# Data Science Setup

Congratulations, you've officially entered the world of data, algorithms, and the occasional *"why isn't this running?"* moment. Before you start training models, let's get your machine geared up for the adventure ahead.

## Version Control

Version control is essential for tracking changes in your code, collaborating with others, and ensuring you never lose your progress. Git and GitHub will be your best friends throughout the program!

### Create a GitHub Account

GitHub is a platform for hosting and sharing your code. You'll use it to collaborate on projects, submit assignments, and manage your repositories.

1. **Go to the GitHub Website**

- Open your browser and go to https://github.com.

2. **Click "Sign Up"**

- On the GitHub homepage, click the "Sign up" button at the top-right corner.

3. **Enter Your Details**

- Username: Choose a unique username.
- Email address: Provide a valid email.
- Password: Set a strong password.

4. **Complete the CAPTCHA**

- Solve the CAPTCHA to verify you're a human.

5. **Verify Your Email Address**

- GitHub will send you a verification email. Click the link to confirm your email.

6. **Start Using GitHub**

- Once your email is verified, you can start using GitHub, creating repositories, and more!

### Download Git

Git is the version control system that allows you to track changes in your code and push updates to GitHub.

### Mac

1. **Install Homebrew**

- Open Terminal (`cmd + space` and type 'terminal') and paste the following command to install Homebrew:

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/insta
```

2. **Install Git Using Homebrew**

- Once Homebrew is installed, use the following command to install Git:

```
brew install git
```

3. **Verify Installation**

- After installation, confirm that Git is installed by checking its version:

```
git --version
```

4. **Start Using Git**

- Git is now ready to use for version control and other projects!

### Windows

1. **Go to the Git Website**

- Open your browser and go to https://git-scm.com.

2. **Download Git for Windows**

- On the homepage, click the "Download for Windows" button. The installer will automatically download.

3. **Run the Installer**

- Once the installer is downloaded, run the `.exe` file and follow the on-screen instructions.

4. **Verify Installation**

- After installation, open the Command Prompt and type:

```
git --version
```

5. **Start Using Git**

- Git is now ready to use for version control and projects!

### (Optional) Download GitHub Desktop

GitHub Desktop provides a graphical interface for Git, making it easier to manage repositories without using the command line. While optional, it can be helpful for beginners.

### Mac

1. **Go to the GitHub Desktop Website**

- Open your browser and go to https://desktop.github.com.

2. **Download GitHub Desktop**

- Click the "Download for macOS" button to download the `.dmg` file.

3. **Install GitHub Desktop**

- Once the `.dmg` file is downloaded, open it and drag GitHub Desktop to your Applications folder.

4. **Sign In to GitHub**

- Open GitHub Desktop from your Applications folder. You'll be prompted to sign in with your GitHub account.

5. **Verify Installation**

- Once signed in, you can start using GitHub Desktop to manage your repositories easily.

### Windows

1. **Go to the GitHub Desktop Website**

- Open your browser and go to https://desktop.github.com.

2. **Download GitHub Desktop**

- Click the "Download for Windows" button to download the installer.

3. **Run the Installer**

- Once the `.exe` file is downloaded, run it and follow the on-screen instructions to install GitHub Desktop.

4. **Sign In to GitHub**

- After installation, open GitHub Desktop. You'll be prompted to sign in with your GitHub account.

5. **Verify Installation**

- Once signed in, you can start managing your repositories directly from the GitHub Desktop interface.

### (Optional) Join `GW-datasci` Organizational GitHub

Our program has a GitHub organization for students `GW-datasci` please be sure to join!

1. **Request an Invitation**

- A `GW-datasci` organization admin must send you an invitation. Send an email titled "GW-datasci Access" to twallett@gwu.edu
- Check your email or GitHub notifications for an invite link.

2. **Accept the Invitation**

- Open GitHub and sign in.
- Go to Notifications (bell icon) or visit GitHub's Organization Invitations.
- Click Accept Invitation to join the organization.

3. **Verify Access**

- After accepting, visit GW-datasci's GitHub page to confirm your membership.
- You should see repositories and projects available to you.

4. **Start Contributing!**

- You can now collaborate on GW-datasci projects through GitHub.

> `GW-datasci` Benefits
>
> By joining you will have access to shared repositories, course materials, templates and collaboration tools.

## Distribution

A distribution is a pre-packaged set of software and libraries that makes installation easy. This distribution will help you create environments to localize dependencies and avoid conflicts between different projects.

### Download Anaconda

Anaconda is an all-in-one Python and R distribution that includes essential libraries and tools for data science.

### Mac

1. **Go to the Anaconda Website**

- Open your browser and visit https://www.anaconda.com.
- Click "Download" and select the macOS installer (`.pkg` file).

2. **Install Anaconda**

- Open the downloaded `.pkg` file.
- Follow the on-screen instructions to install Anaconda.

3. **Verify Installation**

- Open Terminal and type:

