ANSWERS

F12.7 AI, robots and bias

Technology and society · GCSE · OCR J277 1.6.1, AQA 8525 3.8, Edexcel 1CP2 5.2.2 · about 20 min

What this lesson is about

What AI is good for, where it goes wrong, and auditing a training set for bias.

- What AI and machine learning are
- Where they help
- Where they go wrong
- Auditing a training set

Questions

6 marks in all

1. What is machine learning? [1 mark]

- ☐ **A** A program that finds patterns from examples instead of being given the rules
- ☐ **B** A robot that moves by itself
- ☐ **C** A faster processor
- ☐ **D** A kind of database

> **Answer:** A. The rules are learned from the training data.

2. Why can a face recognition system work worse for some groups? [1 mark]

- ☐ **A** Its training data did not represent them well
- ☐ **B** Their faces are harder to see
- ☐ **C** Cameras are biased by design
- ☐ **D** The software is too fast

> **Answer:** A. A model learns from the data it is shown.

3. Which is a real difficulty with machine learning decisions? [1 mark]

- ☐ **A** It can be hard to explain why a decision was made
- ☐ **B** They are always wrong
- ☐ **C** They cannot use data
- ☐ **D** They need no electricity

> **Answer:** A. The rules are learned, not written, so they cannot simply be read.

4. A self-driving car injures someone. What is the hard question? [1 mark]

- ☐ **A** Who is accountable: owner, manufacturer or programmer
- ☐ **B** Whether the car was fast
- ☐ **C** What colour the car was
- ☐ **D** Which fuel it used

> **Answer:** A. Accountability for automated decisions is still being settled in law.

5. A training set has 20 samples and 3 are 'night'. What percentage is that? [1 mark]

> **Answer:** 15. 3 ÷ 20 × 100.

6. What does this program print? [1 mark]

```
counts = {}
for s in ['day', 'day', 'night']:
    counts[s] = counts.get(s, 0) + 1
print(counts)
```

> **Answer:**
>
> ```
> {'day': 2, 'night': 1}
> ```
>
> Two days and one night.

The task: audit the training set

Count how many samples of each label are in `samples`. Print `<label>: <n> (<share>%)` for each, sorted from the most common to the least, with the share rounded to the nearest whole number. Any label making up less than 15% is under-represented: add `UNDER-REPRESENTED` to its line. At the end print `labels: <n>` and `under-represented: <labels>`, the under-represented labels joined by `, `.

```
# the two lines every program starts with: the commands, then the robot
from bugbot import *
connect()

samples = [
    "day", "day", "day", "day", "day", "day", "day", "day", "day", "day",
    "day", "day", "day", "dusk", "dusk", "night", "night", "night", "indoor", "day",
]
```

The hint students can ask for: Count each label into a dictionary, then sort from the most common down. A label's share is its count out of the total; mark the ones below the threshold and collect them for the last line.

A solution

```
from bugbot import *
connect()
samples = [
    "day", "day", "day", "day", "day", "day", "day", "day", "day", "day",
    "day", "day", "day", "dusk", "dusk", "night", "night", "night", "indoor", "day",
]
counts = {}
for s in samples:
    counts[s] = counts.get(s, 0) + 1
under = []
for label, n in sorted(counts.items(), key=lambda kv: kv[1], reverse=True):
    share = n / len(samples) * 100
    if share < 15:
        under.append(label)
        print(f"{label}: {n} ({share:.0f}%) UNDER-REPRESENTED")
    else:
        print(f"{label}: {n} ({share:.0f}%)")
print("labels:", len(counts))
print("under-represented:", ", ".join(under))
```

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