

# Software + AI Setup

---

1405

Instructor: Ruiqing (Sam) Cao

# Checklist

☐ **Anaconda + Jupyter Notebook**

☐ **Virtual Environment (venv1405)**

☐ **Visual Studio Code**

☐ **VS Code + GitHub Copilot Integration**

# Identify Your Operating System (OS)

---

- If your laptop is one of the following brands, it likely runs **Windows OS**: *Lenovo, HP (Hewlett-Packard), Dell, Acer, Microsoft (Surface series), ASUS, Huawei*
- If your laptop is a MacBook, it likely runs **macOS** and uses
  - **Intel chip** if your macOS matches one of the following versions: *Catalina (10.15), Mojave (10.14), High Sierra (10.13), Sierra (10.12), El Capitan (10.11), Yosemite (10.10), Mavericks (10.9), Mountain Lion (10.8)*
  - **M1 chip** if your macOS version is *Sonoma (14.x), Sequoia (15.x), Tahoe (26)*
  - *Either an Intel chip or an M1 chip* if your macOS matches one of the following versions: *Ventura (13.x), Monterey (12.x), Big Sur (11.x)*

# Download Anaconda for Your OS

- Go to <https://www.anaconda.com/download/success> (or visit the Download page for Anaconda and skip registration)

## Choose Your Download

WindowsMacLinux

### Anaconda Distribution

Complete package with 8,000+ libraries, Jupyter, JupyterLab, and Spyder IDE. Everything you need for data science.

[↓ Windows 64-Bit Graphical Installer](#)

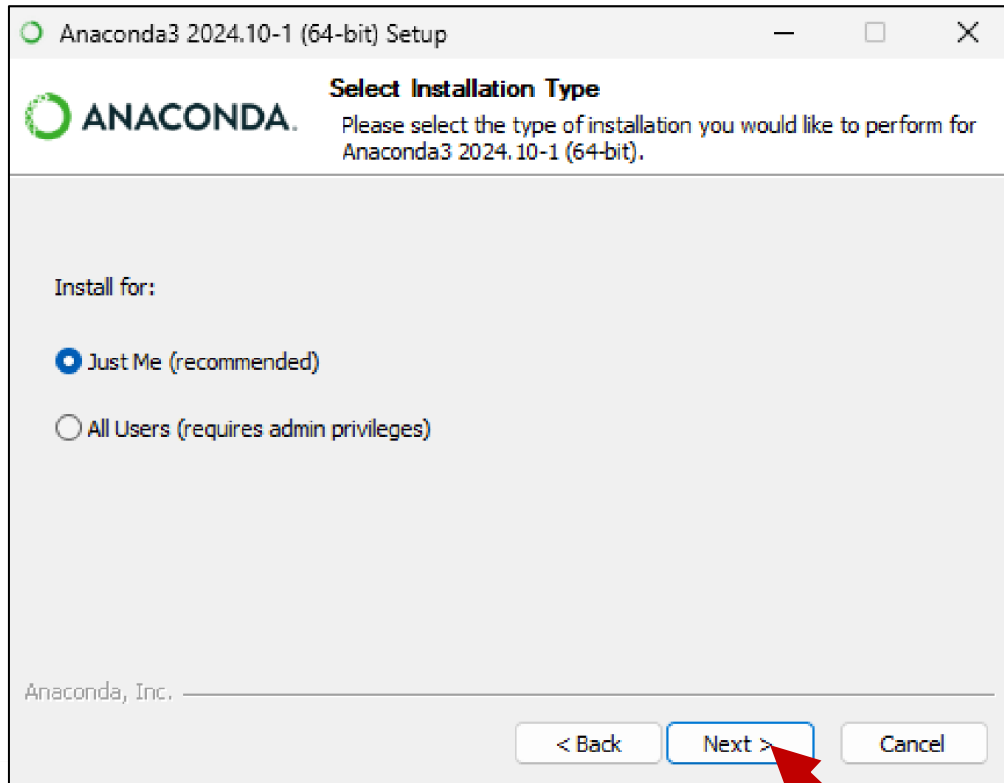
### Miniconda

Minimal installer with just Python, Conda, and essential dependencies. Install only what you need.

