LESSON 02

STARTING WITH PYTHON NOTEBOOKS AND COLAB

Programs in this course use the Python programming language. Python is the most widely used and supported language in many areas of computer science, including machine learning. There is roughly one Python program for you to try for each lesson of this course. Even if you've never written programs in Python—or any language—you can still run these programs to get a feel for what they do.

This Course's Approach

The approach that has been laid out for users of this course is

1. browser-based programs
2. using Python notebooks
3. hosted on GitHub that
4. run through Google Colaboratory (Colab).

Python notebooks offer a mode of interacting with the complexities of machine learning programs that is relatively easy for beginners and empowering for everyone.

Here's what you should know about each of the four components of this approach:

1. It's recommended that you run the Python programming examples from your web browser. Doing so allows you to avoid having to install the software on your computer. In addition, you get access to powerful server machines for running the code, which can be much faster than machine learning programs run on your own computer. Plus, making use of professionally maintained servers can save you some other headaches, such as needing to store and manage large data files or reinstalling libraries after software upgrades.
2. This course has been designed so that you are working with Python code by way of a special file format known as an interactive Python notebook, also called a Jupyter notebook.* The extension for an interactive Python notebook file format is `.ipynb`, and that's the main format for files provided for this course. The advantage of an interactive `.ipynb` file over a static `.py` Python file†—often used for Python programs—is that an interactive Python notebook integrates code, documentation, and the result of running the code all in one place.

---

* The IPython project, which began in 2001, adds interactive features to basic Python programming. Since 2014, interactive Python has also served as the kernel for Project Jupyter to extend the same interactive notebook approach across many other programming languages.

† Python notebook files are not directly usable as plain Python program files in the `.py` format. These notebooks cannot be usefully opened and edited as plaintext in a text editor like Notepad. The Python notebooks that have been provided for this course were not designed to run top to bottom as static `.py` files.

3. The Python notebooks are hosted and shared with you through a widely used code-sharing service called GitHub. You don't need to go to GitHub directly from a browser search engine. Instead, you can get there from this link, which takes you to a landing page that has been set up for this course:

   www.TheGreatCourses.com/MachineLearning

4. The browser-based interpreter for Python that's recommended for these notebooks is a free service called Colaboratory, or Colab, provided by Google to anyone who has a Google account. The notebooks can also run from a browser on the Jupyter website, or even on your local computer if you download and install the Jupyter environment.

Starter Example

To work through a starter example in Colab, visit this link in your browser:

https://colab.research.google.com/github/mlittmancs/great_courses_ml/blob/master/L02.ipynb

The link will take you to a file on GitHub, where you will see a "Hello, World!" program in Python.

Colab is optimized for only a few browsers—typically Firefox, Safari, and Chrome—so if the link fails or you have difficulty following any of the later steps, make sure you are using one of those browsers. You can also check the Great Courses landing page.

If you scroll through the file, you can see the code and its output from the last time it was run before the file was posted.

Anyone can see a file in Colab, but to run files in Colab, you'll need to be logged into a Google account. Once you are logged in, you can run code by clicking on the triangular play symbol on the boxes.

However, running the code out of order might result in errors if the code refers to variables or function names that have not been defined yet.

You can also edit the code and run it; you are not restricted to only running the provided code. Just click on the code and edit. Your changes won't be saved, but they will run nevertheless.

Finally, if you want to keep your changes, you can save your own personal copy of the file, preserving any edits that you made.

But before you can save, Colab insists that you make your own copy. By doing so there's no way anything you do in your own copy can ever affect the original hosted file. If you return to the hosted file, the original version will always be what appears.

The options for saving your own copy are under the File menu.

- You can download the file to your Google Drive by clicking "Save a copy in Drive". Then, when you click on the file in Google Drive, it will start up a new Colab session for you.
- Or you can save your copy of the file to GitHub in a few different ways if you register for a free GitHub account: You can click "Save a copy as a GitHub Gist," which is GitHub's simpler format for saving single snippets of code; or you can click "Save a copy in GitHub," which means you have established your own code repository, where your files can access more features.

After saving in Drive, GitHub Gist, or GitHub, you can use the plain Save command when you make updates.

When you run the code, the first block contains the definition of a function called `helloworld`. Click on this block to reveal the play symbol in the upper left. Click on that play symbol.

Google Colab will warn you that the file was "not authored by Google." But that's ok, because it was authored for this course.

If you read the fine print on the warning, it says the code "may request access to your data stored with Google, or read data and credentials from other sessions" and that you should "review the source code before executing this notebook."

This sort of notice is pointing out—correctly—that software automates actions on your behalf, and you may not want those actions taken.

In the case of Colab, the code is not running on your computer and does not have access to the files on your computer. But it does have access to your Google account. However, even if you were to download and run a malicious notebook, that code would not automatically have access to your account. It would request access.

Before you allow any request for access, ask yourself these three questions:

1. **Is the code coming from someone you trust?** In this case, the code was specifically authored for your use in this course.
2. **If the code asks for access to resources, do those resources line up with what you expect the code to do?** If a program asks for access to all of your files just to show you a video, you should be suspicious. In this case, the code is asking for access to your account with Google because that's how Google validates, and places an upper limit on, your free use of their machines. If the code ever asks for deeper access, make sure that additional access lines up with what the code is supposed to do.
3. **Is this the safest alternative for you to get what you need**? If you have another way to get the job done without needing to trust a third party, that would be preferable. Also, consider whether you actually need what you thought you needed.

