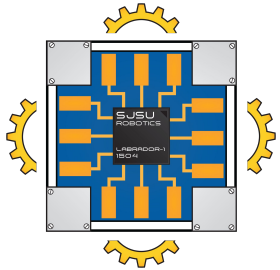


SJSU Robotics Team

Intelligent Systems Division
Summer 2020

Version Control with `git` and GitHub

Summary

Version control allows you to track and rollback changes to your code repository. Pushing your code repository to GitHub allows you to share your code with others.

Here are examples for using git to version control your work in the curriculum repositories.

- ros-curriculum
- ml-curriculum

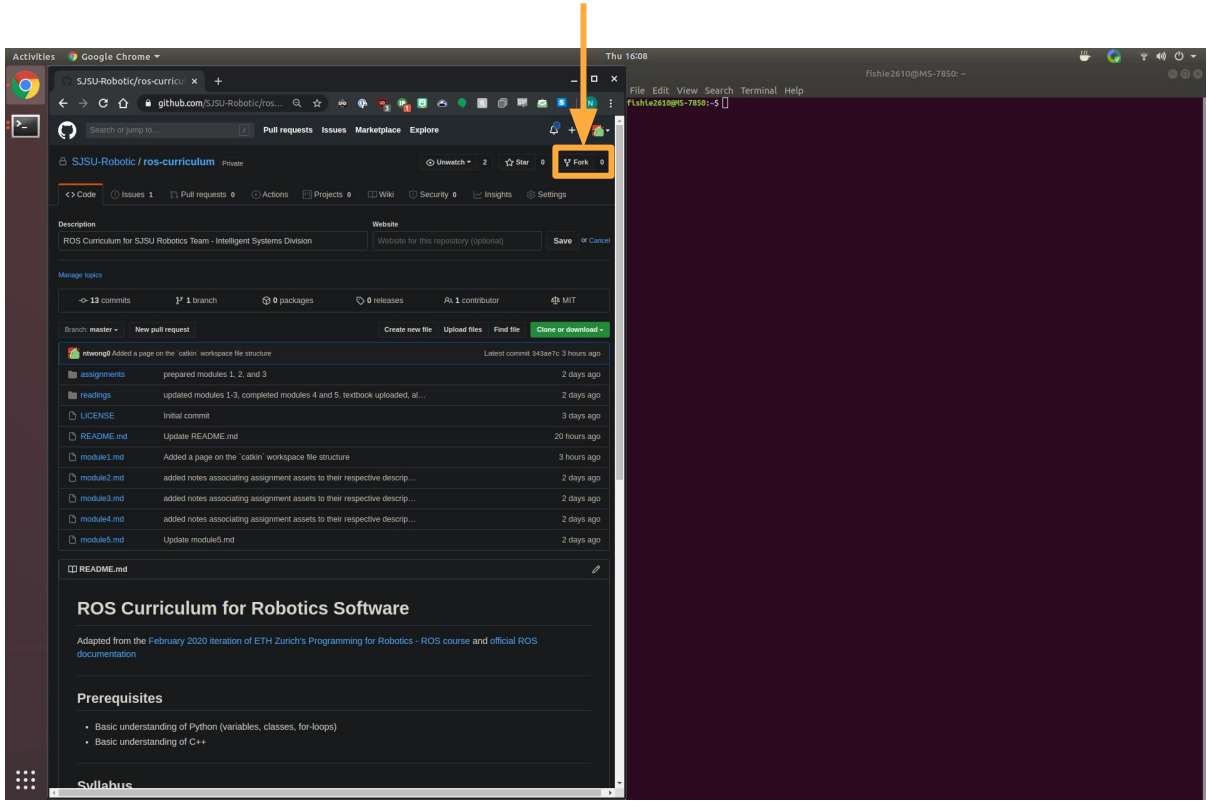
ros-curriculum

ros-curriculum

You can use the course repo as a container for your ROS workspace.

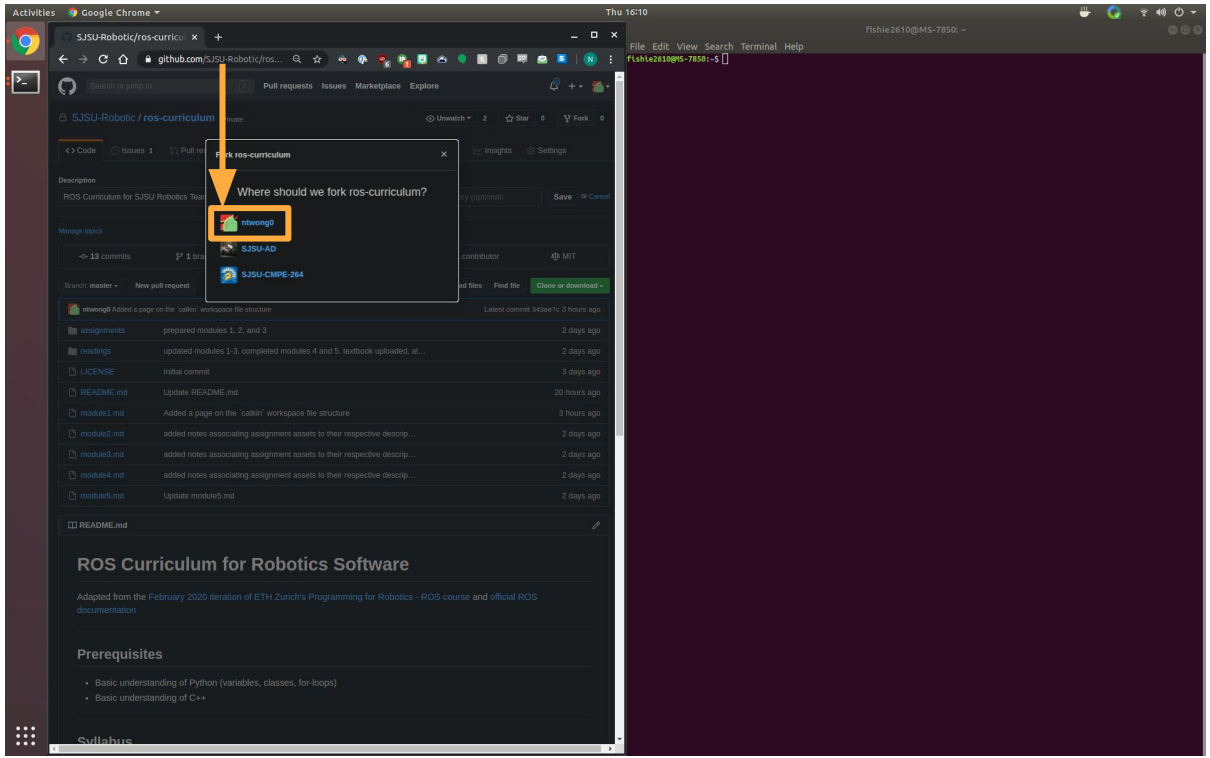
First, we start with the repo page on the left, and a terminal window on the right.

[Create a fork of the repo](#) to your GitHub account by selecting the [Fork] button.



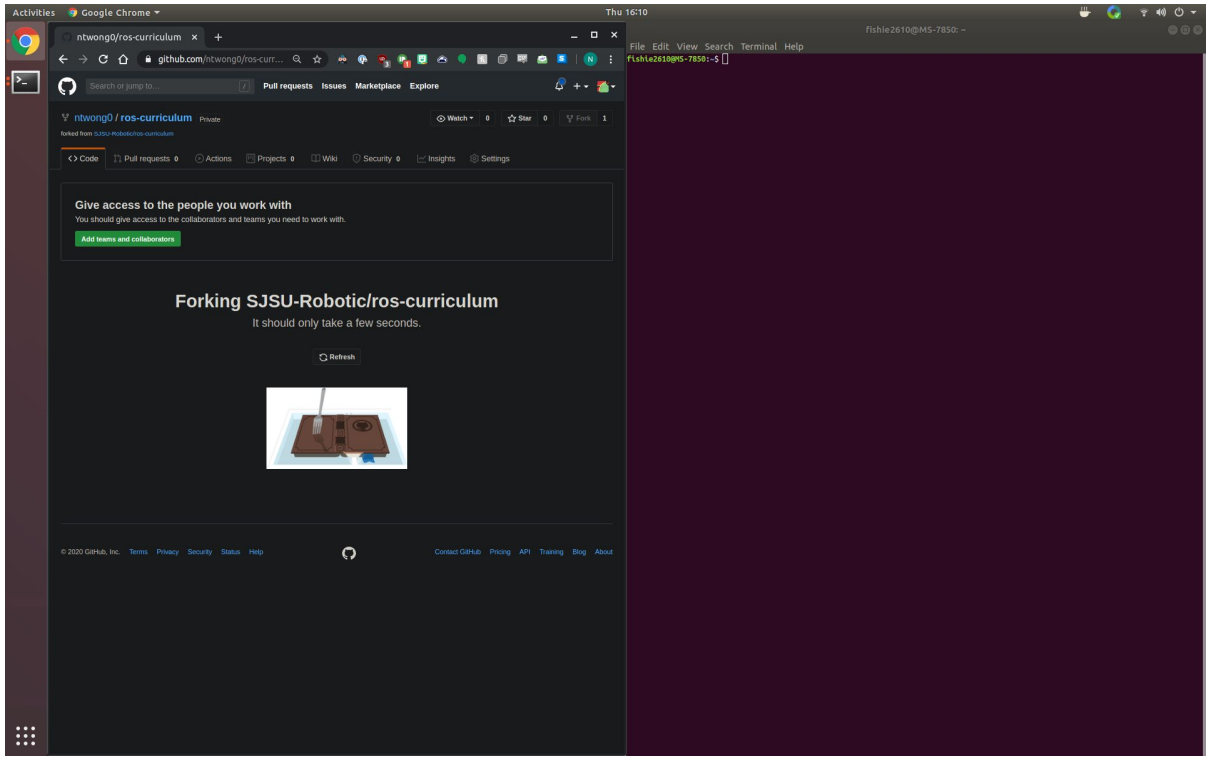
ros-curriculum

Select your account as the fork destination.



ros-curriculum

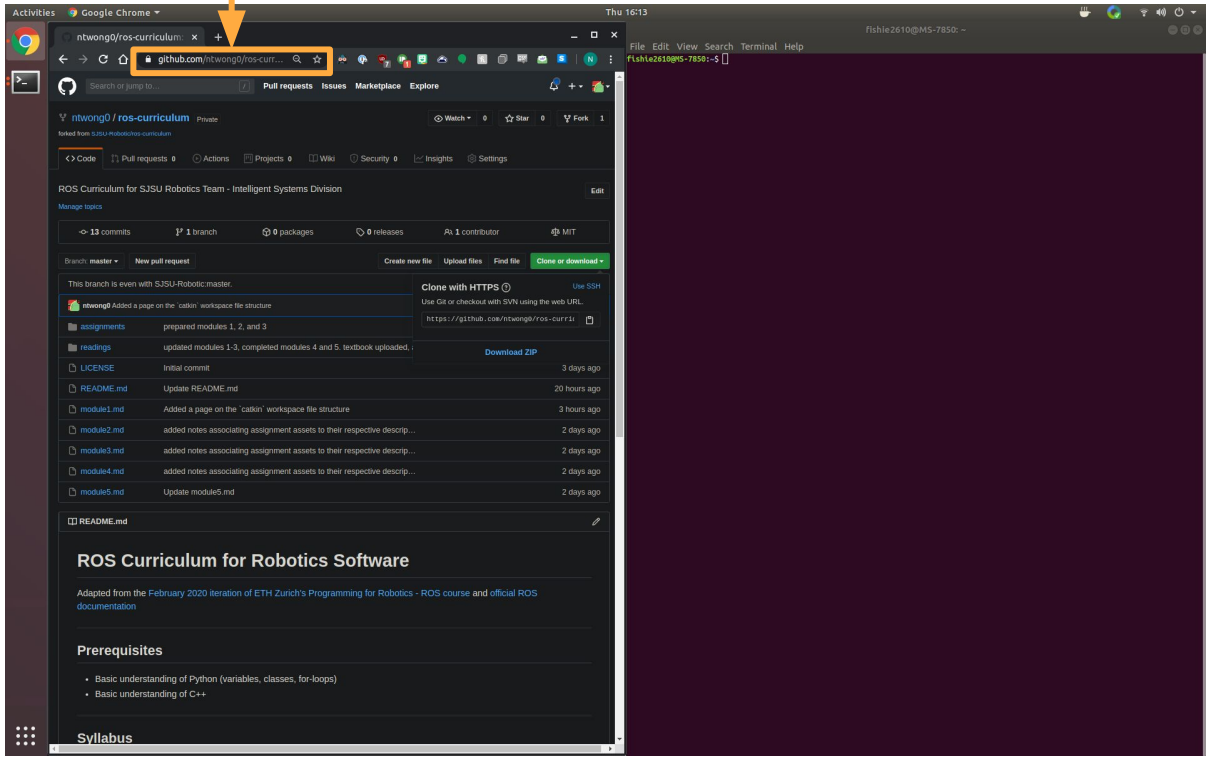
Forking will take a few seconds.