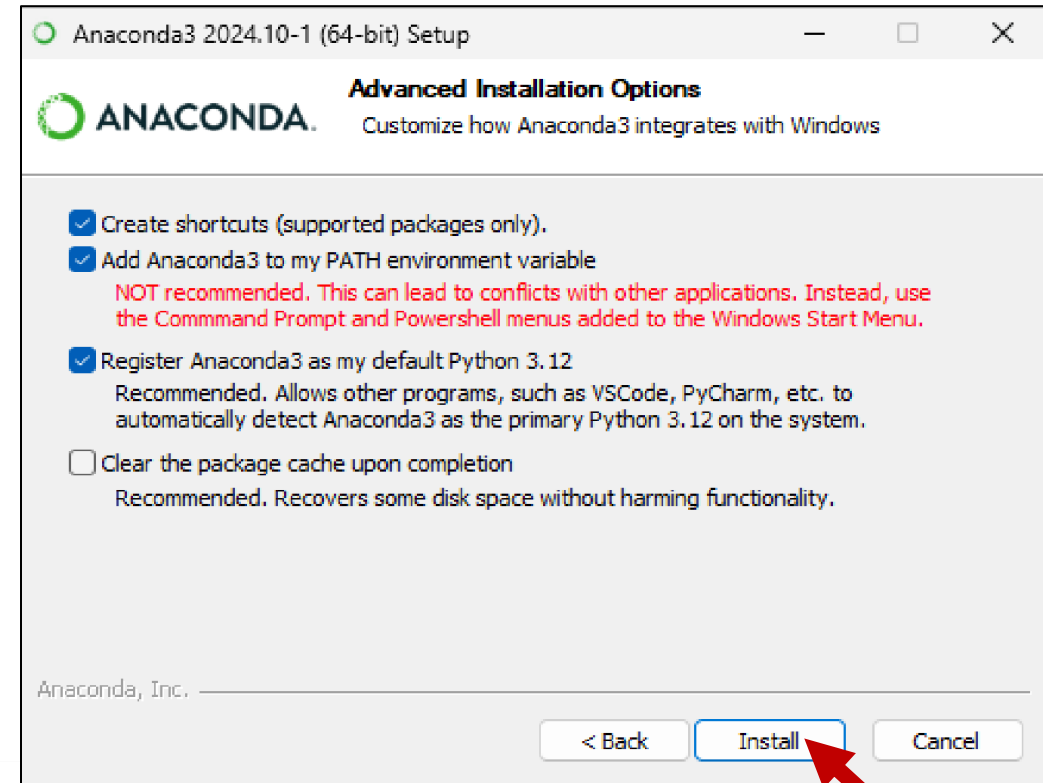
[↓ Windows 64-Bit Graphical Installer](#)

# Install Anaconda (Windows)

Select “Just Me” to only install Anaconda for the local user



Select “Add Anaconda 3 to my PATH” (ignore recommendation)



# Check Anaconda Directory & Version

- Open Anaconda Prompt and run "echo %PATH%" to locate Anaconda. Look for file paths containing "anaconda3"

**C:\Users\Sam\anaconda3**



Anaconda path

- Run "conda --version" to confirm Anaconda installation. The output should display the version number, indicating it is ready to use

**conda 24.9.2**



version

# Checklist

☒ **Anaconda + Jupyter Notebook**

☐ **Virtual Environment (venv1405)**

☐ **Visual Studio Code**

☐ **VS Code + GitHub Copilot Integration**

# Create a Virtual Environment

---

- Create separate virtual environments for different projects, especially when the projects involve distinct libraries (e.g., visualization, traditional ML, deep learning)
- Make sure you are connected to the Internet. Open Anaconda Prompt and run the command below to create a virtual environment named venv1405 for this course

```
conda create -n venv1405 python=3.14
```

- When prompted `proceed ([y]/n)?` Type `y` and hit Enter
- You will see this when venv1405 is set up:

```
# To activate this environment, use
#
#     $ conda activate venv1405
#
# To deactivate an active environment, use
#
#     $ conda deactivate
```



# Use a Virtual Environment

---

Open Anaconda Prompt

- To activate the virtual environment venv1405, run:  
`conda activate venv1405`
- To list all Conda environments and see which one is active (marked with an asterisk \*), run:  
`conda info --envs` or `conda env list`
- To deactivate the current virtual environment, run:  
`conda deactivate`

# Register a Virtual Environment

---

To use a virtual environment in Jupyter Notebook, register it in the IPython kernel with these steps:

- Activate the virtual environment

```
conda activate venv1405
```

- Make sure `ipykernel` is installed inside the virtual environment

```
pip install ipykernel
```

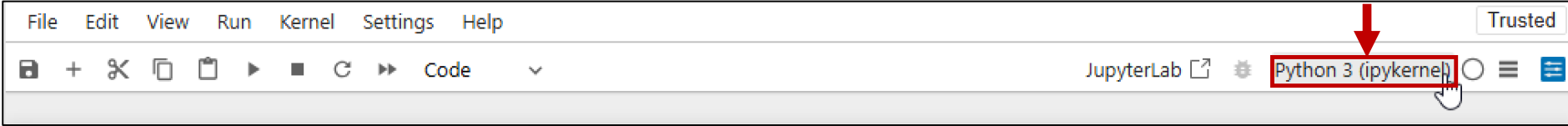
- Add the virtual environment to Jupyter as a kernel, and choose a display name for it within Jupyter