```
conda --version
```

- If installed correctly, it will display the Anaconda version.

4. **(Optional) Set Up Conda in Terminal**

- Run the following command to ensure Conda is available:

```
conda init
```

5. **Start Using Anaconda**

- Open Anaconda Navigator from Launchpad or use Terminal to manage environments with Conda.

### Windows

1. **Go to the Anaconda Website**

- Open your browser and visit https://www.anaconda.com.
- Click "Download" and select the Windows installer (`.exe` file).

2. **Install Anaconda**

- Run the downloaded `.exe` file.
- Follow the installation wizard, selecting default settings.
- Ensure "Add Anaconda to my PATH environment variable" is checked (optional but recommended).

3. **Verify Installation**

- Open Anaconda Navigator from the Start menu.
- Or open Command Prompt and type:

```
conda --version
```

- If installed correctly, it will display the Anaconda version.

4. **(Optional) Set Up Conda in Command Prompt**

- To ensure Conda is available in Command Prompt, run:

```
conda init
```

5. **Start Using Anaconda**

- Open Anaconda Navigator or use conda commands to manage environments and packages.

## Integrated Development Environments (IDEs)

An IDE is where you'll write, test, and debug your code. Here at the Data Science program we recommend VS Code or PyCharm.

> ⚠ **Pick Your Coding Home Wisely!**
>
> Your IDE is like your favorite coffee shop—you'll spend *a lot* of time there. So choose one that feels comfortable to you!

### Download an IDE

### VS Code

1. **Go to the VS Code Website**

- Open your browser and visit https://code.visualstudio.com.
- Click "Download for Windows" or "Download for macOS" based on your system.

2. **Install VS Code**

- Windows: Run the downloaded `.exe` file and follow the installation instructions.
- macOS: Open the `.zip` file and move Visual Studio Code to the Applications folder.

3. **Verify Installation**

- Open VS Code and check if it runs correctly.

### PyCharm

1. **Go to the PyCharm Website**

- Visit https://www.jetbrains.com/pycharm/download.
- Select either Community Edition (free) or Professional Edition (paid).
- Click Download for Windows or macOS.

2. **Install PyCharm**

- Windows: Run the `.exe` installer and follow the instructions.
- macOS: Open the `.dmg` file and drag PyCharm to the Applications folder.

3. **Verify Installation**

- Open PyCharm and create a new Python project to ensure it's working.

### Connect IDE to GitHub Account

### VS Code

1. **Install GitHub Extension in VS Code**

- Open VS Code.

- Go to Extensions (`Ctrl+Shift+X` / `Cmd+Shift+X`).
- Search for "GitHub Pull Requests and Issues" and install it.

2. **Sign In to GitHub**

- Open VS Code Command Palette (`Ctrl+Shift+P` / `Cmd+Shift+P`).
- Search for "GitHub: Sign in to GitHub" and select it.
- Follow the on-screen instructions to authenticate with GitHub.

### PyCharm

1. **Sign In to GitHub from PyCharm**

- Open PyCharm and go to "File > Settings (Windows)" / "PyCharm > Preferences (Mac)".
- Navigate to "Version Control > GitHub".
- Click Add Account and sign in using your GitHub credentials or a personal access token.

### Extensions/Plugins

> Extensions (or plugins) are add-ons that enhance functionality, such as adding language support, debugging tools, or AI-powered coding assistance. They help customize the IDE to improve productivity, automation, and development workflows.

### VS Code: Extensions

- **Python**
  Adds Python support, IntelliSense, debugging, and Jupyter Notebook functionality.
- **R**
  Provides language support for R, including syntax highlighting and code completion.
- **Pylance**
  Offers advanced linting, type checking, and autocomplete features for Python.

- **Git**
  Provides Git integration for version control within VS Code.
- **GitHub**
  Allows you to manage and interact with GitHub repositories directly from VS Code.
- **Code Runner**
  Enables you to run code snippets for multiple languages, including Python, directly within VS Code.
- **Jupyter**
  Lets you run and edit Jupyter Notebooks directly in VS Code.
- **EditCSV**
  Allows for easy editing of CSV files directly within VS Code.
- **HTML Preview**
  Provides an HTML preview of your code directly in the editor.
- **Quarto**
  Provides support for creating and rendering Quarto documents within VS Code.
- **Remote - SSH**
  Allows you to open remote folders and develop on remote machines over SSH.
- **TensorBoard**
  Enables the viewing of TensorFlow logs directly within VS Code.
- **SVG Preview**
  Lets you preview SVG files within VS Code.
- **Copilot (Paid)**