ros-curriculum

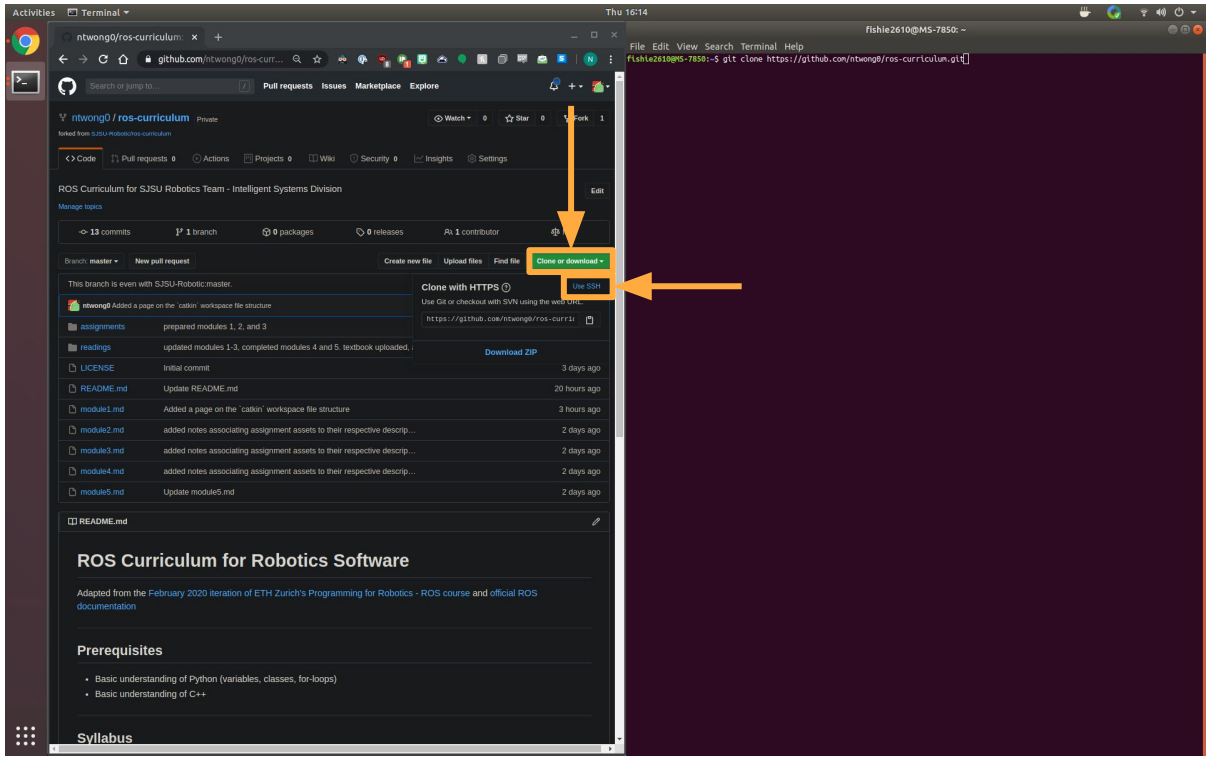
The fork is now complete, and it is now accessible at

github.com/<yourAcct>/ros-curriculum



ros-curriculum

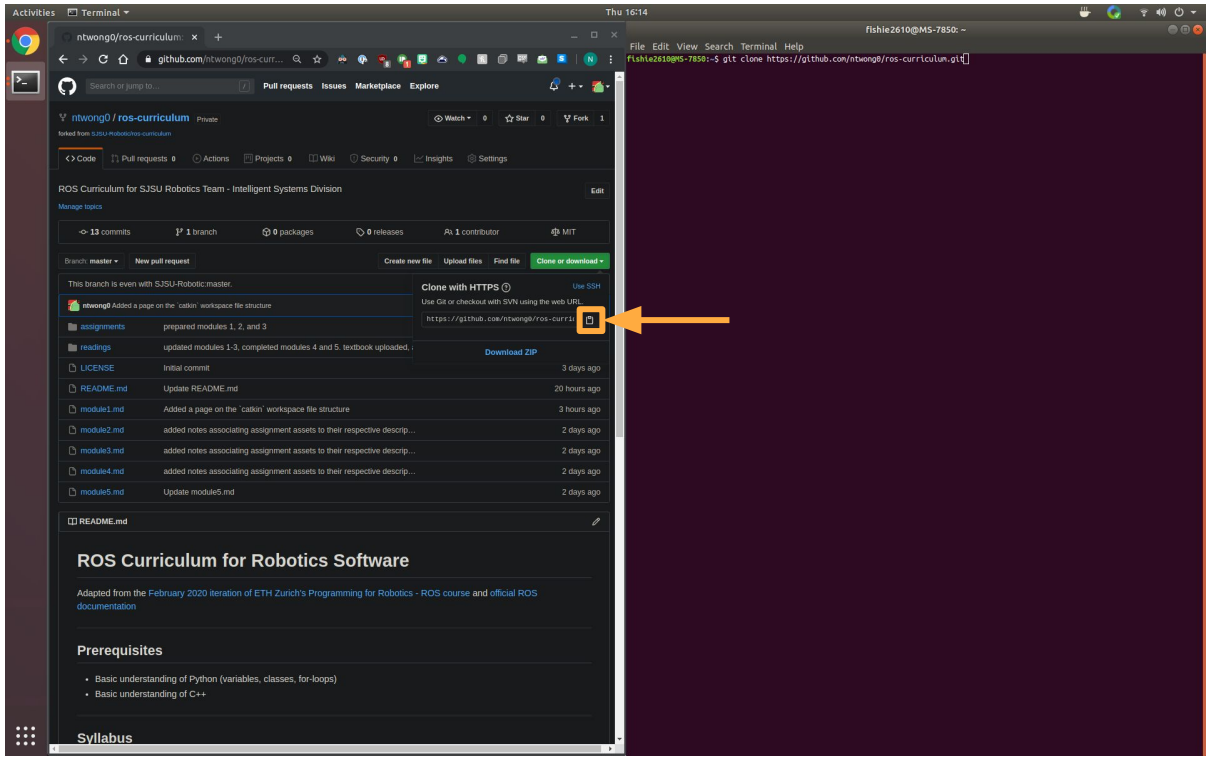
Select [Clone or download], then select [Use HTTPS] to reveal the URL.



ros-curriculum

Select [Clone or download], then select [Use HTTPS] to reveal the URL.

Copy the URL to your clipboard.



ros-curriculum

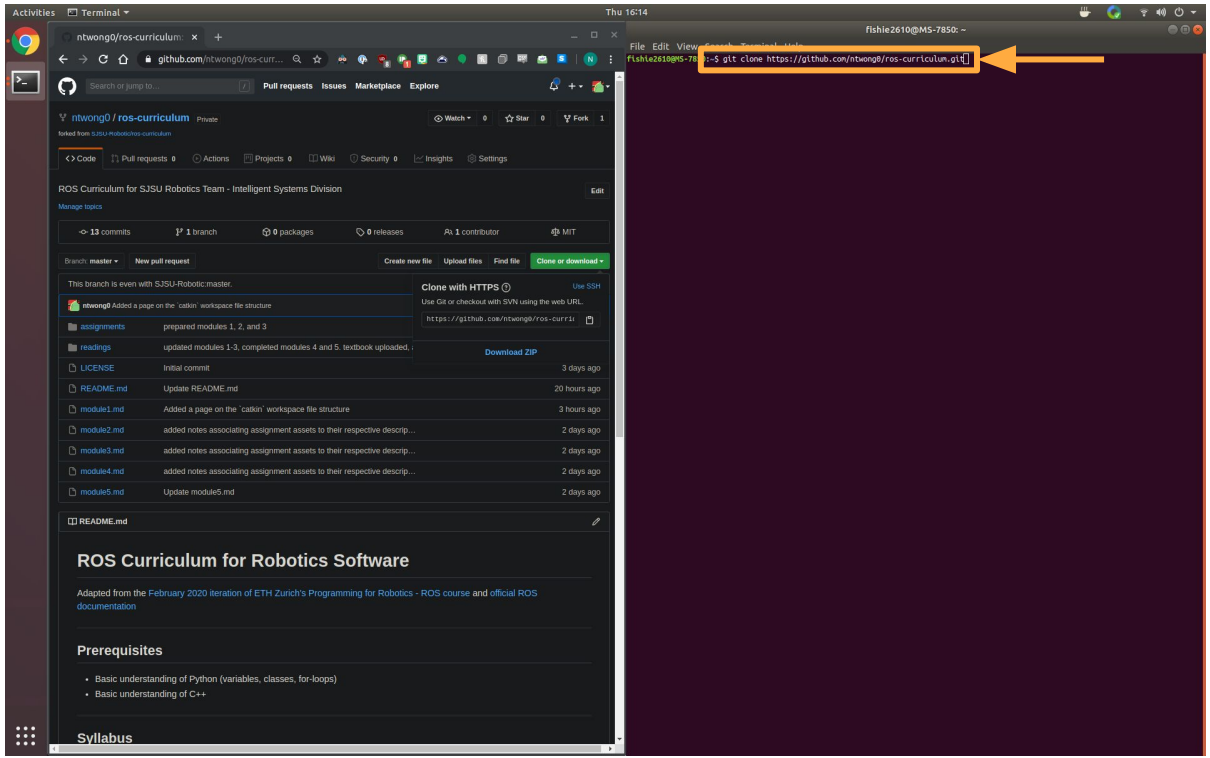
Select [Clone or download], then select [Use HTTPS] to reveal the URL.

Copy the URL to your clipboard.

In the terminal window, type

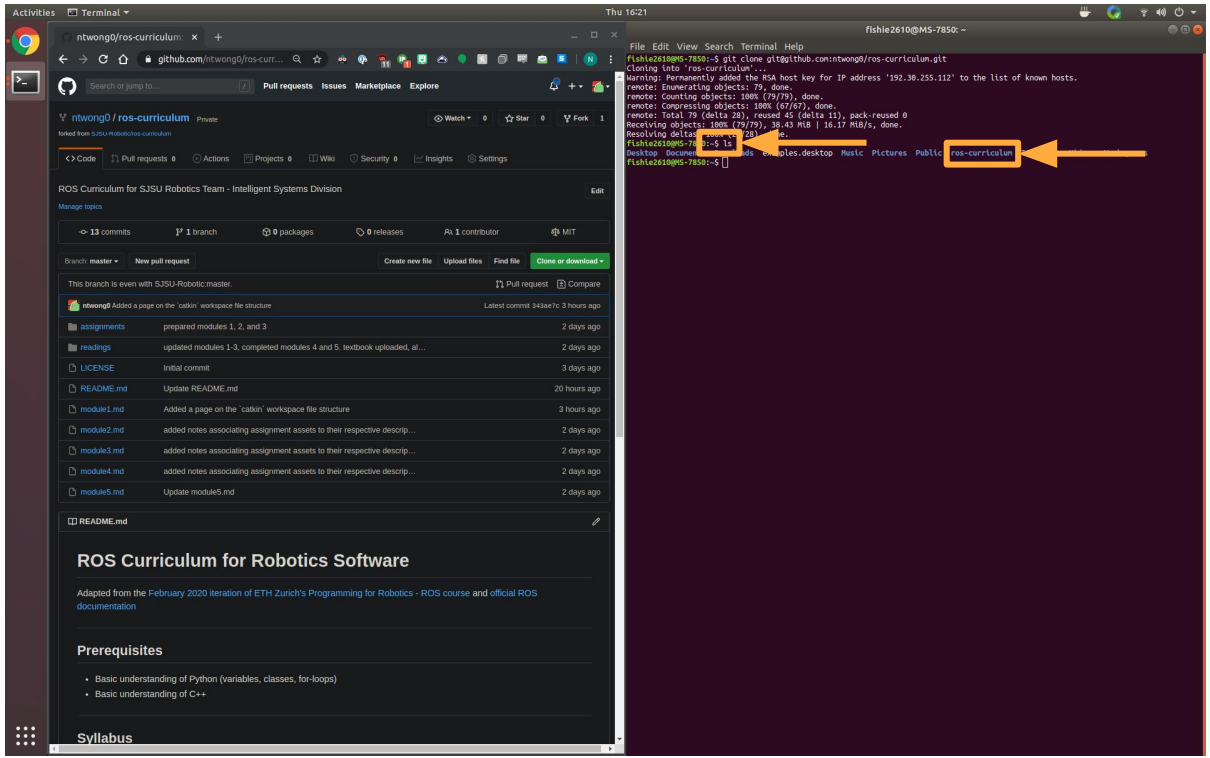
```
git clone
```

then paste the URL, and hit enter.



ros-curriculum

The repo is cloned into a folder at the current working directory. If you use the `ls` command to list the working directory's contents, the `ros-curriculum` folder is listed

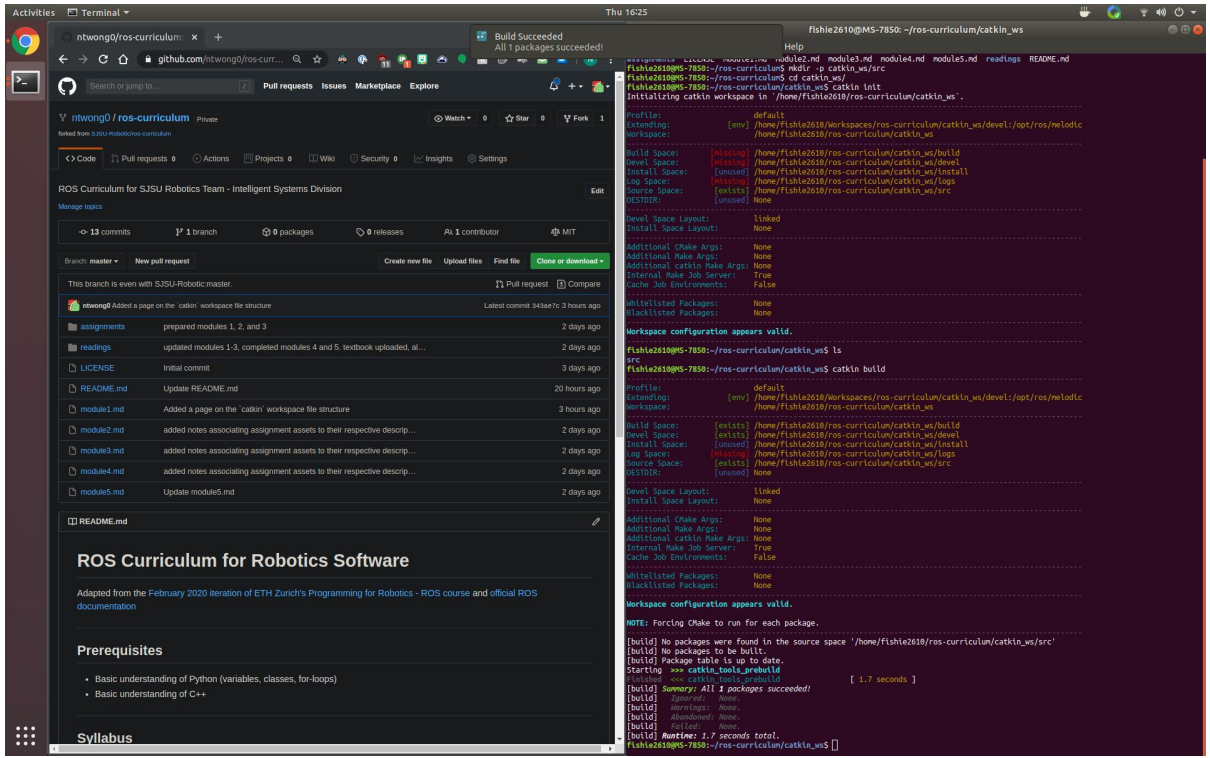


ros-curriculum

If you haven't created the workspace, you can do that now.

