LESSON 1

What AI Models Are, from Chatbots to Agents

40 minutes | How Machines Learn

WHAT STUDENTS SHOULD BE ABLE TO DO

- Identify different kinds of AI model in plain terms, from generative AI (chatbots, image generators, code assistants) to recommendation systems, classifiers and AI agents
- Explain how each kind can be applied in coding, from an assistant that drafts or explains code to an agent that can plan and carry out a task
- Build a rule-based guesser in Scratch and find the cases its rules cannot cover

BEFORE THE BELL

- Students at devices with Scratch open at scratch.mit.edu. Signed in is best, because the project saves to their account. Anyone not signed in can still build and run the guesser, and saves it to the computer at the end (File, then Save to your computer).
- The five 'Guess the Model' examples are all on the step at once, so take them one at a time: the class calls each one before you give its answer. Answers: the next video, a recommendation system; sentence to picture, a generative model; finishing your code, a generative model (a code assistant); spotting a face, a classifier; booking the table, an agent.
- Model the first rule on the board; the rest of the build follows the same shape
- Class save routine agreed and displayed, as the guesser is saved at the end

What AI Models Are, from Chatbots to Agents

HOW THE LESSON RUNS

4 min	introduction	Start: What We're Building
4 min	content	Guess the Model
2 min	content	Four Kinds, and What Each Is for
2 min	content	Create a New Scratch Project
3 min	content	Your First Rule: Does It Bark?
3 min	content	Test It, and Find the Gap
3 min	content	Rule 2: Does It Purr?
3 min	content	Rule 3, and an Answer for Everything
3 min	content	Run It on the Easy Cases
4 min	content	Now the Awkward Ones
3 min	activity	Your Turn
3 min	content	Make Sense: What Worked, What Broke
2 min	content	Reflect
1 min	summary	What You Covered

TEACHING MOVES

- Keep 'Guess the Model' fast and noisy. It is a hook that draws on what students already meet daily rather than a test, so take answers and move on
- Build rule 1 on the board and ask for a prediction before you run it. The script ending silently on 'n' is the thing that makes rule 2 feel necessary
- The second wait until, the one that waits for the key to come back up, is worth naming out loud. Without it a single long press answers every question at once, and the bug looks like magic
- Do not rush the basenji. The guesser being confidently wrong is the lesson, not a detour from it, and it is what makes a model that learns from examples worth having

WHAT TO WATCH FOR

- Holding a key down rather than tapping it, so one press satisfies several wait until blocks and the cat answers instantly: point at the wait for the key release
- Adding more rules to catch the basenji and breaking a case that already worked: ask them to re-run the three easy cases after every new rule
- Expecting the guesser to be clever. It only ever does what the three rules say, which is exactly the point being made
- Building on a new sprite rather than the cat that is already there, so the say blocks appear to do nothing

DIFFERENTIATION

- Support: put the three finished rules on the board and have students build only the last one, then spend the time on the awkward cases
- Extension: add a third animal and count how many new rules it costs. The cost growing faster than the animals is the argument for learning from data
- Extension: draw the finished guesser as a flow chart and mark the path a basenji takes through it