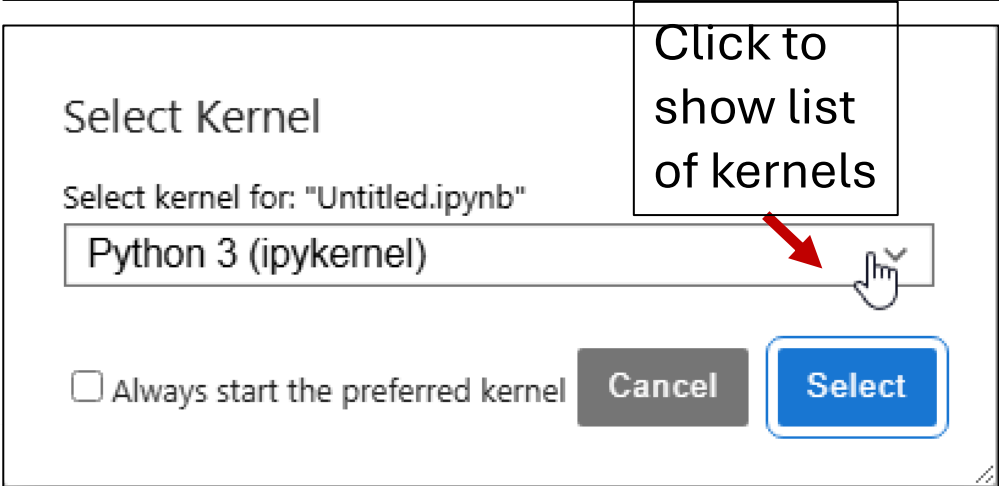
```
python -m ipykernel install --user --name=venv1405 -  
-display-name "Python (venv1405)"
```

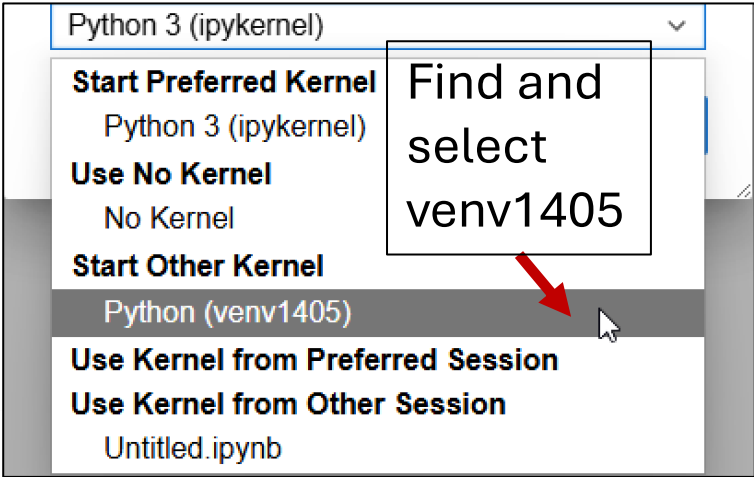
# Select a Virtual Environment in Jupyter

Open Anaconda Prompt (make sure venv1405 is deactivated here)

- Run “jupyter notebook” to open the Jupyter Notebook interface in your web browser. Create a new Notebook (.ipynb) or open an existing one, and click on ipykernel from the top-right corner

1. 

2. 

3. 

# Checklist

✓ **Anaconda + Jupyter Notebook**

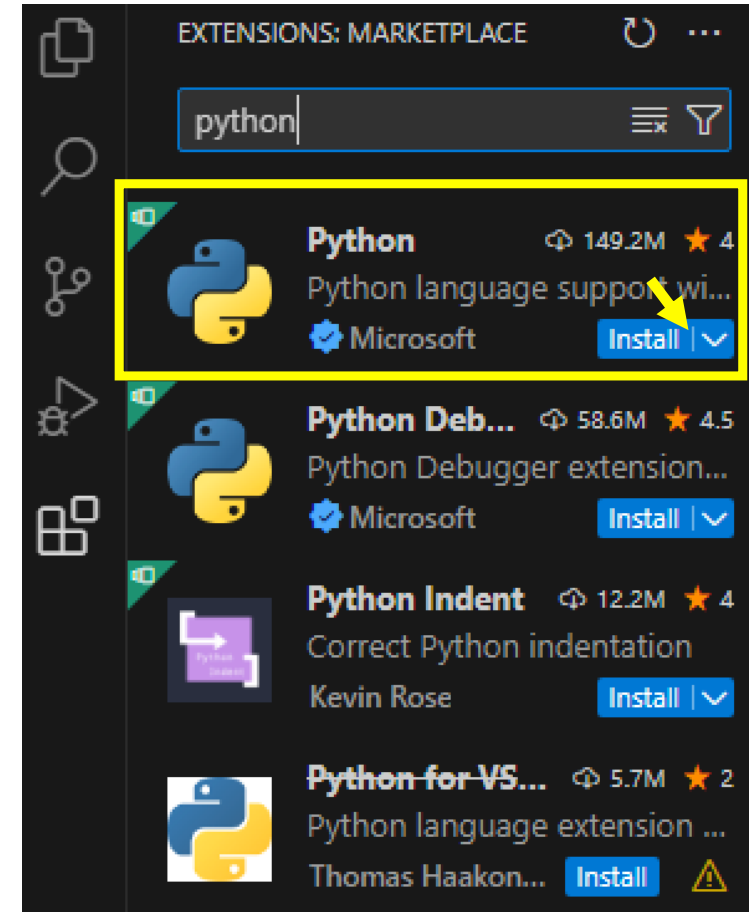
✓ **Virtual Environment (venv1405)**

☐ **Visual Studio Code**

☐ **VS Code + GitHub Copilot Integration**

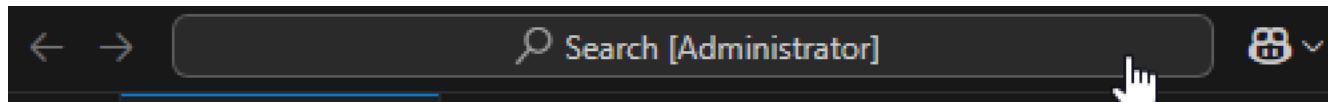
# Install Visual Studio Code (VS Code)