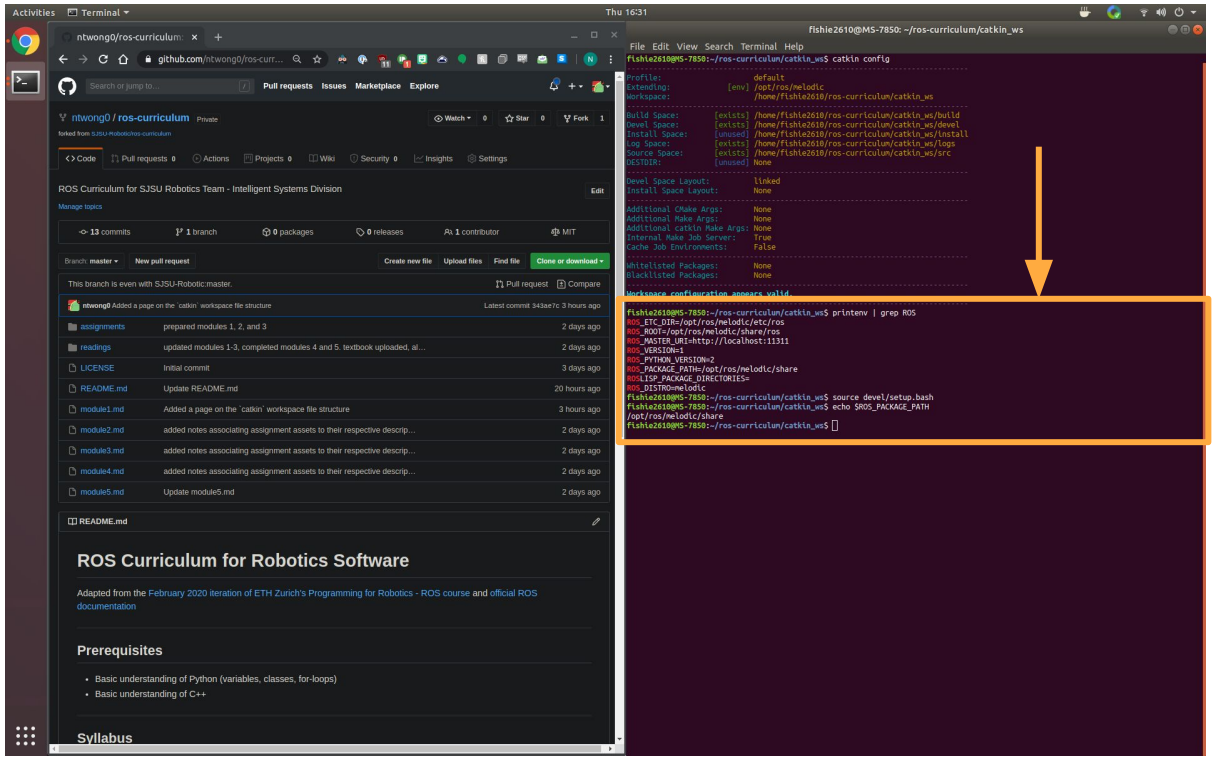
Here, we follow the example from the [ROS tutorial for configuring your ROS environment]...

...with a twist. In this case, we are using `catkin-tools` instead of `catkin_make`/`catkin_create_pkg`



ros-curriculum

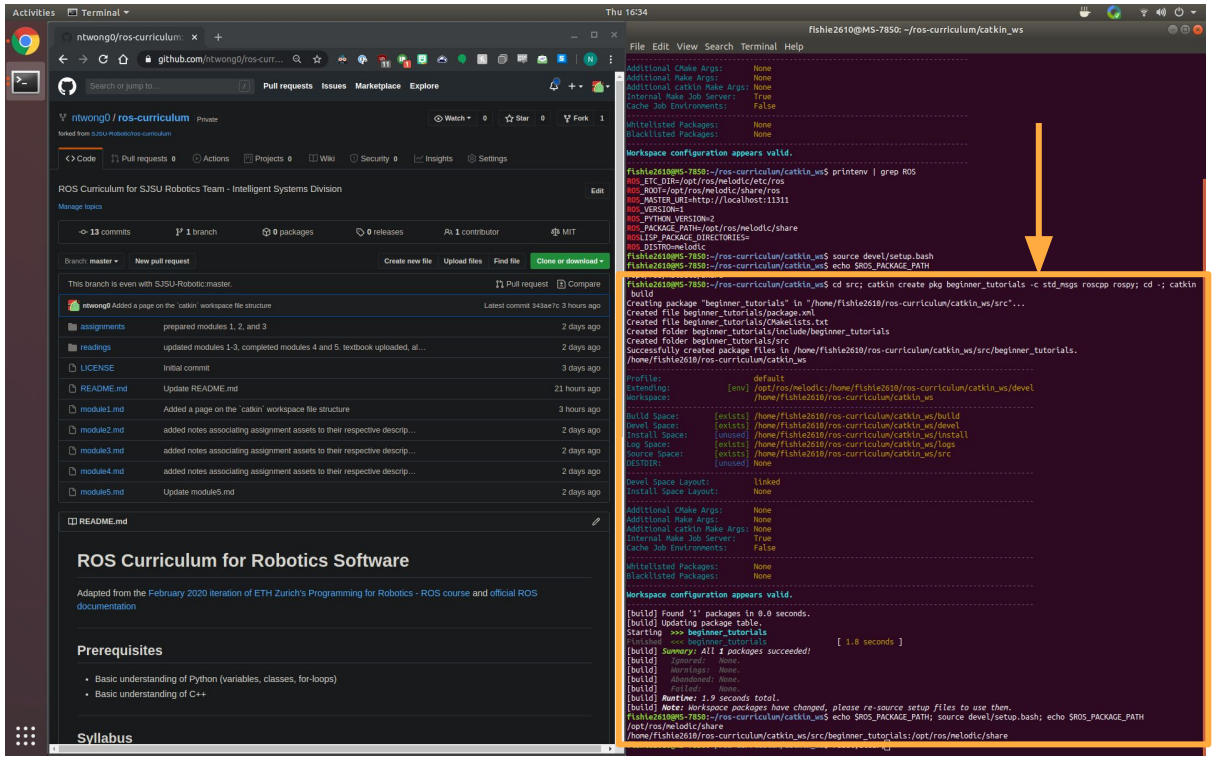
Note that catkin-tools does NOT append to `$ROS_PACKAGE_PATH` if your workspace has no packages in it, unlike `catkin_make`.



ros-curriculum

Note that catkin-tools does NOT append to `$ROS_PACKAGE_PATH` if your workspace has no packages in it, unlike `catkin_make`.

If we create a package inside of catkin_ws and build it, we will see that `source devel/setup.bash` will now update `$ROS_PACKAGE_PATH`

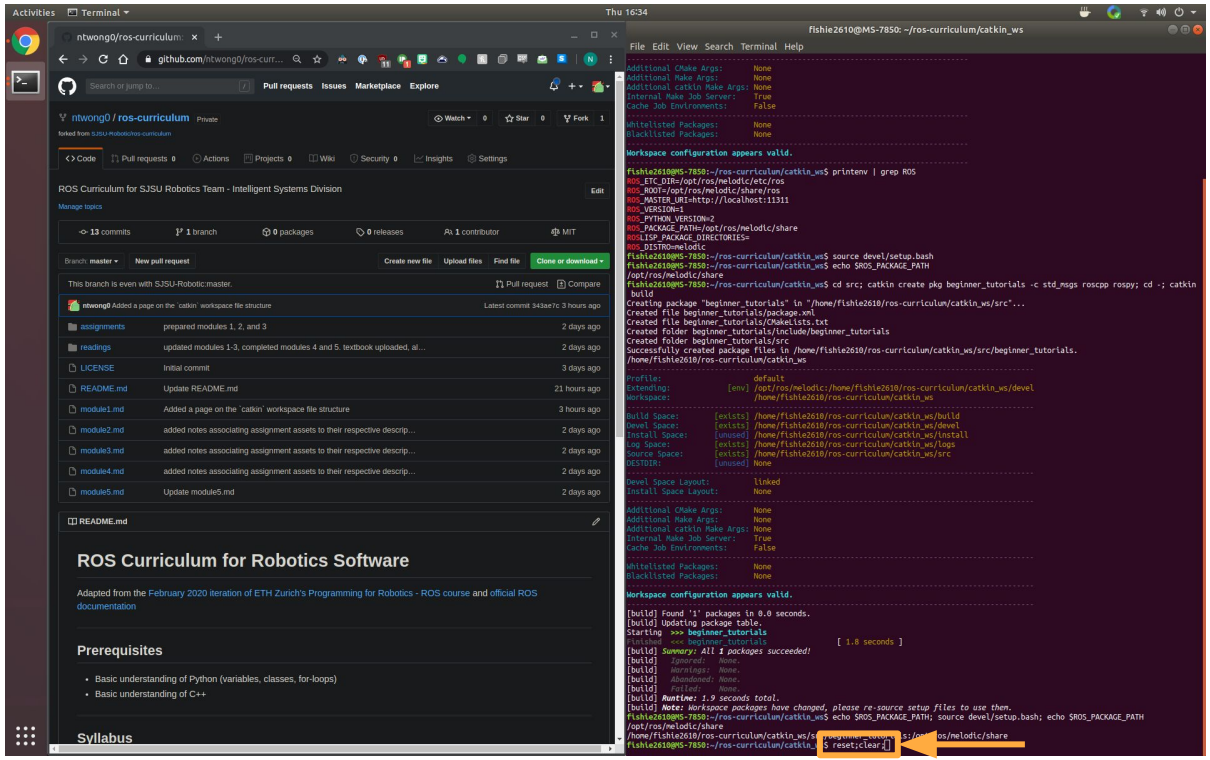


ros-curriculum

Note that catkin-tools does NOT append to `$ROS_PACKAGE_PATH` if your workspace has no packages in it, unlike `catkin_make`.

If we create a package inside of catkin_ws and build it, we will see that `source devel/setup.bash` will now update `$ROS_PACKAGE_PATH`

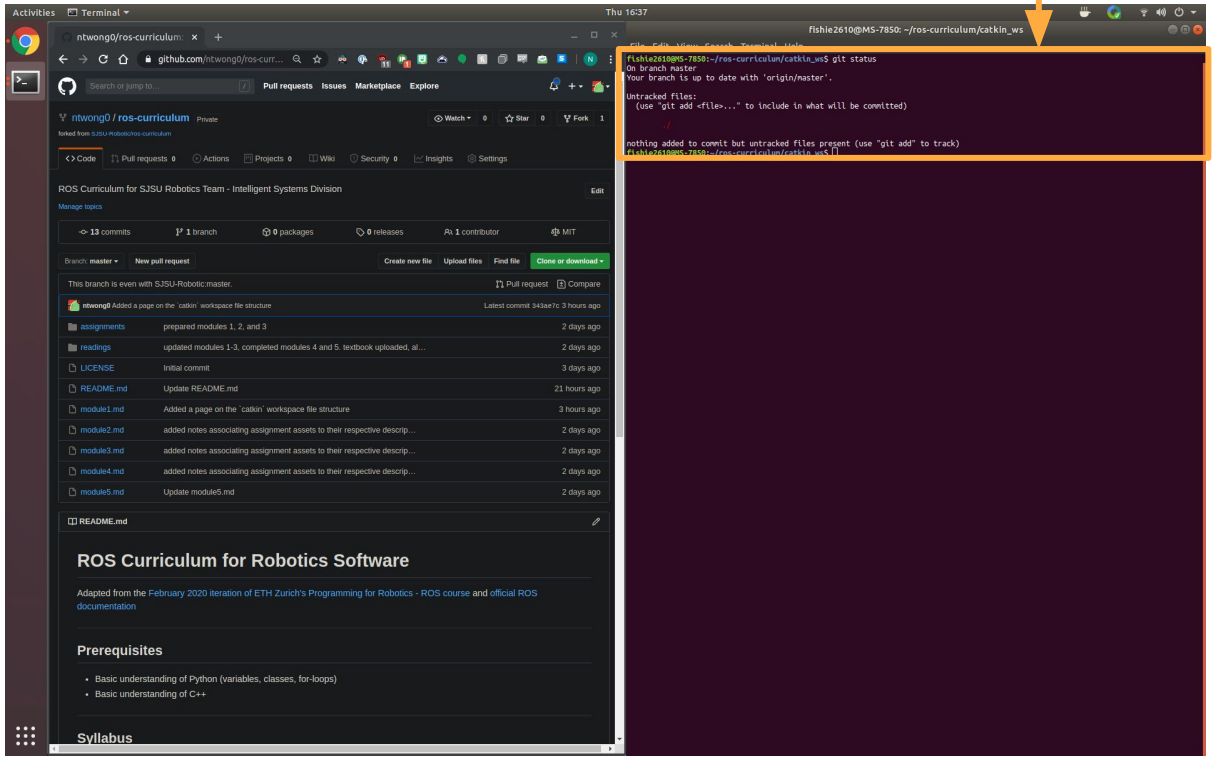
One more note
- `reset` re-initializes the terminal, and
- `clear` clears out text in the terminal



ros-curriculum

Since we created the workshop in the cloned repo, we can add the new files and folders we created.

First, let's check what files and folders we created in the repo with a `git status`. As we can see, the current working directory (`catkin_ws`, indicated with `./`) is untracked, which means it is new to the repo.

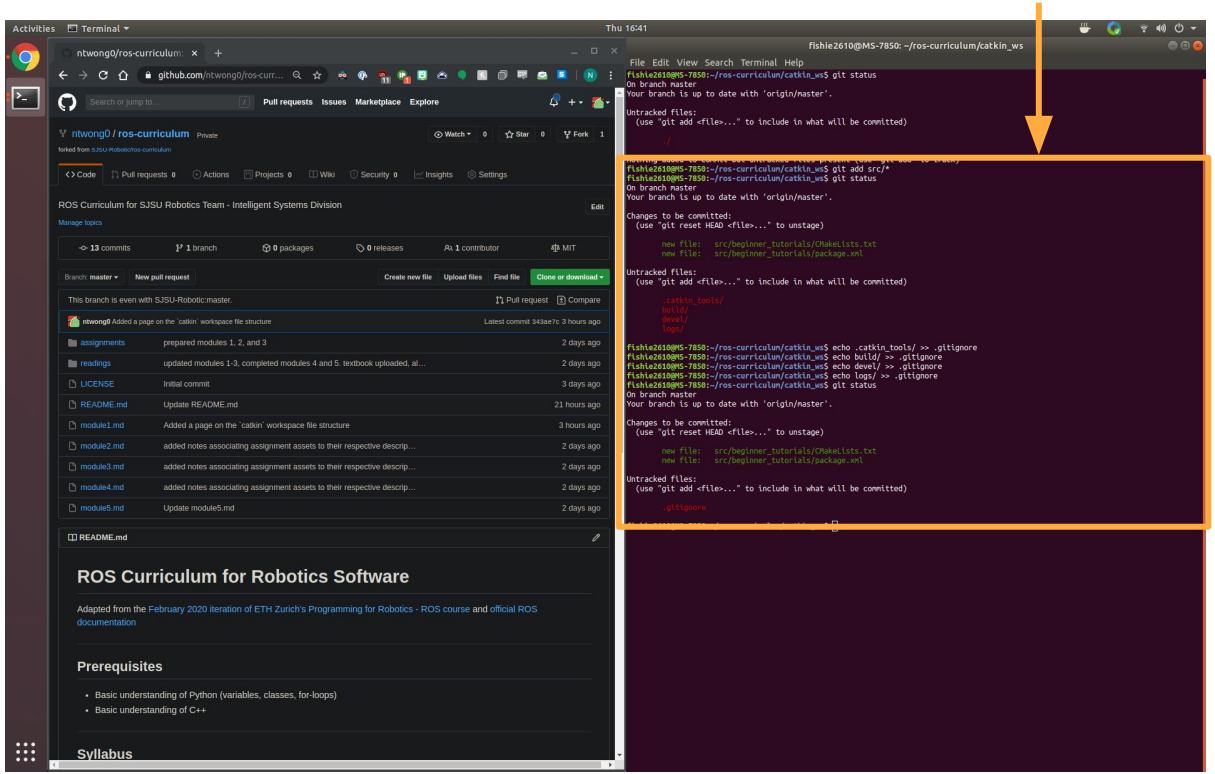


ros-curriculum

Since we created the workshop in the cloned repo, we can add the new files and folders we created.