If you answer yes to all of these questions, you can feel comfortable clicking "Run Anyway" to run the code—even without attempting to "review the source code."

You might also be asked about resetting runtimes. *Runtime* is Colab's word for the connection between the notebook and one of their servers. Resetting a runtime will make it forget any calculations it has done on this notebook. You won't lose the code, just the value of any variables that have been defined since you opened the notebook. You can just approve resetting all runtimes and then run the code again.

Now that you've clicked on the play symbol, the program code will execute. In this case, it doesn't do anything obvious, except that the circle around the play symbol indicates activity. But behind the scenes, it is running the block that causes the `helloworld` program to be defined.

Now look at the second block, where it says `helloworld(4)`. Under that, you can see the output: `hello! hello! hello! hello!` (in other words, four hellos). That's the result already obtained from running `helloworld(4)` when the notebook was set up.

Now run it just to check. Click on the triangular play symbol next to the command. The hellos briefly disappear and then reappear. That small change tells you that the code is live.

To confirm that you're not merely looking at a static web page, click on the `helloworld(4)` and change the four to some other number, such as 10. Now click play and you should see 10 hellos appear in response.

If you want to add your own coding block to the document, here are a few of the many options:

- You can click `+Code` in the top-left section of the screen.
- You can click on any empty block next to a pair of brackets: []. Start typing in the block and you are editing a program or command of your own.
- You can add code to an existing block. For example, you can see where `4+5` was typed and it returned `9`. You can type in any Python command next to `4+5` and then hit the play symbol to execute it. Or you can hit Shift+Enter—a keyboard shortcut‡ for running the current block—to run the code without having to take your hands off the keyboard.

If there are errors in what you type, you'll see an error message appear in the box below. Otherwise, it'll run what you ask.

Python notebooks run only the blocks you specify, in whatever order you run them. Again, it's usually a good idea to run the blocks in order—from top to bottom. § If you had run the helloworld(4) block before running the helloworld definition block, it would give you an error pointing out that helloworld is not yet defined.

Python programs can produce text or graphics or sometimes basic user interactions. If the text output is sufficiently long, the output box will be scrollable. That lets you look at all the output or just scroll to the next block.

If you run a program that does not terminate, you might have to click the stop symbol—which appears in place of the play symbol, inside a spinning circle—while a block is running.

Just remember that if you want to save any of your changes, you have to make your own copy and save that copy elsewhere. Here are some options:

- Save to your computer's hard drive by clicking on the File menu in the upper left of the screen and then selecting "Download .ipynb".
- Save to your personal GitHub account.
- Save to a personal account on Google Drive.

‡ Under the Tools menu, there's a list of other "Keyboard shortcuts".

§ Running blocks out of order sometimes does come in handy—for example, if you're in the middle of developing a new program.

For beginners, the easiest option may be to save to your own account on Google Drive. When you save, a folder called something like "Colab Notebooks" should be automatically created to hold whatever you save from Colab to your Google Drive. Clicking on files there later will return you to your version. If necessary, manually select for it to be reopened with Colab and rerun the blocks to get back to where you left off.

If a run ever hangs, or gets too out of hand, you should be able to just kill the browser tab that Colab is running in and then open a new tab whenever you're ready to try some more.

All the programs for this course are available at the Great Courses landing page. For some lessons, you'll see that also included is a file with "aux" in the name, where you can run auxiliary programs that were used but not displayed in the video lesson.

Here are two work-arounds if you are unable to access files via the main link to the landing page address:

1. You could try searching the GitHub site for the string "great-courses-ml". In that case, you'll need to go through the search results GitHub presents at the time to find the correct repository. The page you get will have links to all of this course's program files.
2. If the GitHub collection of files ever becomes unusable, a zipped collection of all instructional program files used in the course is available from the Great Courses landing page.¶

In Colab, the exclamation mark symbol (**!**) at the front of a command results in running an operating system command on the Colab computer. You will occasionally use this feature to fetch files and install libraries.

When running this kind of block, you'll typically get some commentary from the download process in your output window, which you can skip over. But in this case just below the commentary, you'll see a friendly ASCII picture message at the end.

These tools, and the experience you gain with them, will let you dive as deep as you want into the world of machine learning.

¶ Due to intellectual property considerations, some of the data files you are permitted to use as an individual accessing from a site like GitHub are not hosted on The Great Courses.

Much of machine learning is less about hand-coding programs from scratch and more about making use of the variety of resources that have already been created. So this course was designed to strike a balance that exposes you to some programming details while always trying to keep a bigger picture in view.

Whenever you want to explore more details, there are plenty of online resources with further information about working with Python, its libraries, and Jupyter notebooks.

The main two libraries you'll be using in this course are Scikit-learn for classic machine learning programs and Keras for deep learning programs, with others brought in as needed. There's a vibrant and active user community associated with Python, Colab, and the various machine learning libraries.

Try It Yourself

Follow along with the video lesson via the Python code:

L02.ipynb

QUESTIONS

1. Run `helloworld(20)`.
2. Make and run a `goodbye(x,y)` program that writes `x` copies `hello` followed by `y` copies of `goodbye`.

Answers on page 182