- Visit <https://code.visualstudio.com> and download the appropriate Visual Studio Code version for your OS
- Locate and run the downloaded executable (.exe), and follow the instructions to install VS Code
- Open Visual Studio Code, go to *View > Extensions*, search for "Python" in the Marketplace, and install the Python extension by Microsoft

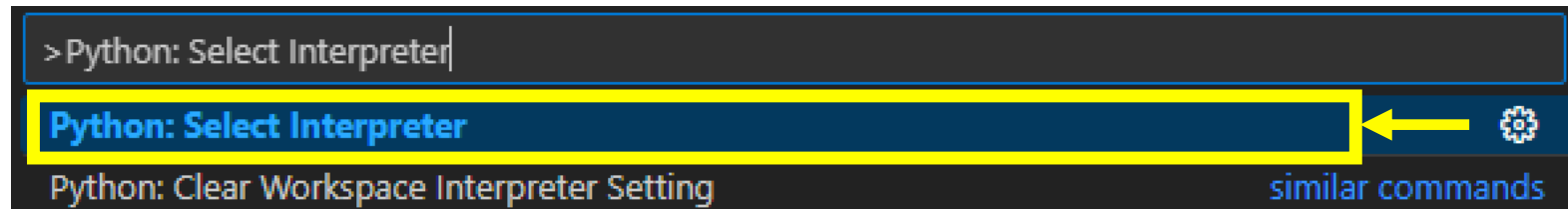


# Configure Python Interpreter in VS Code

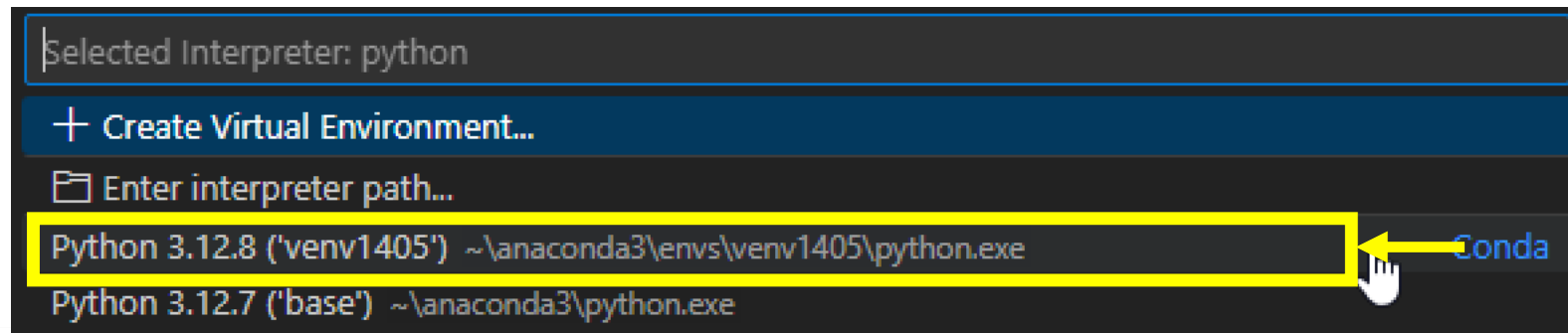
- Find the search bar



- Type ">Python: Select Interpreter" and select from the dropdown list