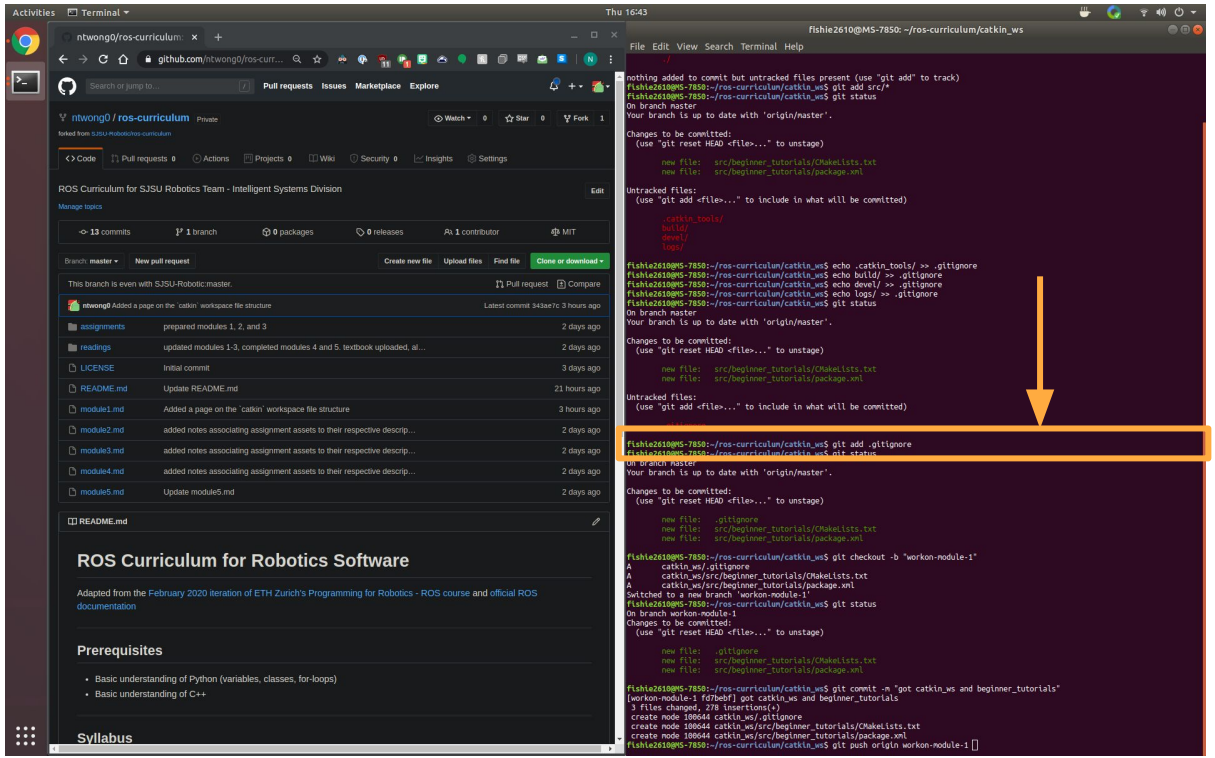
First, let's check what files and folders we created in the repo with a `git status`. As we can see, the current working directory (`catkin_ws`, indicated with `./`) is untracked, which means it is new to the repo.

We only need the contents of the `src/` folder with `git add src/*`, and we can ignore the other files by appending them to the `.gitignore` file



ros-curriculum

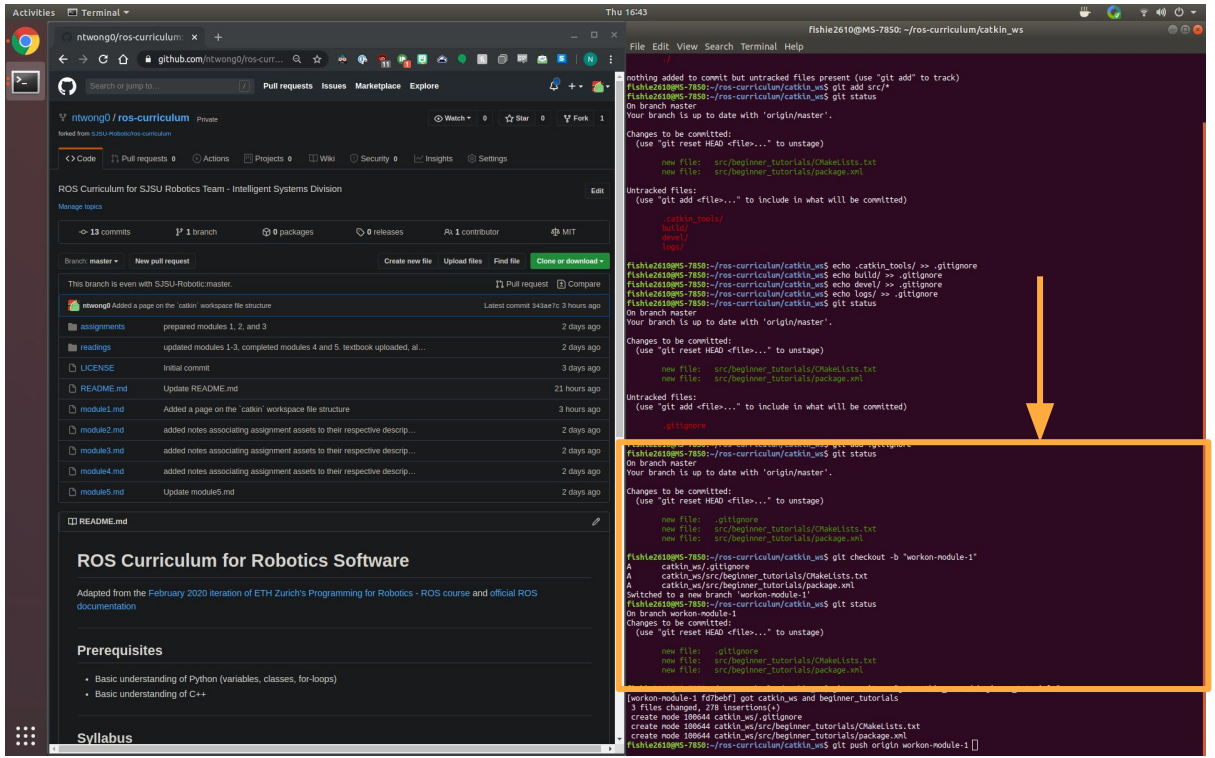
Don’t forget to `git add` the ignore file.



ros-curriculum

Don’t forget to `git add` the ignore file.

Here, we will also switch to a different branch to contain our work related to module1. To switch over to the branch, which previously did not exist, we use `git checkout -b “workon-module-1”`

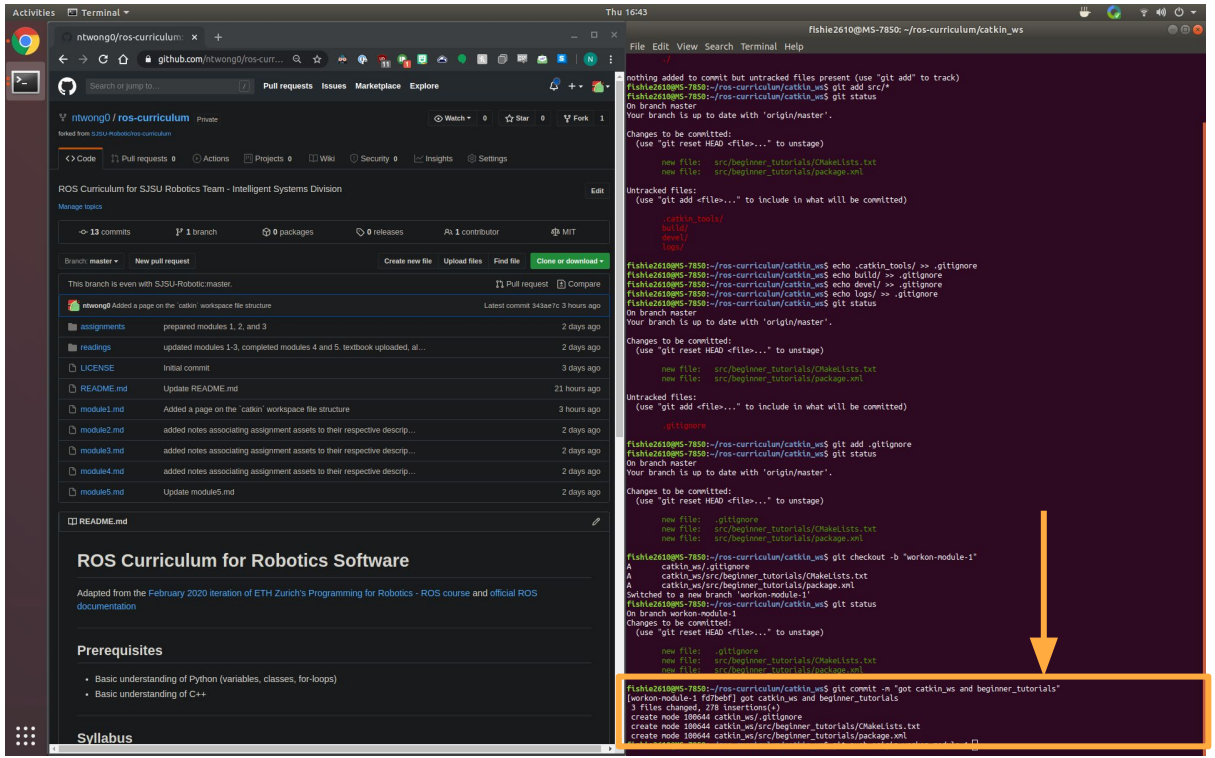


ros-curriculum

Don’t forget to `git add` the ignore file.

Here, we will also switch to a different branch to contain our work related to module1. To switch over to the branch, which previously did not exist, we use `git checkout -b “workon-module-1”`

We then `git commit` our changes, and we use the `-m` flag to append a message describing this commit.



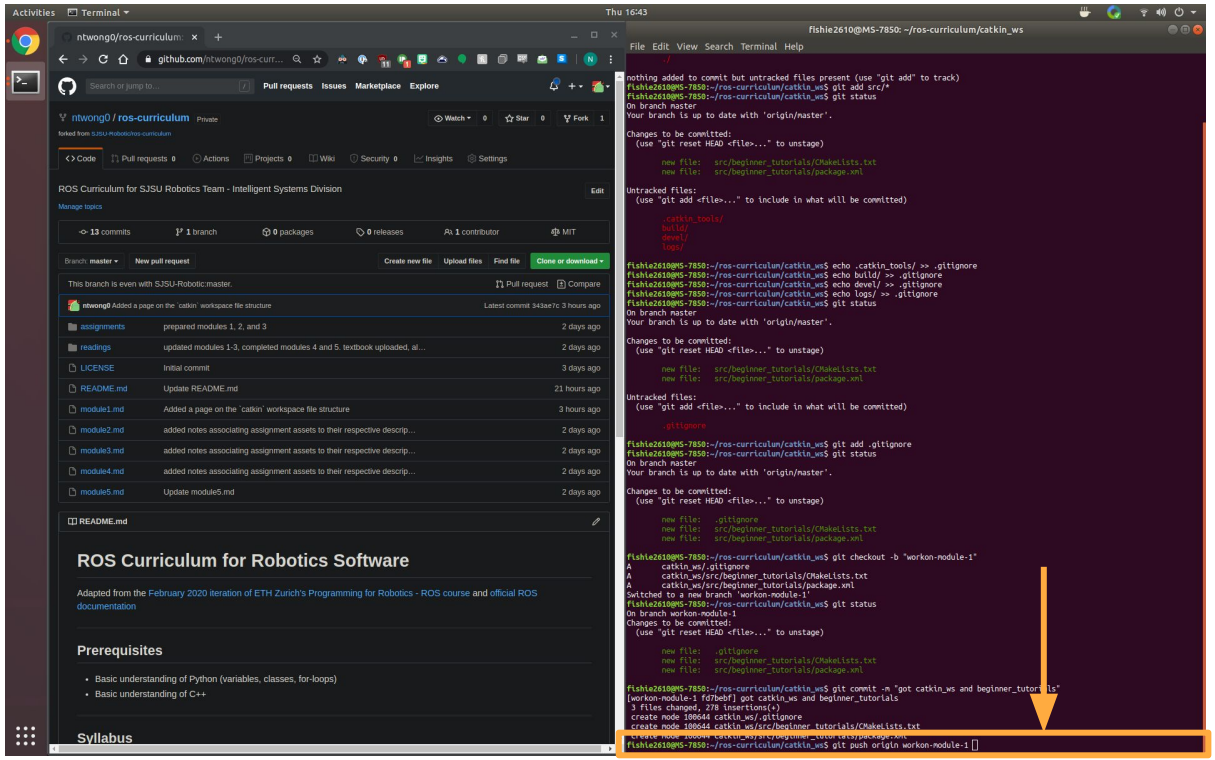
ros-curriculum

Don’t forget to `git add` the ignore file.

