Live coding

Table of contents

Lesson objectives

- Explain the advantages and limitations of participatory live (coding) instruction
- Summarize the key dos and do nots of participatory live (coding) instruction
- Demonstrate participatory live (coding) instruction
- Apply feedback to improve teaching practice

Overview

One thing we try to do when incorporating data science into the classroom is commonly referred to as "live coding": instructors do not use slides to teach, but work through the lesson material, typing in the code or instructions, with the students following along. This section explains how it works, why we use it, and gives general tips for an effective participatory live coding presentation. We will finish this section by practicing live coding and providing feedback for each other.

Why Participatory Live Coding?

We talk about the practice of "live coding". However, you are not live coding in a vacuum. Importantly, students are strongly encouraged to "code-along" with you. We refer to the practice of having the instructor live code and the students code along as "participatory live coding" or, less formally, as "code-along sessions".

Exercise

List some advantages and challenges of participatory live coding from both a student's and an instructor's point of view in the collaborative document or discuss with your group.

Live coding fits well into the model we have been discussing - by providing students with continuous opportunities for practice (every time they type in a line of code) and continuous feedback (their code either works or fails with an error message). It is important to keep in mind, however, that feedback is not helpful if you cannot understand it. Many error messages are obscure and not written with novices in mind. With this in mind, if you are doing live coding and make an error, it is an especially good opportunity to explain to your students *how* you go about addressing the error. It is totally OK to include Googling for the answer as one of your steps to address the error (but maybe not the *first* step you take).

Exercise

Watch this first participatory live coding demo video: https://youtu.be/bXxBeNkKmJE and this second demo video: https://youtu.be/SkPmwe_WjeY. Summarize your feedback on both videos in the collaborative document. Use the 2x2 (positive/negative, content/presentation) rubric for feedback we discussed earlier. In the videos, the instructor is teaching students how to use loops to repeat a task. Note: Sometime sounds in the room can be poor. Turning on closed captioning by pressing the [cc] button will improve the accessibility of these videos.

Round One

Exercise

Now it's your turn for participatory live coding

1. Split into groups of three.
2. Assign roles, which will rotate: presenter, timekeeper, note-taker.
3. Have each group member teach 5 minutes of your chosen lesson episode using live coding. For this exercise, your peers will not "code-along." Before you begin, briefly describe what you will be teaching and what has been learned previously.
4. After each person finishes, each group member should share feedback (starting with themselves) using the same 2x2 rubric as before (positive/negative, content/presentation). The timekeeper should keep feedback discussion to about 2 minutes per person; this may leave some time at the end for general discussion. The note-taker should record feedback in the collaborative document.
5. Trade off roles.

Top eight tips for live coding

1. Stand up and move around the room if possible. This makes the experience more interactive and less monotonous. Use a microphone if one is available to make it easier for people with hearing difficulties to hear you.
2. Go slowly. For every command you type, every word of code you write, every menu item or website button you click, say out loud what you are doing while you do it. Then point to the command and its output on the screen and go through it a second time. This slows you down and allows students to copy what you do, or to catch up. Do not copy-paste code.
3. Mirror your students' environment. Try to create an environment that is as similar as possible to what your students have to reduce cognitive load. Avoid using keyboard shortcuts.
4. Use your screen wisely. Use a big font, and maximize the window. A black font on a white background usually works better than a light font on a dark background.
5. Use illustrations to help students understand and organize the material. You can also generate the illustrations on the board as you progress through the material. This allows you to build up diagrams, making them increasingly complex in parallel with the material you are teaching. It helps students understand the material, makes for a more lively workshop and gathers the students' attention to you as well.

6. Leave no student behind. We will talk later in the workshop about strategies that can be used to gauge students' progress and understanding.
7. Embrace mistakes. No matter how well prepared you are, you will make mistakes. This is OK! Use these opportunities to do error framing and to help your students learn the art of troubleshooting.
8. Have fun! It is OK to use humor and improvisation to liven up the workshop. This becomes easier when you are more familiar with the material, and more relaxed. Start small, even just saying 'that was fun' after something worked well or you fixed an error is a good start.

Round Two

Exercise

Return to your groups and repeat the previous live-coding exercise, re-teaching the same content as before. This time, the presenter should incorporate changes based on feedback received, and everyone should try to "level up" their feedback.

When you are finished, add some thoughts on this process to the collaborative document: What did you change? Did it work better or worse with the change? How might you do it if you were to teach it again?

Feedback on the day

Your instructor will ask for you to provide feedback on this session.