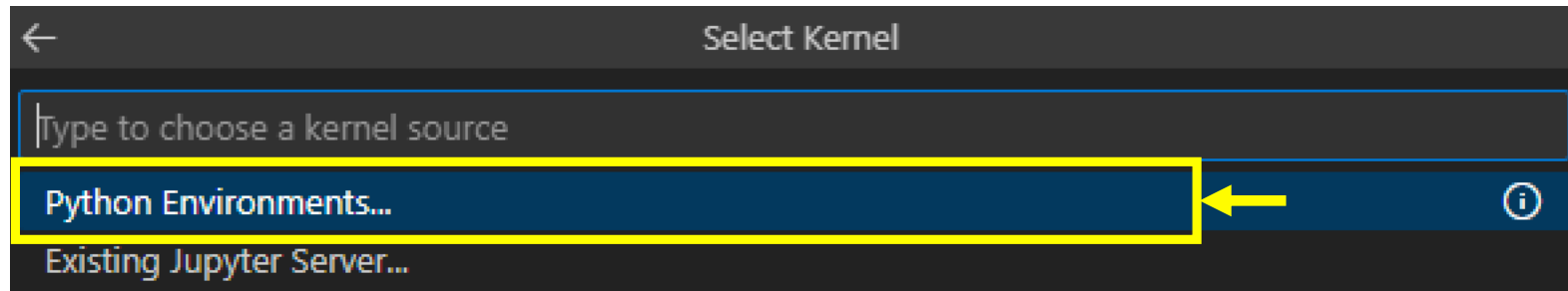


- Find the virtual environment "venv1405" and click on it

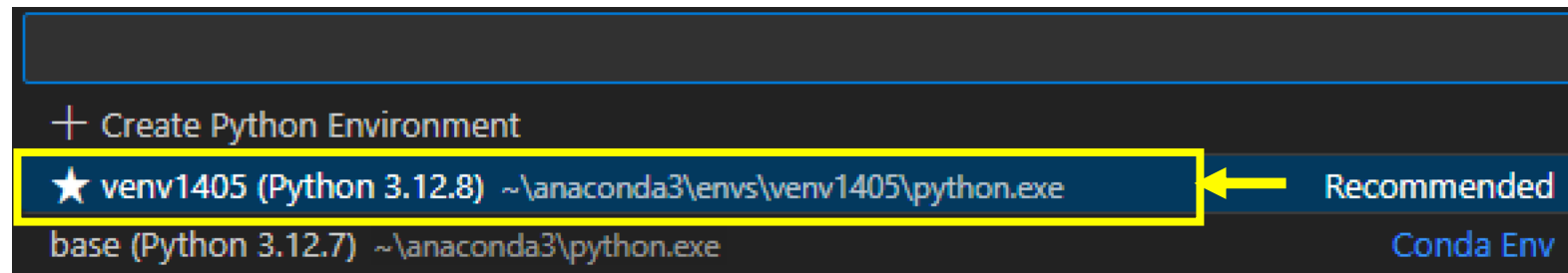


# Select Kernel to Run Jupyter in VS Code

- If you try to run a code block in a Jupyter notebook (.ipynb), you may be asked to select a kernel, click on “Python Environments”



- Select the virtual environment venv1405 created for the course



# Checklist

✓ **Anaconda + Jupyter Notebook**

✓ **Virtual Environment (venv1405)**

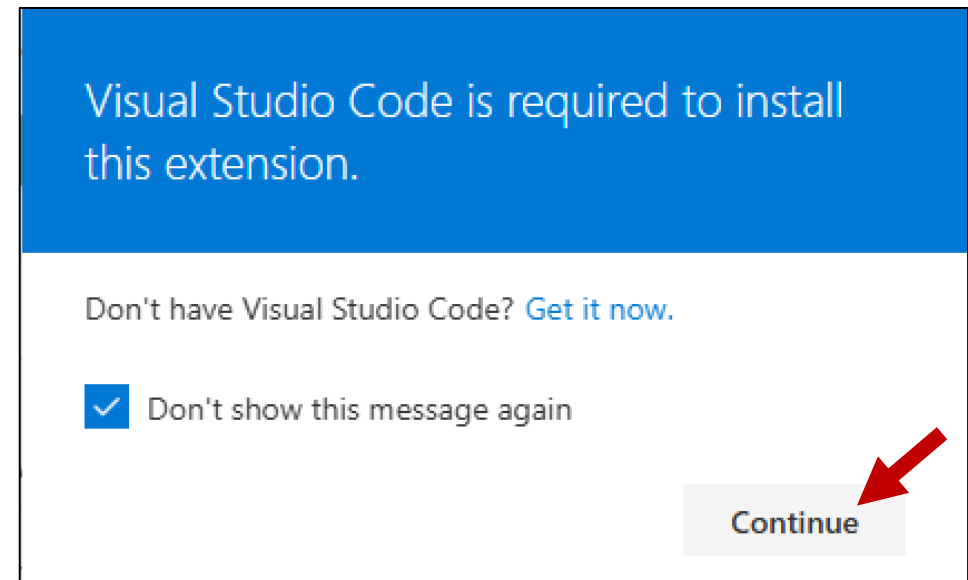
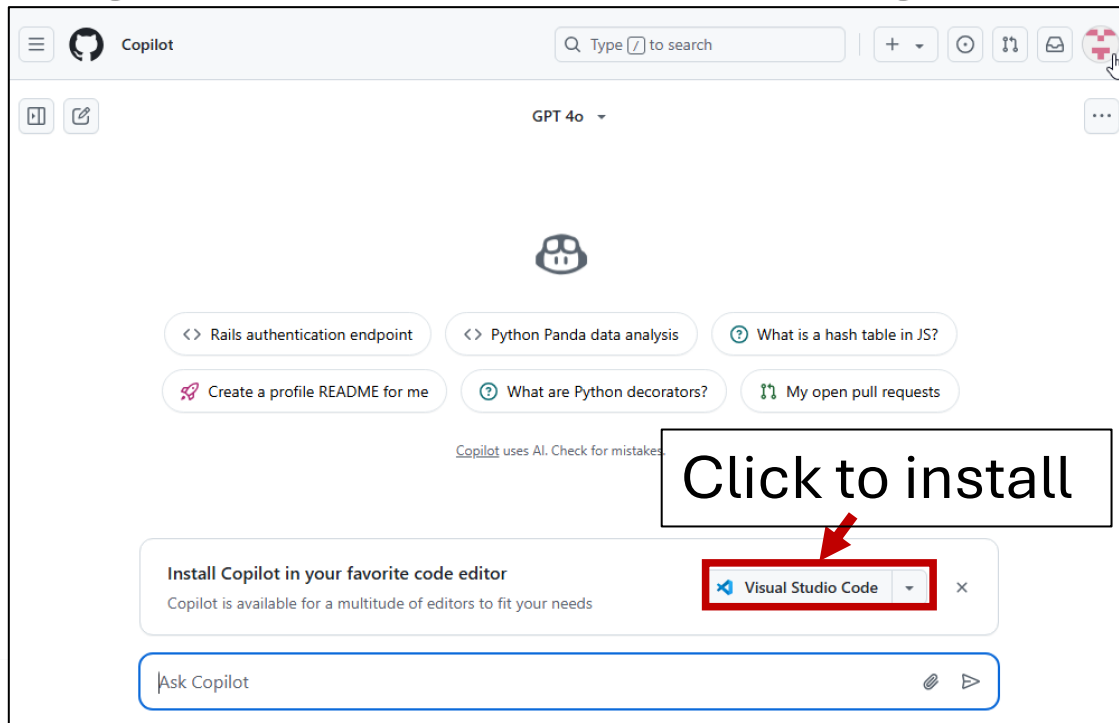
✓ **Visual Studio Code**