Here, we will also switch to a different branch to contain our work related to module1. To switch over to the branch, which previously did not exist, we use `git checkout -b “workon-module-1”`

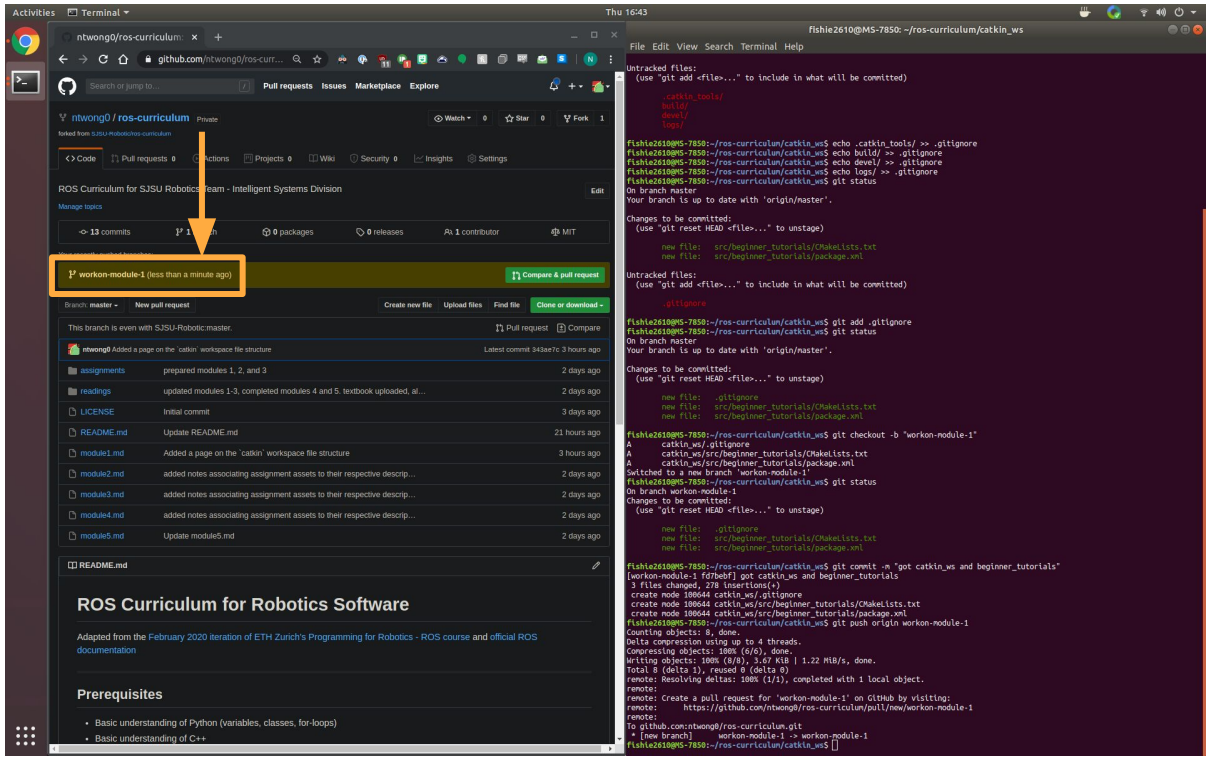
We then `git commit` our changes, and we use the `-m` flag to append a message describing this commit.

Finally, we `git push` onto the `origin` repo (the repo on our GitHub account) our committed changes from the `workon-module-1 branch`



ros-curriculum

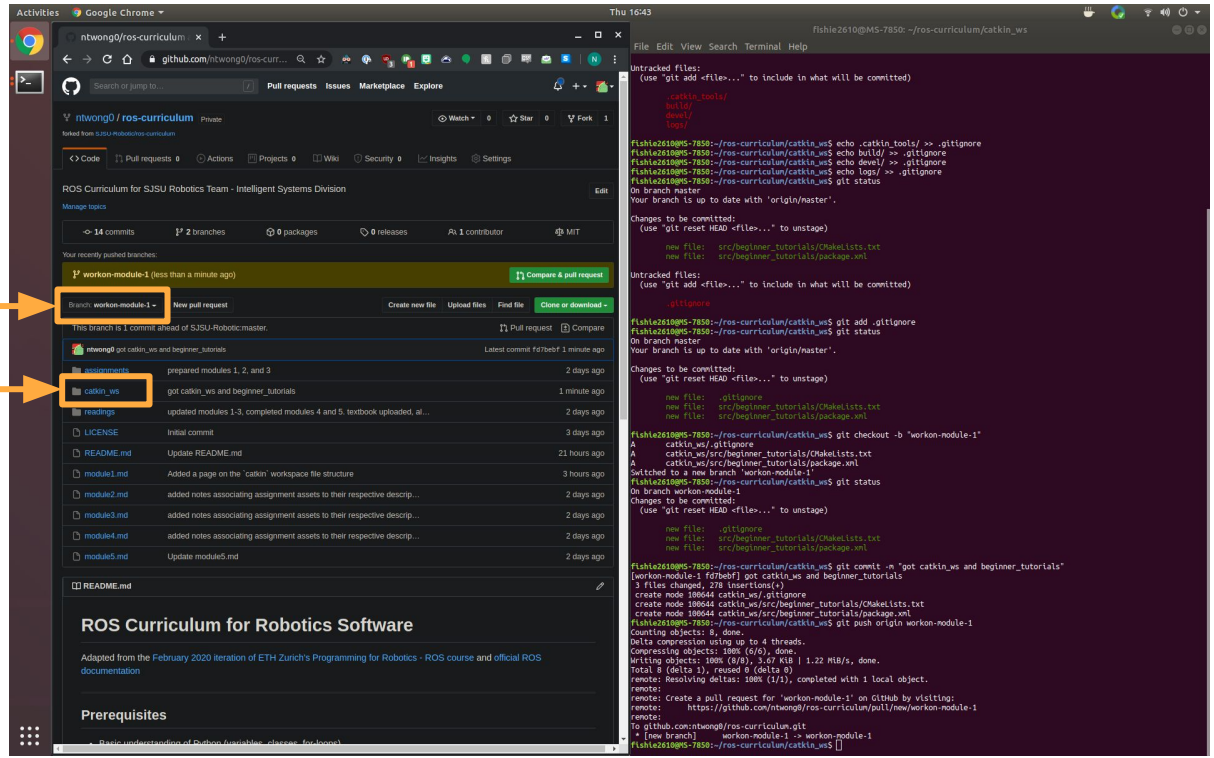
Once the commit is pushed, we can see the result GitHub. Selecting `workon-module-1` will display the freshly uploaded branch.



ros-curriculum

Once the commit is pushed, we can see the result GitHub. Selecting `workon-module-1` will display the freshly uploaded branch.

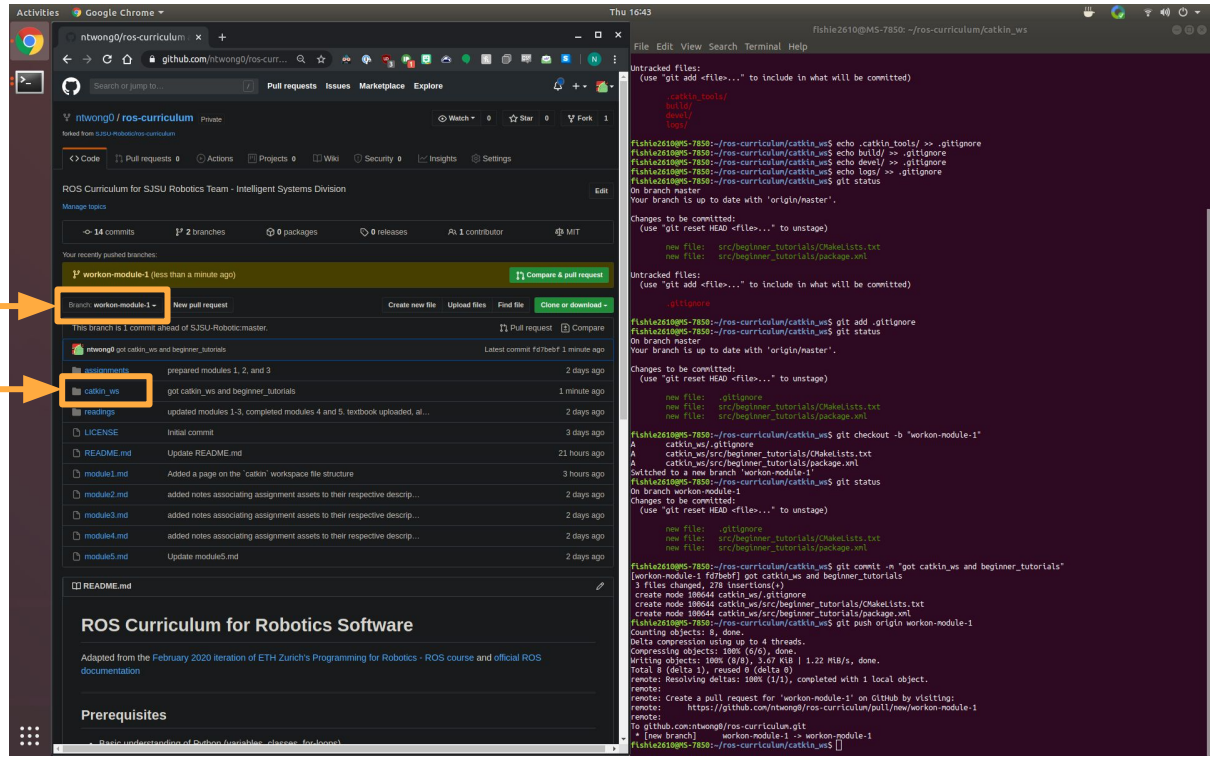
As we can see in the `workon-module-1` branch, the `catkin_ws` directory now appears



ros-curriculum

Once the commit is pushed, we can see the result GitHub. Selecting `workon-module-1` will display the freshly uploaded branch.

As we can see in the `workon-module-1` branch, the `catkin_ws` directory now appears



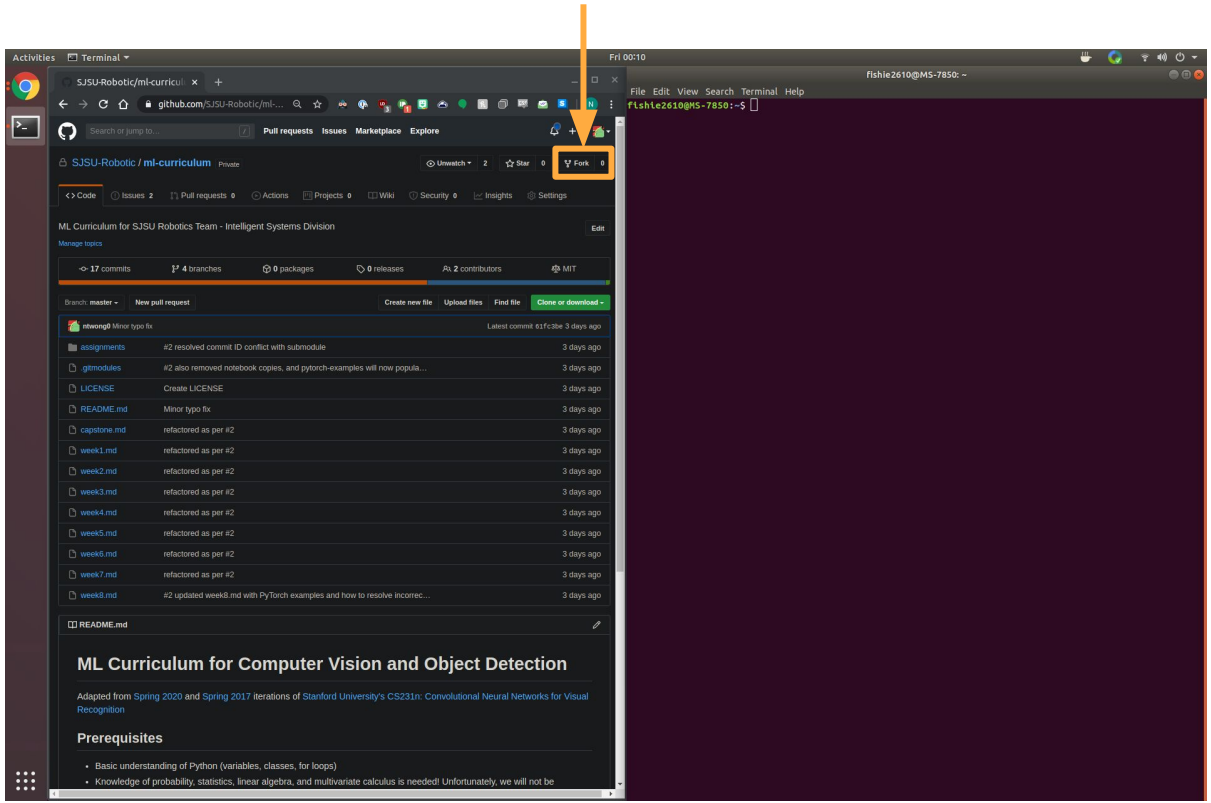
ml-curriculum

ml-curriculum

You can use the course repo as a container for your ROS workspace.

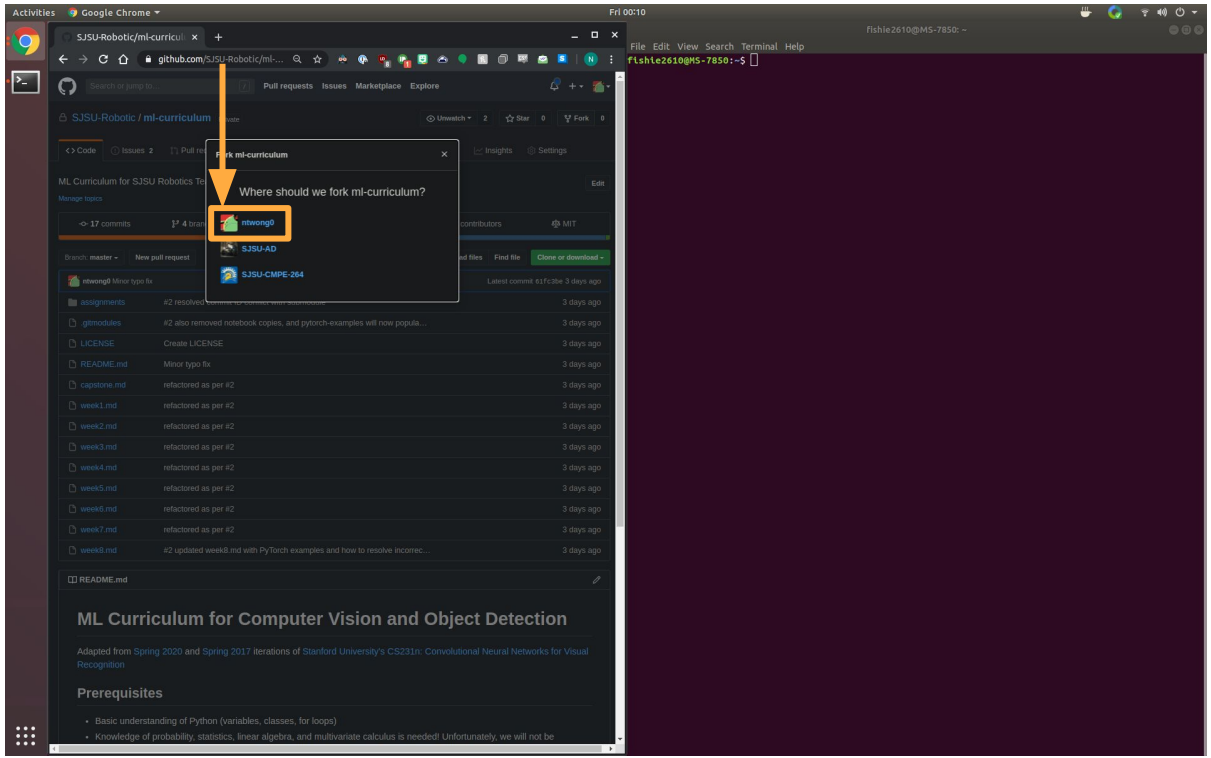
First, we start with the repo page on the left, and a terminal window on the right.

