ANSWERS

U12.1 Learning a behaviour

Learning, and the capstone · University · about 30 min

What this lesson is about

What learning buys over a hand-written rule, what it costs, and the smallest honest example.

- What learning is, exactly
- Three settings you will meet
- When learning is worth it
- When it is not
- About `position()` in this module
- The smallest example there is

Questions

7 marks in all

1. Which three things must be present for there to be anything to learn? [1 mark]

Tick every answer that is true.

- ☐ **A** A family of candidate answers, the hypothesis class
- ☐ **B** A score: a loss to make small or a reward to make large
- ☐ **C** A search that finds a good candidate
- ☐ **D** A neural network
- ☐ **E** Thousands of labelled examples

> **Answer:** A, B, C. Driving a leg, measuring it and dividing has all three. Networks and big data sets are one way to fill the slots, not requirements.

2. The data sheet says 20 cm/s at command 100. The robot drives at command 60 for 4 s and travels 44.2 cm. What does this print? [1 mark]

```
nominal = 20.0 * 60 / 100.0 * 4.0
measured = 44.2
print(nominal, measured, round(measured / nominal, 3))
```

> **Answer:**
>
> ```
> 48.0 44.2 0.921
> ```
>
> The data sheet predicts 12 cm/s for 4 s, 48 cm. The robot's gain is 44.2 / 48 = 0.921.

3. What is the name for the family of candidate answers a learner chooses between, such as all straight lines with a slope and an intercept? [1 mark]

> **Answer:** hypothesis class. Choosing the hypothesis class is a modelling decision you make, not something the data decides.

4. A robot tries lateral offsets on short trials and keeps whichever scores best. Which setting is this? [1 mark]

- ☐ **A** Policy search: the data is trials and their scores
- ☐ **B** Supervised learning: the data is inputs with the right answers attached
- ☐ **C** System identification: the data is commands and what the robot did
- ☐ **D** None, because no model is fitted

> **Answer:** A. Nobody can tell the robot the right offset, only how well a trial went, so the search has to be done by trying.

5. Which of these is a poor use of learning? [1 mark]

- ☐ **A** Learning the inverse kinematics from data when the geometry gives them exactly
- ☐ **B** Measuring this robot's actual speed per unit of command
- ☐ **C** Refitting the drive model after changing to a new mat
- ☐ **D** Learning what 64 depth readings mean about which way is passable

> **Answer:** A. When you have the equation, learning it wastes data and does worse than the formula. The others are measurable but not derivable, or they change.

6. In this module `position()` is available. What is the rule for using it? [1 mark]

- ☐ **A** The truth may inform the score, never the policy
- ☐ **B** It may be used anywhere, because it is the lab's measuring rig
- ☐ **C** It may be used by the policy but not by the score
- ☐ **D** It must not be used at all

> **Answer:** A. Ground truth labels and scores training, as motion capture does in a real lab. A policy that reads it could never be deployed.

7. The single gain from one 4 s run predicts the robot's travel better than the data sheet. What is still missing? [1 mark]

- ☐ **A** Any idea of how much to trust it: one run gives no spread
- ☐ **B** A loss function
- ☐ **C** A hypothesis class
- ☐ **D** A search

> **Answer:** A. The run has a hypothesis class, a loss and a search. Repeating it and reporting the mean and spread is what says how far to believe it.

The task: the data sheet is not your robot

Print `nominal:` the travel the data sheet predicts at command 60 for 4 seconds, `measured:` what this robot actually did, and `gain:` the ratio.

```
from bugbot import *
connect()

CMD = 60
SECONDS = 4.0
V_MAX = 20.0
```

The hint students can ask for: The data sheet says 20 cm/s at command 100, so command 60 for 4 seconds should be 48 cm. Drive it, measure what happened with the lab's camera, and divide. That ratio is a model of this robot with one parameter in it, fitted from one run.

A solution

```
from bugbot import *
connect()

CMD = 60
SECONDS = 4.0
V_MAX = 20.0                        # the data sheet: cm/s at command 100

nominal = V_MAX * CMD / 100.0 * SECONDS
print("nominal:", round(nominal, 1))

# the lab's overhead camera, which the robot itself does not have
x0, y0 = position()
forward(CMD)
wait(SECONDS)
stop()
wait(0.5)
x1, y1 = position()
measured = ((x1 - x0) ** 2 + (y1 - y0) ** 2) ** 0.5
print("measured:", round(measured, 1))
print("gain:", round(measured / nominal, 3))
```

Any program that meets the task's checks is marked correct in the simulator; this is one way, not the only way.