☐ **VS Code + GitHub Copilot Integration**



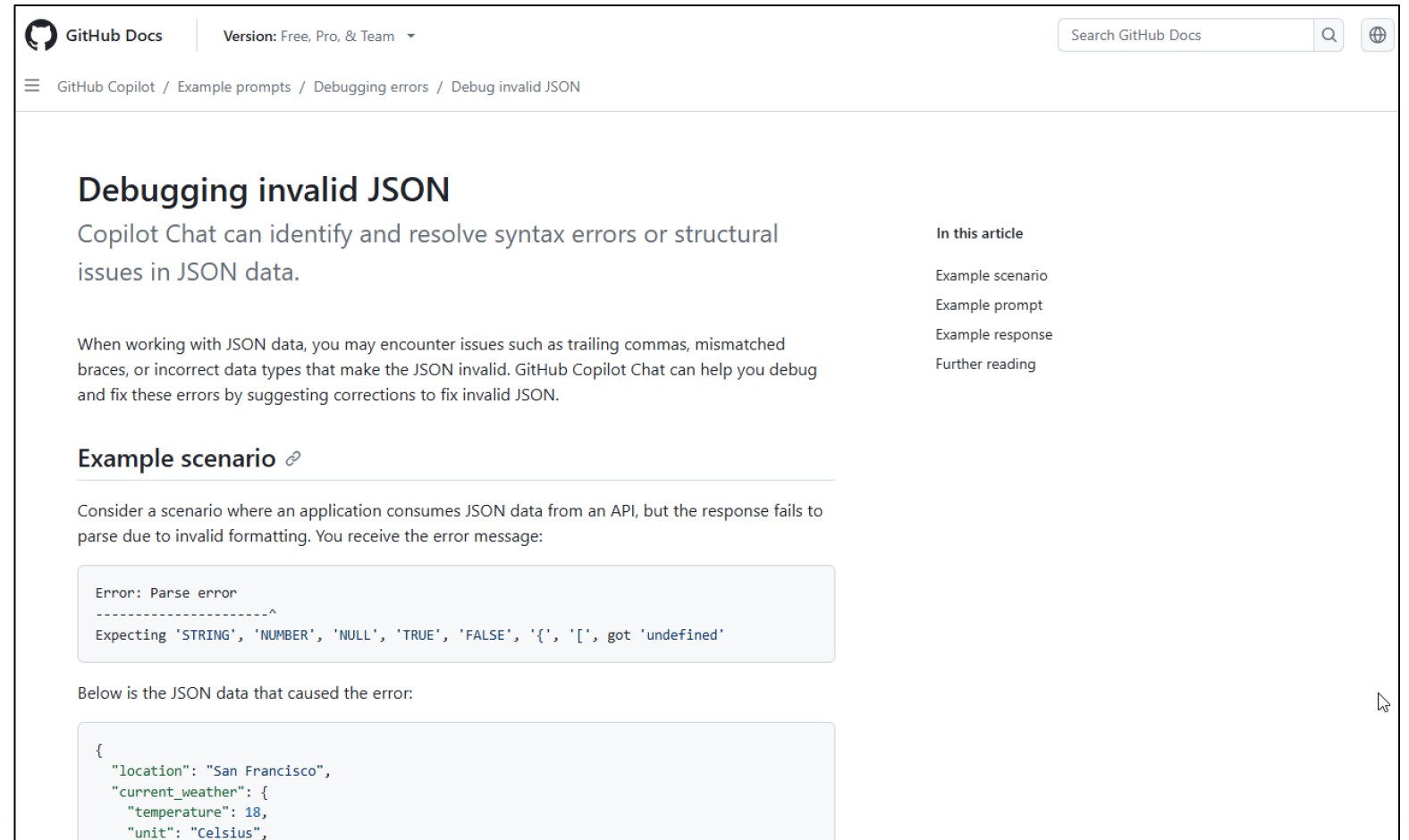
# Install GitHub Copilot for VS Code

- GitHub offers **free** Copilot integration for VS Code. Sign up for a GitHub account at <https://github.com> using a valid email address
- Log into your account, and go to <https://github.com/copilot>



# GitHub Copilot Documentation

For a user guide and examples of using GitHub Copilot, see <https://docs.github.com/copilot>