  A paid extension by GitHub that provides AI-powered code suggestions and autocompletions.
- **Markdown All in One**
  Offers a comprehensive suite of features for editing and previewing Markdown files.
- **Docker**
  Adds Docker support to VS Code, allowing you to manage containers and images.

### PyCharm: Plugins

- **Key Promoter X**
  Displays keyboard shortcuts every time you use the mouse, helping you learn and use shortcuts more efficiently.
- **Markdown Navigator**
  Provides enhanced markdown editing and previewing support.

- **R Plugin**
  Adds support for R scripts and notebooks, allowing you to work with R code directly in PyCharm.
- **.env Files Support**
  Loads environment variables from `.env` files, helping you manage sensitive information like API keys and credentials.
- **Pandas Helper**
  Provides quick previews, descriptions, and structure analysis for Pandas DataFrames, making it easier to inspect data.
- **DeepBugs**
  Uses machine learning to detect common Python coding mistakes and bugs in your code.
- **Docker**
  Adds Docker support to PyCharm, allowing you to manage containers, images, and other Docker resources directly from the IDE.
- **Jupyter**
  Provides full Jupyter Notebook support in PyCharm, including the ability to run and edit notebooks.
- **GitHub**
  Integrates GitHub repositories and allows you to work with your projects directly from PyCharm.
- **Python Scientific**
  Adds support for scientific libraries like NumPy, SciPy, and Matplotlib, helping you visualize and analyze data.
- **Tabnine**
  An AI-powered code completion tool, improving your coding speed by suggesting relevant completions.
- **Python Docstring Generator**
  Helps you generate consistent Python docstrings with a single shortcut, saving time on documentation.
- **Database Navigator**
  Provides easy database connection, navigation, and management capabilities within PyCharm.
- **Flake8**
  Adds linting support for Python, helping you maintain clean and readable code by checking for errors and style issues.

### Debugger

A debugger is a tool that helps find and fix errors in code by allowing step-by-step execution, breakpoints, and variable inspection. It makes troubleshooting easier by showing how code runs in real time.

### VS Code

1. **Set Breakpoints**

- Click on the gutter (the area next to the line numbers) to set breakpoints. A red dot will appear where the breakpoint is set.

2. **Open the Debug Panel**

- Press `Ctrl+Shift+D` (Windows/Linux) or `Cmd+Shift+D` (macOS) to open the Debug Panel.

3. **Configure the Debugger**

- If this is your first time using the debugger, you may need to create a launch configuration.
- Click on "create a launch.json file" in the Debug Panel.
- Select Python from the dropdown and VS Code will generate the default configuration for you.

4. **Start Debugging**

- Click the green play button or press `F5` to start debugging. Your code will run, and the debugger will stop at any breakpoints.

5. **Use Debugging Controls**

- Step Over (`F10`): Executes the next line of code without stepping into functions.
- Step Into (`F11`): Steps into the code inside a function if the line contains a function call.
- Step Out (`Shift+F11`): Steps out of the current function and back to the caller.
- Continue (`F5`): Continues running the program until the next breakpoint.
- Variables: View and edit variables in the Debug panel.
- Watch: Add expressions to track their values in real-time.

### PyCharm

1. **Set Breakpoints**

- Click on the gutter (next to line numbers) to set breakpoints. A red dot will appear at the breakpoint.

2. **Open the Debug Tool Window**

- To open the Debug Tool Window, go to View > Tool Windows > Debug, or use the shortcut `Alt+5` (Windows/Linux) or `Cmd+5` (macOS).

3. **Start Debugging**

- Click on the bug icon or press `Shift+F9` to start debugging. Your program will run, and it will stop at breakpoints.

4. **Use Debugging Controls**

- Step Over (`F8`): Executes the next line of code without stepping into functions.
- Step Into (`F7`): Steps into the function being called.
- Step Out (`Shift+F8`): Steps out of the current function.
- Resume Program (`F9`): Continues execution until the next breakpoint.
- Variables: View and edit variables in the Variables tab.
- Watches: Add custom expressions to evaluate while debugging.

5. **View and Edit Variables**

- In the Debug tool window, under the Variables tab, you can inspect the values of variables in your current scope.
- You can also right-click on variables to change their values at runtime, allowing you to test different scenarios.