[Create a fork of the repo](#) to your GitHub account by selecting the [Fork] button.



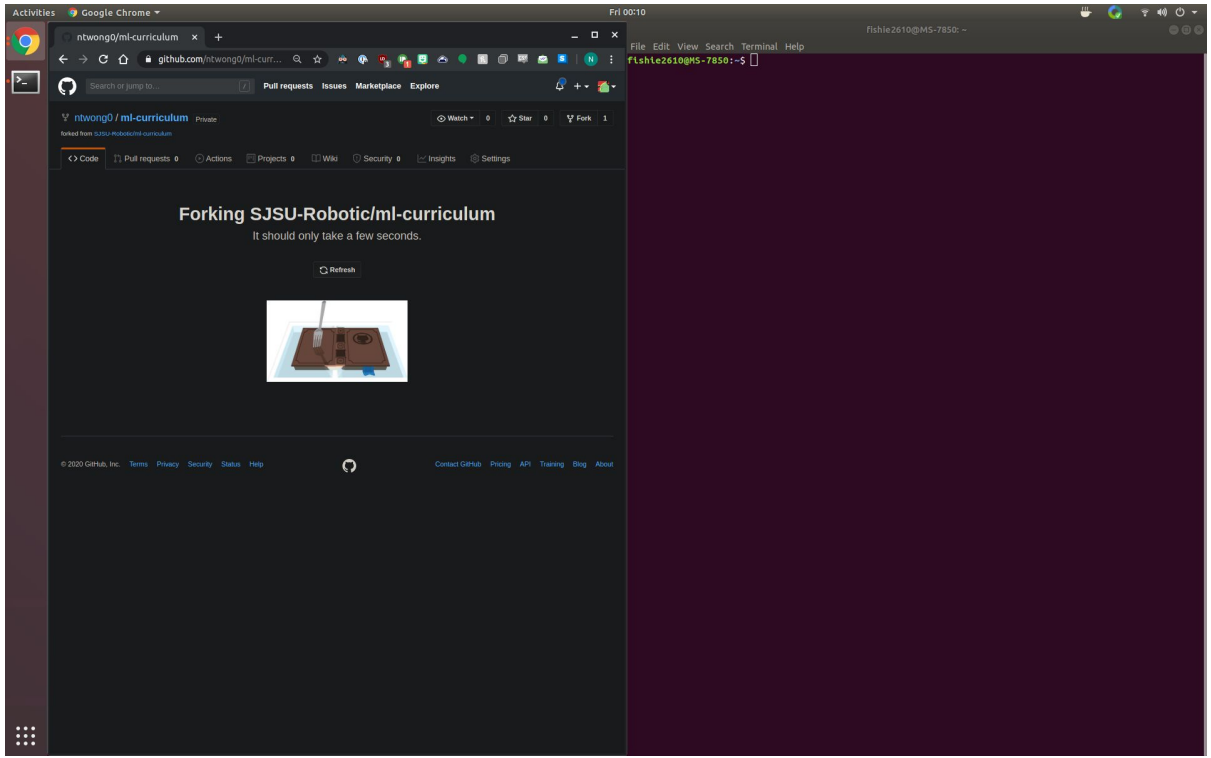
ml-curriculum

Select your account as the fork destination



ml-curriculum

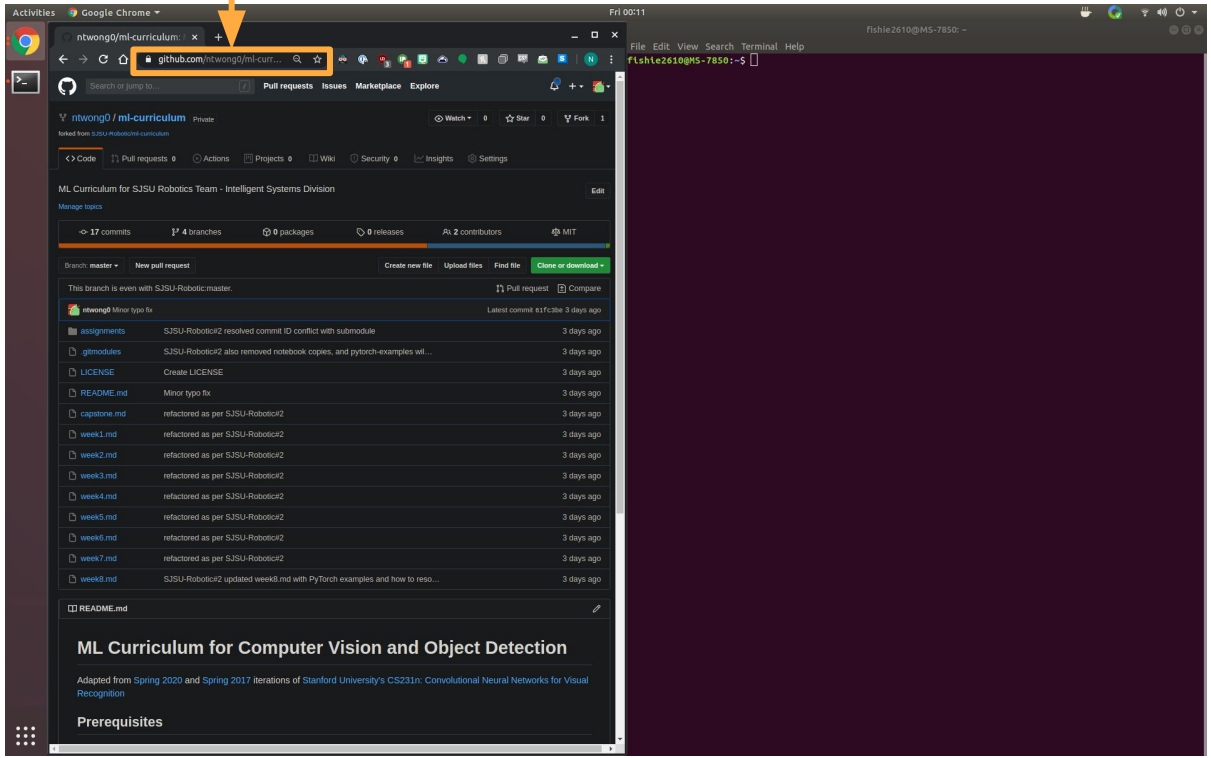
Forking will take a few seconds.



ml-curriculum

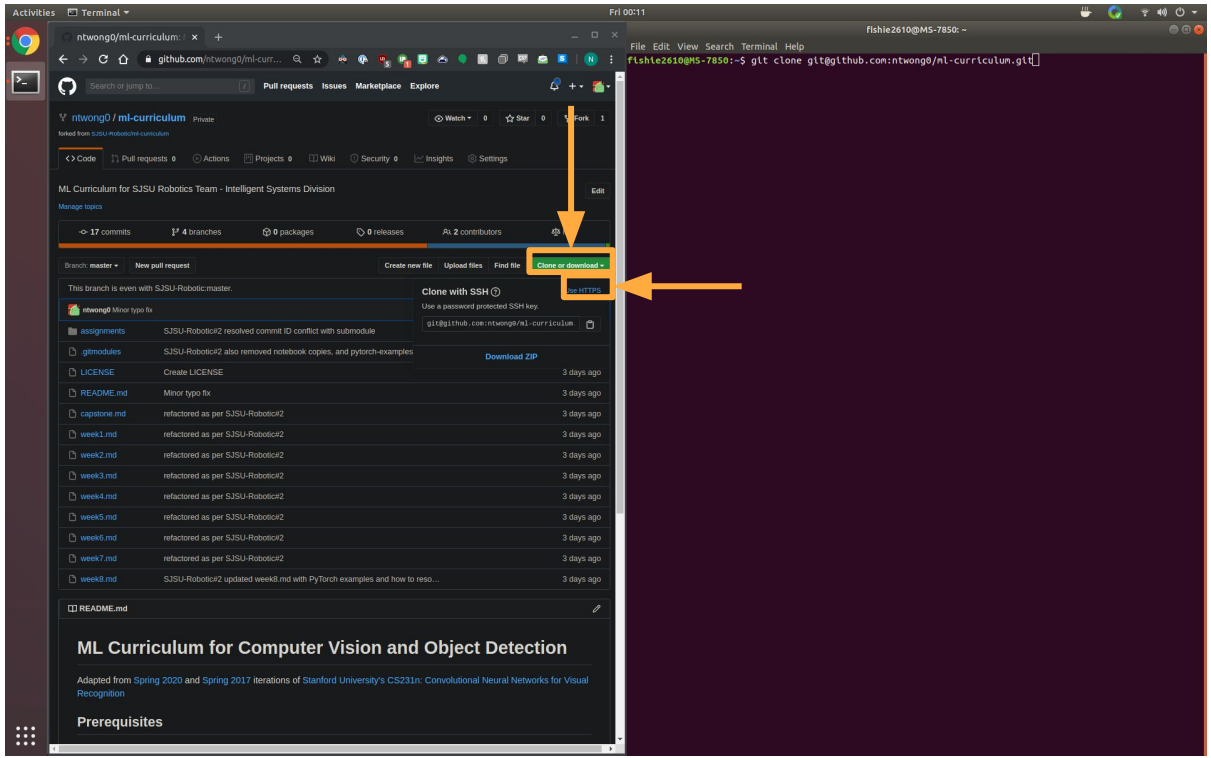
The fork is now complete, and it is now accessible at

github.com/<yourAcct>/ml-curriculum



ml-curriculum

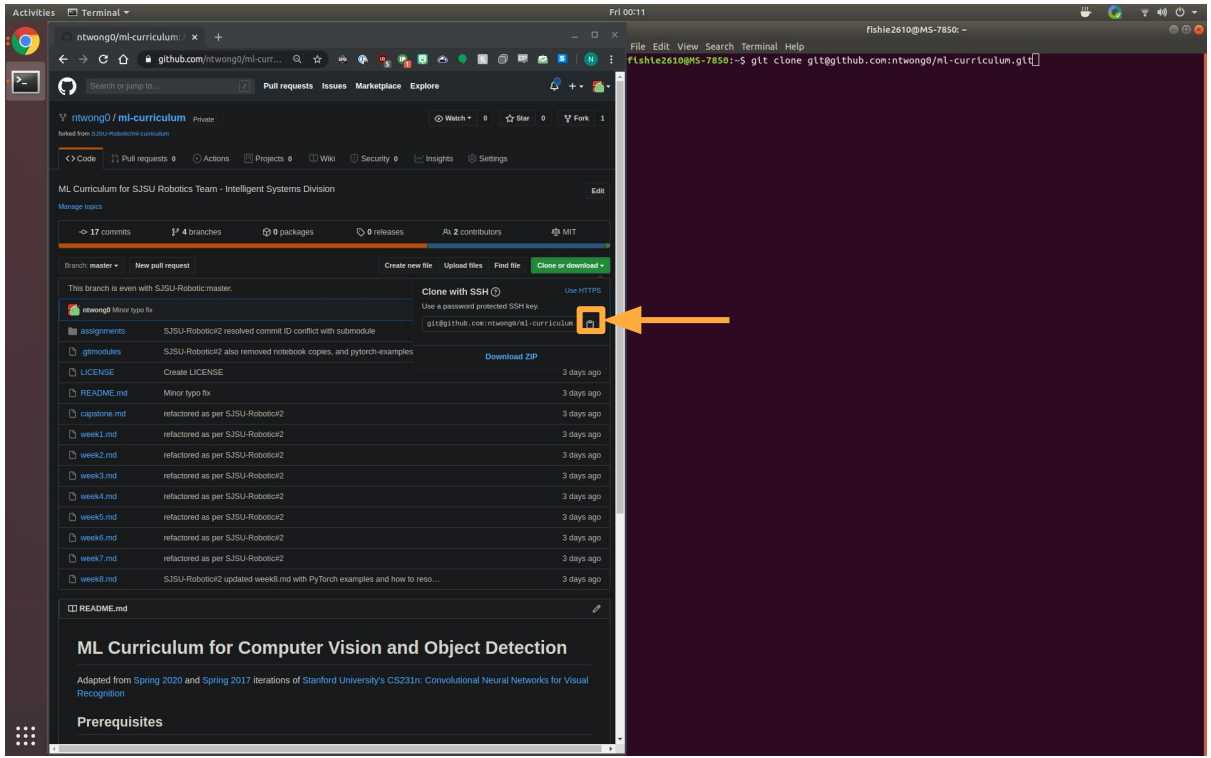
Select [Clone or download], then select [Use SSH] to reveal the URL.



ml-curriculum

Select [Clone or download], then select [Use SSH] to reveal the URL.

Copy the URL to your clipboard.



ml-curriculum

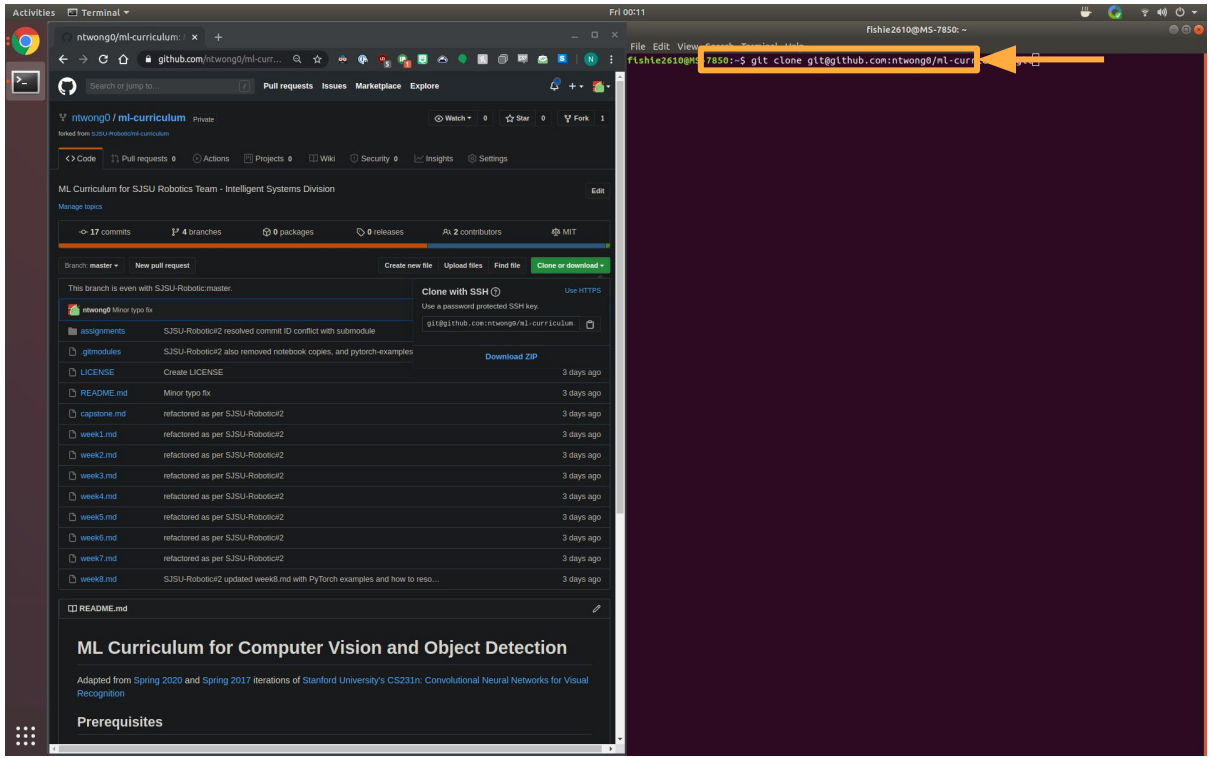
Select [Clone or download], then select [Use SSH] to reveal the URL.