The screenshot shows the GitHub Docs interface. At the top, there's a header with the GitHub logo, 'GitHub Docs', and a version dropdown set to 'Free, Pro, & Team'. A search bar on the right contains the text 'Search GitHub Docs'. Below the header, a breadcrumb trail reads 'GitHub Copilot / Example prompts / Debugging errors / Debug invalid JSON'. The main content area is titled 'Debugging invalid JSON' and explains that Copilot Chat can identify and resolve syntax errors or structural issues in JSON data. It provides an example scenario where an API response fails to parse due to invalid formatting, showing an error message: 'Error: Parse error' and 'Expecting 'STRING', 'NUMBER', 'NULL', 'TRUE', 'FALSE', '{', '[', got 'undefined''. Below this, it states 'Below is the JSON data that caused the error:' and shows a JSON object: { "location": "San Francisco", "current\_weather": { "temperature": 18, "unit": "Celsius",

**Debugging invalid JSON**

Copilot Chat can identify and resolve syntax errors or structural issues in JSON data.

When working with JSON data, you may encounter issues such as trailing commas, mismatched braces, or incorrect data types that make the JSON invalid. GitHub Copilot Chat can help you debug and fix these errors by suggesting corrections to fix invalid JSON.

**Example scenario** [↗](#)

Consider a scenario where an application consumes JSON data from an API, but the response fails to parse due to invalid formatting. You receive the error message:

```
Error: Parse error
-----^
Expecting 'STRING', 'NUMBER', 'NULL', 'TRUE', 'FALSE', '{', '[', got 'undefined'
```

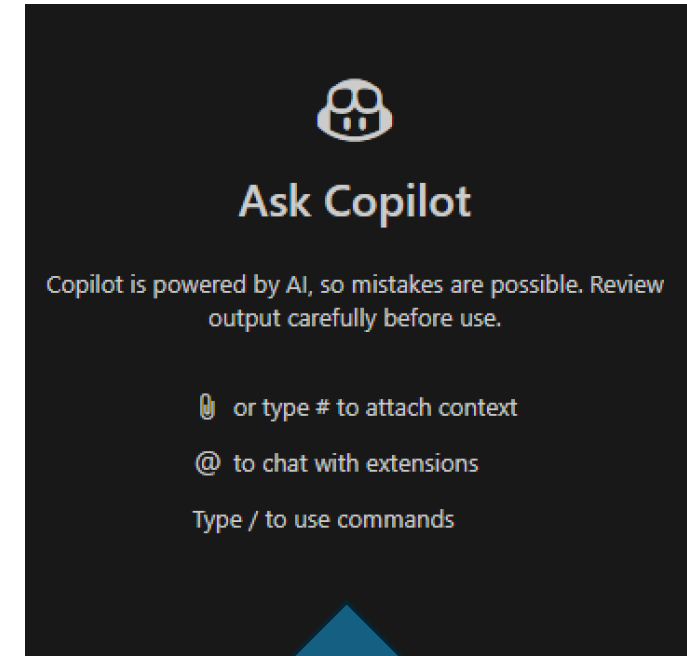
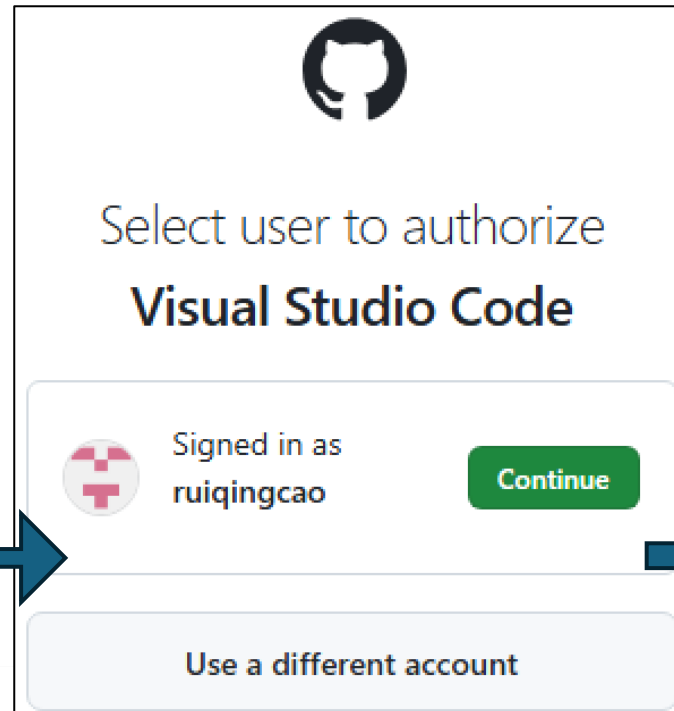
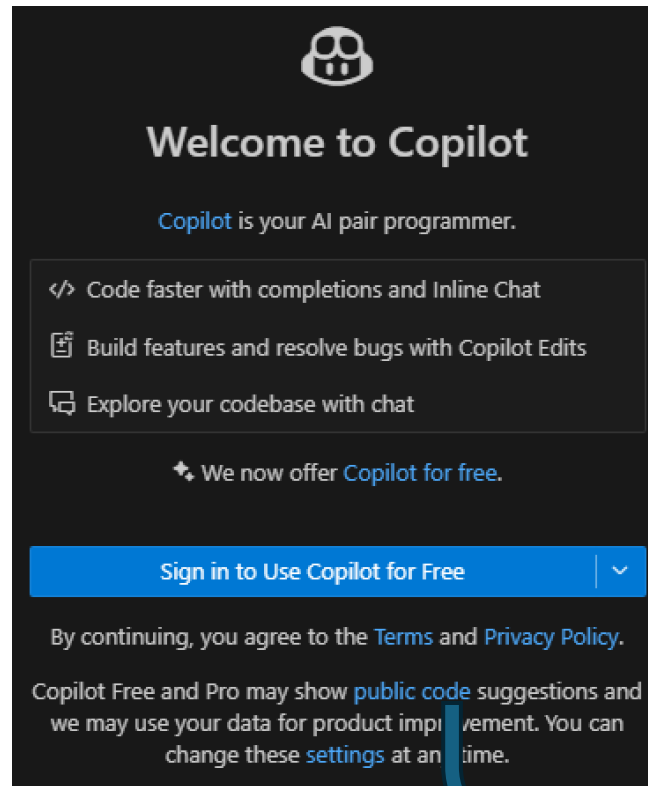
Below is the JSON data that caused the error:

```
{
  "location": "San Francisco",
  "current_weather": {
    "temperature": 18,
    "unit": "Celsius",
```

