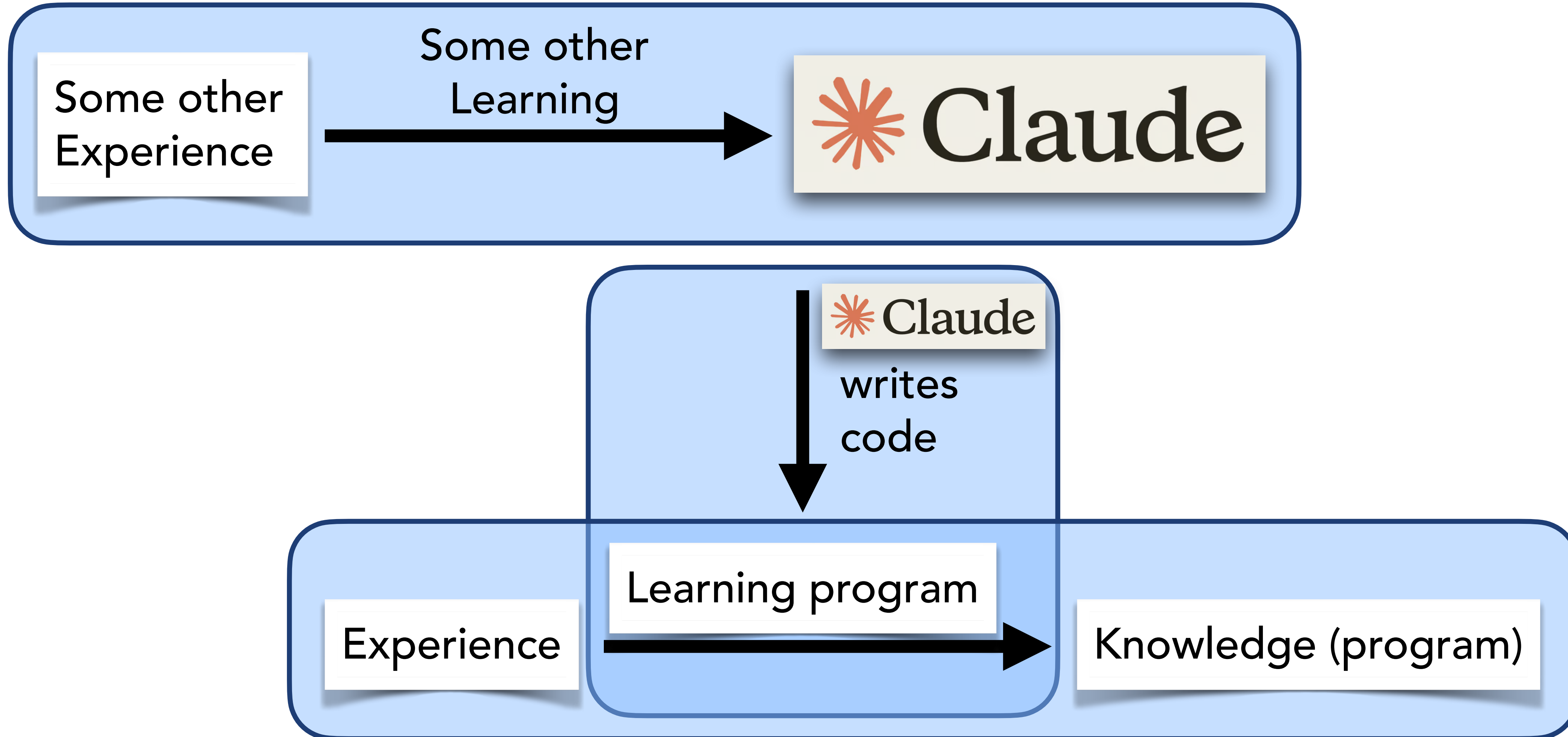


# Wait a Minute...

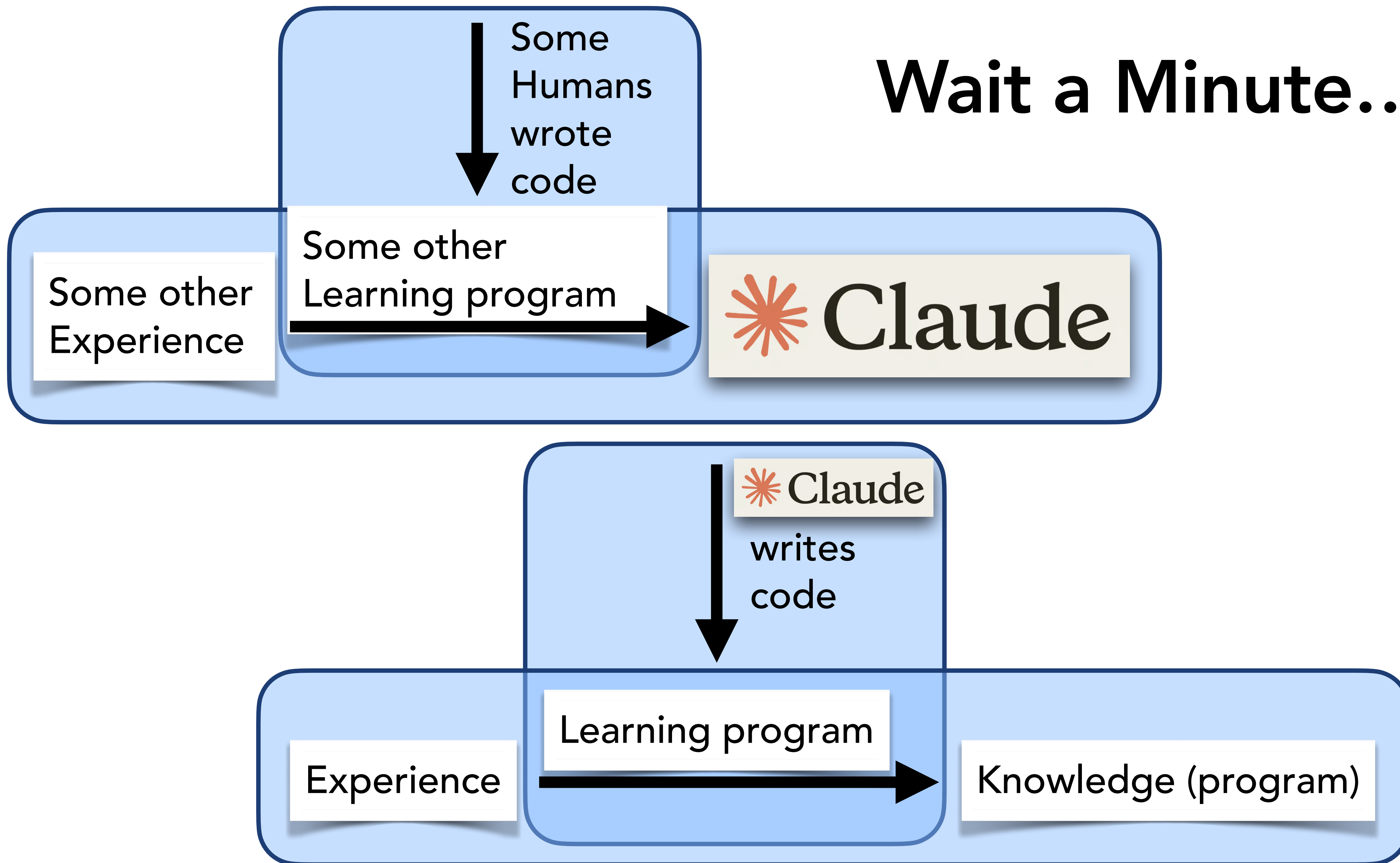





# Wait a Minute...




# Wait a Minute...






Why should you learn to write  
programs that can learn,  
If another program (ex:  Claude)  
can do it for you?

Why should you learn to write  
programs that can learn,  
If another program (ex:  Claude )  
can do it for you?

**My Answer:**

1. If something doesn't work, then you can fix it



**Why should you learn to write  
programs that can learn,  
If another program (ex:  Claude)  
can do it for you?**

**My Answer:**

1. If something doesn't work, then you can fix it
2. Its fun and feels like magic :)