Copy the URL to your clipboard.

In the terminal window, type

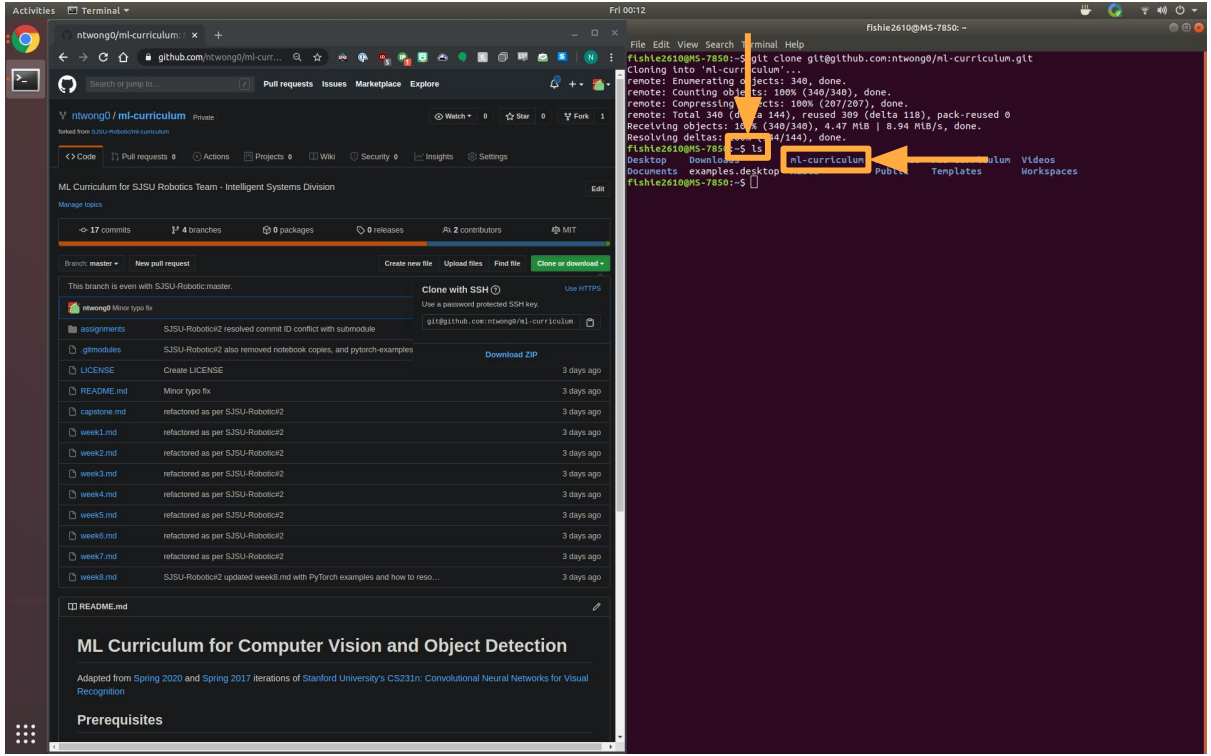
```
git clone
```

then paste the URL, and hit enter.



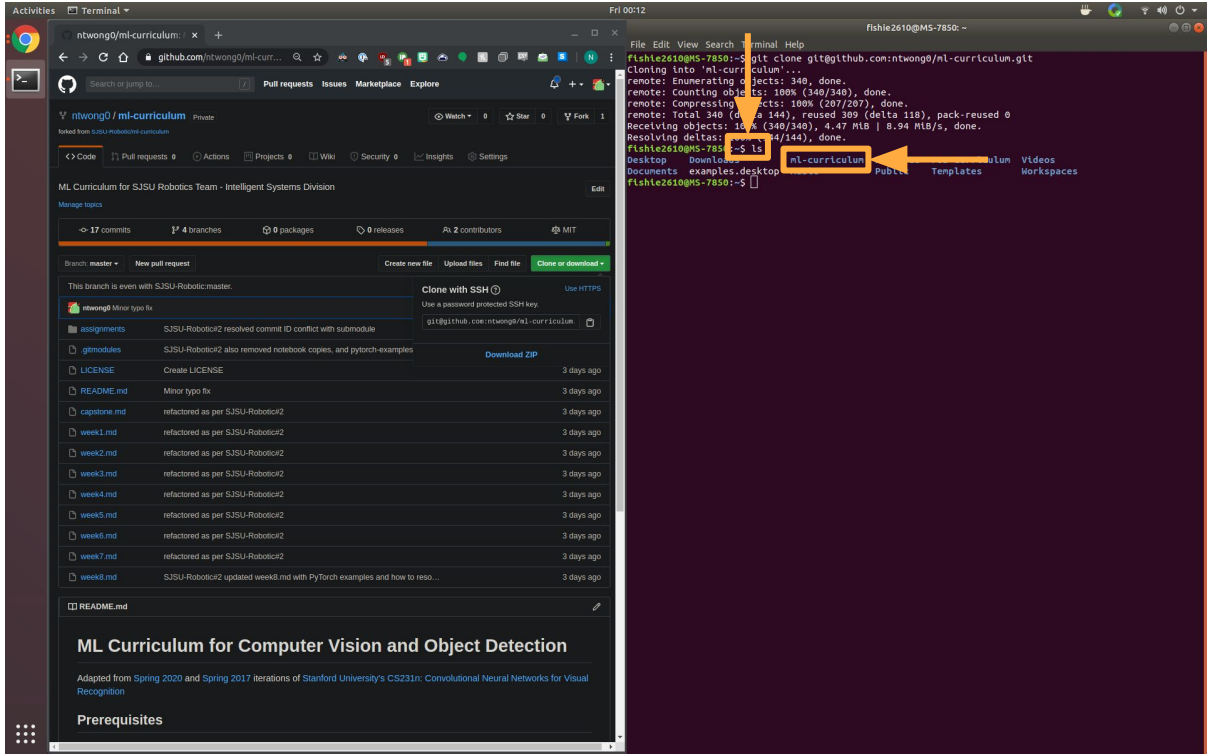
ml-curriculum

The repo is cloned into a folder at the current working directory. If you use the `ls` command to list the working directory's contents, the `ros-curriculum` folder is listed



ml-curriculum

The repo is cloned into a folder at the current working directory. If you use the `ls` command to list the working directory's contents, the `ros-curriculum` folder is listed

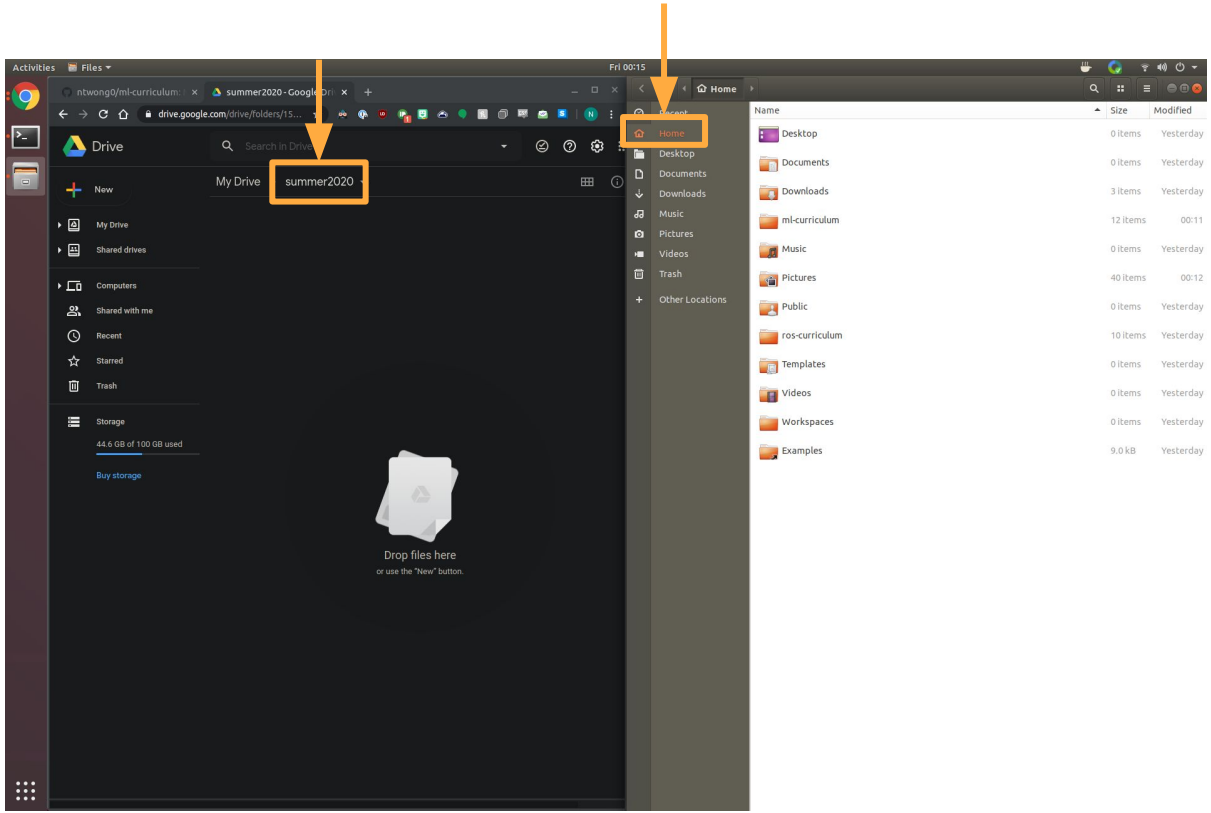


ml-curriculum

Now that we have cloned the folder, let's upload the assignment files to Google Drive.

In our browser, open Google Drive, and create a folder called `summer2020`

Let's also open a file explorer window (you can use the `Ctrl + N` hotkey), and select the Home directory



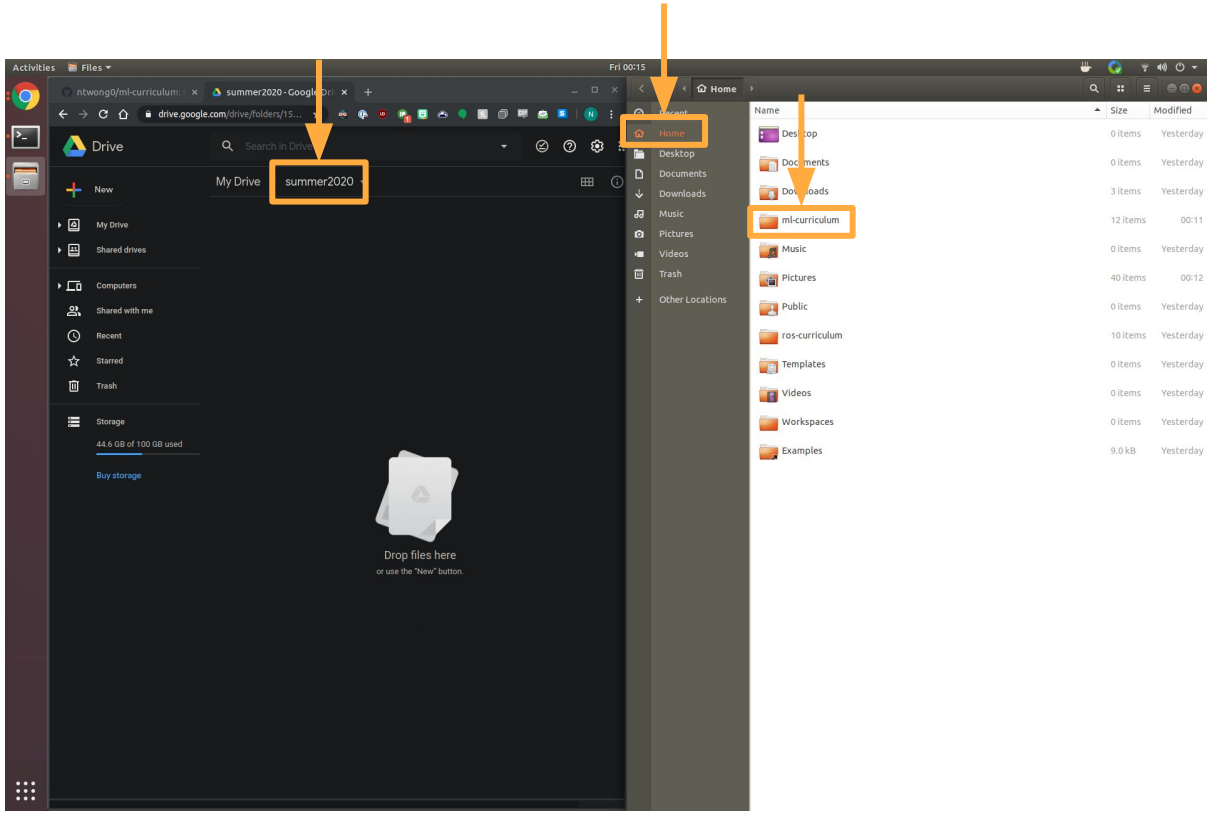
ml-curriculum

Now that we have cloned the folder, let's upload the assignment files to Google Drive.