**In this article**

- Example scenario
- Example prompt
- Example response
- Further reading

# Sign Into GitHub from VS Code



# Checklist

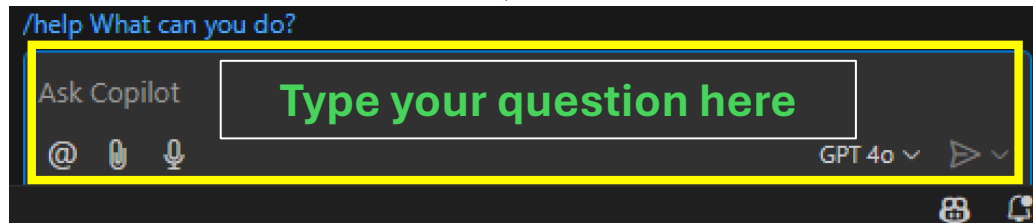
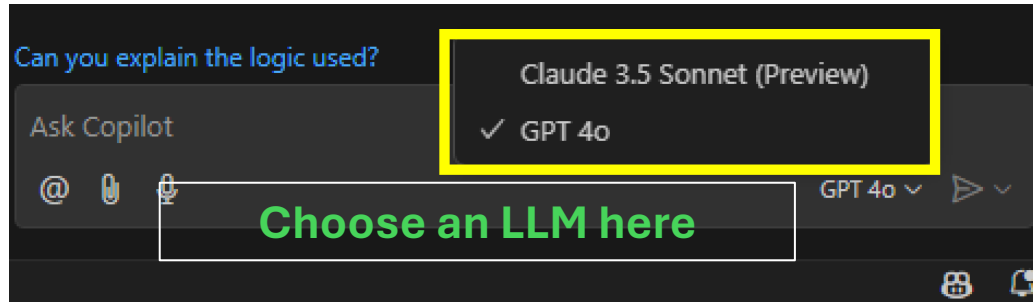
✓ **Anaconda + Jupyter Notebook**

✓ **Virtual Environment (venv1405)**

✓ **Visual Studio Code**

✓ **VS Code + GitHub Copilot Integration**

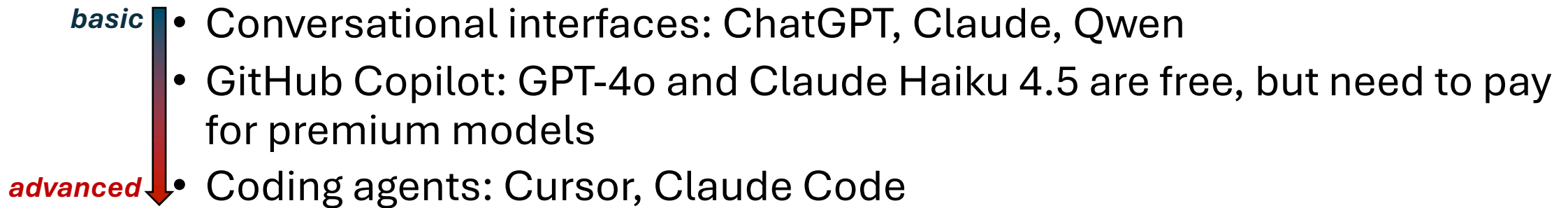
# Ask Copilot for Programming Help



# Use AI Tools for Programming Help

---

- Too many options these days...



- You can ask programming questions directly to an LLM

→ But make sure to double check AI-generated answers, and modify any AI-generated code as needed to ensure accuracy

- Prompting tips:

(1) Break a task into smaller steps (2) Provide clear instructions for each step

# GenAI Code Annotation Example

---

- Model: Claude Opus 4.5 (Anthropic)
- Prompting process (details in “få\_leksakslösenord.ipynb”):
  - Copy and paste the code from “få\_leksakslösenord.ipynb” into the Claude message box and press Enter
  - Claude will analyze the code and provide a detailed step-by-step explanation of each command
  - To request inline comments on the code, type: "Please add inline comments to the Python code" and press Enter
  - Claude will add comments to the code which explain the purpose of each line