In our browser, open Google Drive, and create a folder called `summer2020`

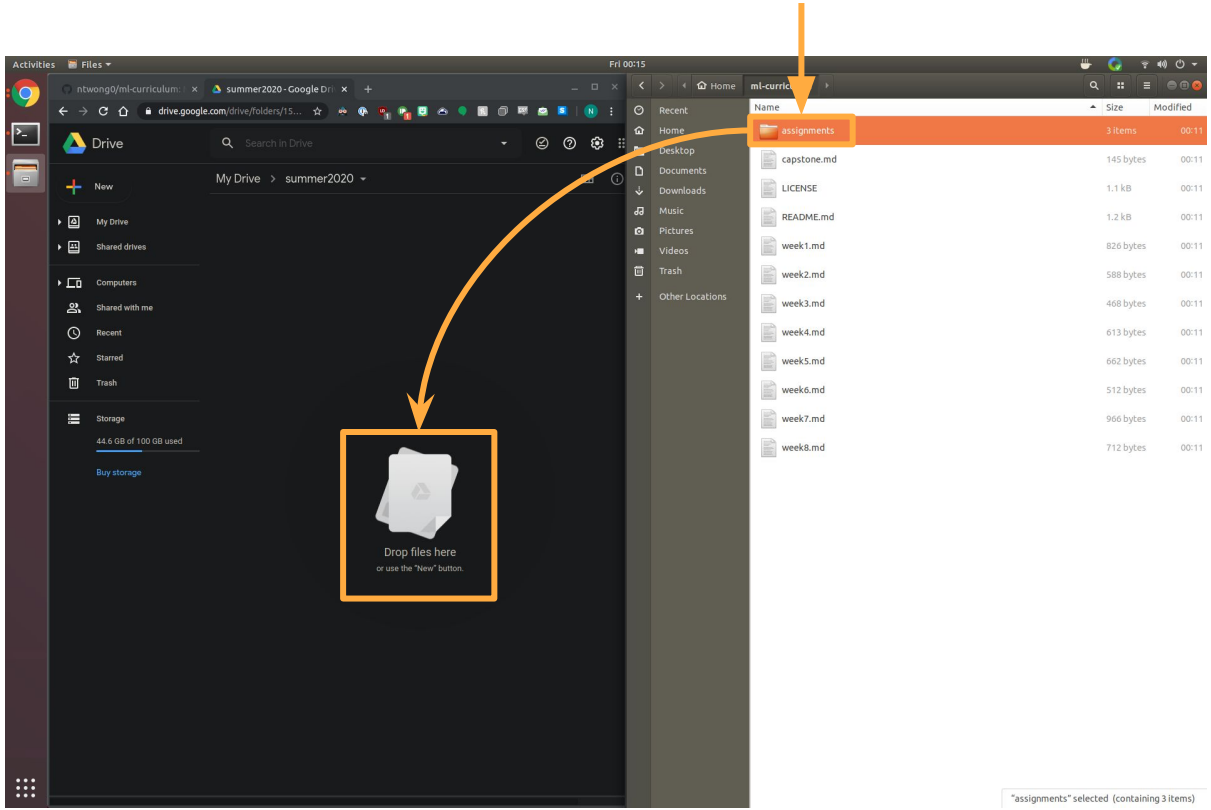
Let's also open a file explorer window (you can use the `Ctrl + N` hotkey), and select the Home directory.

Then, double-click to open the `ml-curriculum` directory



ml-curriculum

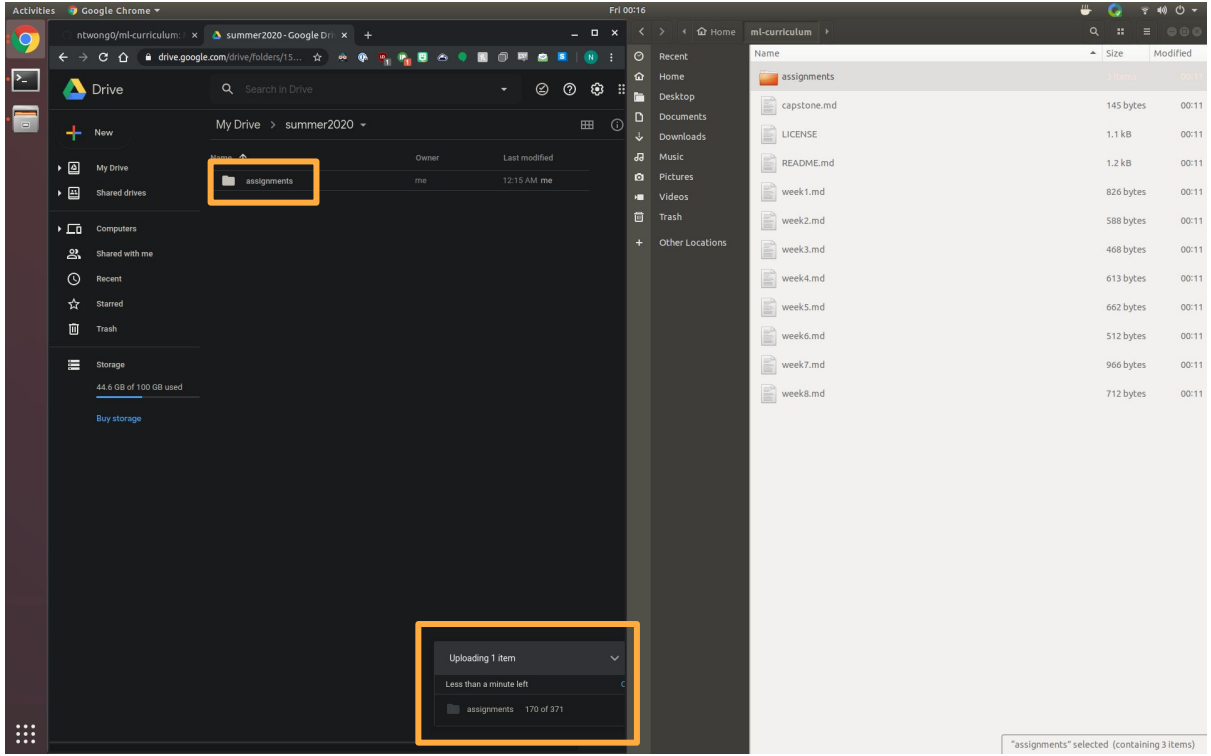
Then, select the `assignments` directory, and drag-and-drop the assignments directory into `summer2020`...



ml-curriculum

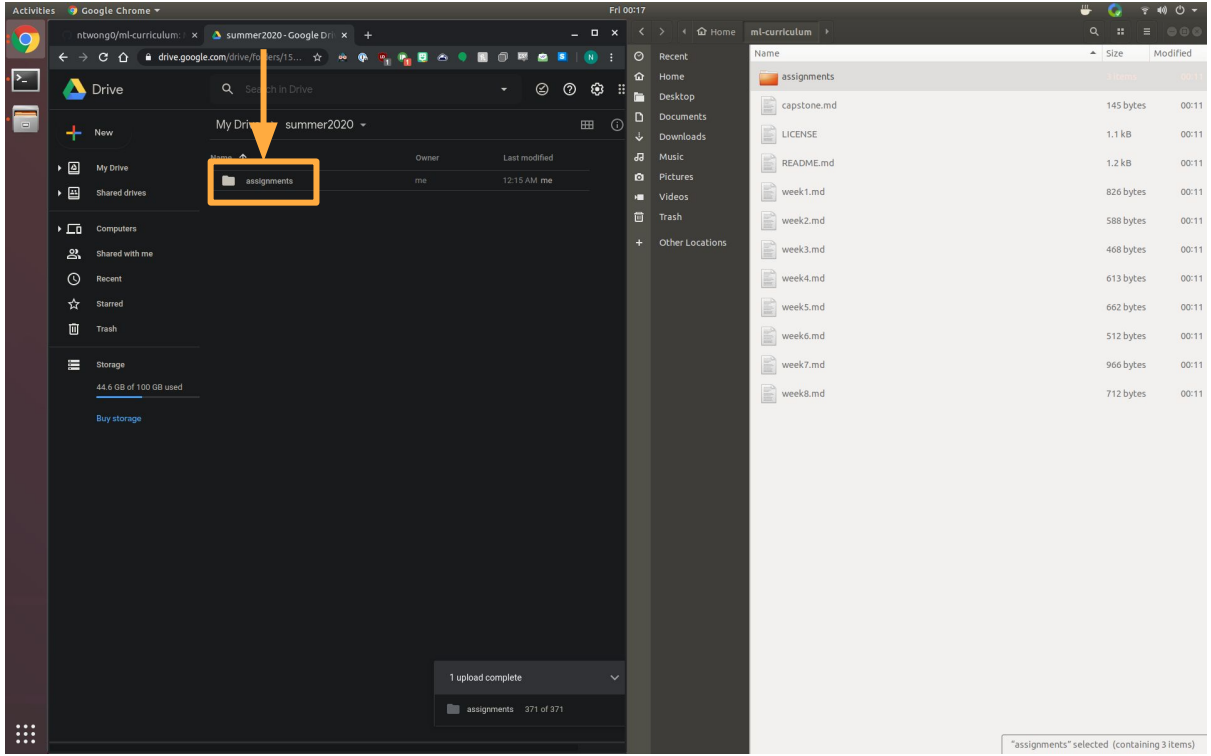
Then, select the `assignments` directory, and drag-and-drop the assignments directory into `summer2020`...

...and the directory will begin to upload to your Google Drive.



ml-curriculum

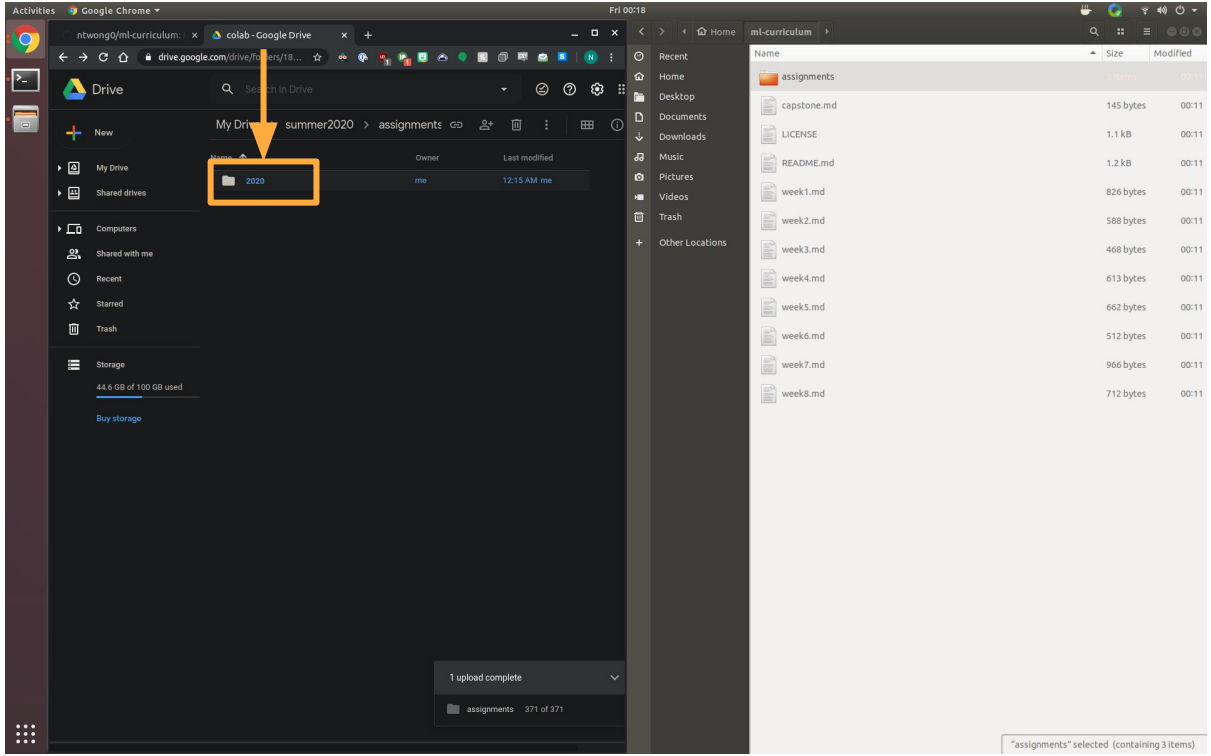
Once the upload is complete, double-click to open the `assignments` folder...



ml-curriculum

Once the upload is complete, double-click to open the `assignments` folder…

...then double-click to open the `2020` folder...



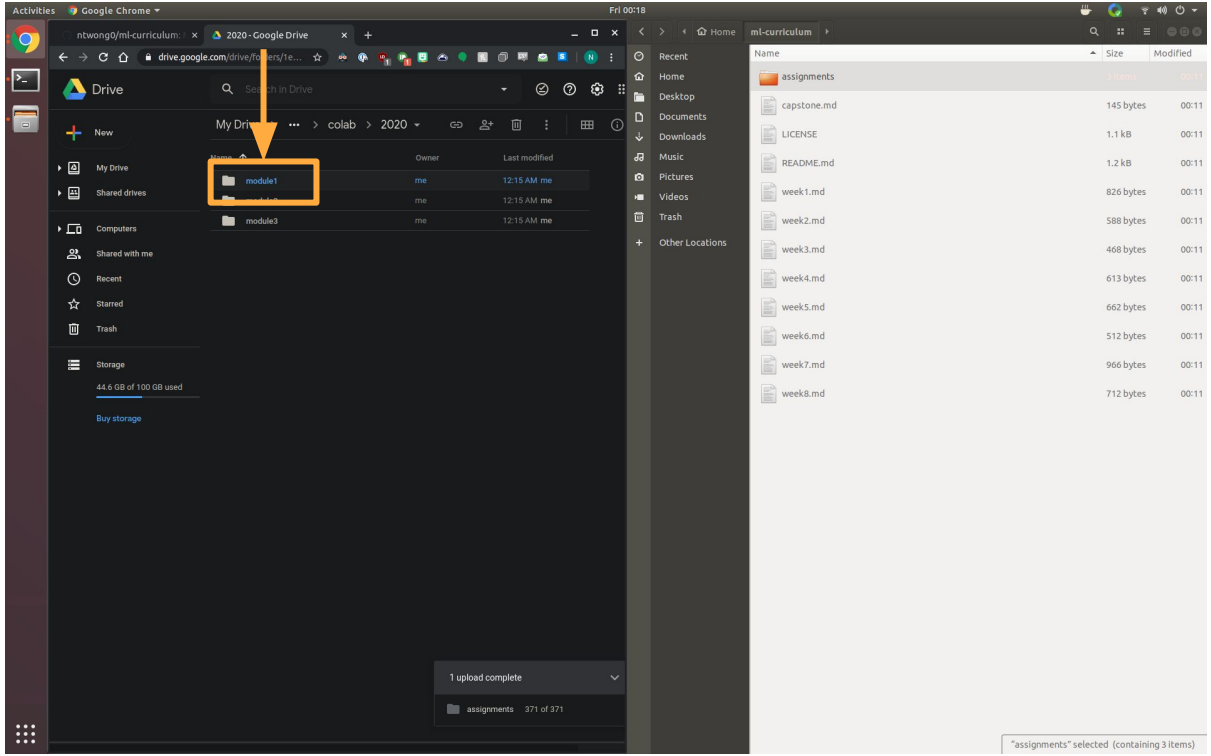
ml-curriculum

Once the upload is complete, double-click to open the `assignments` folder…

...then double-click to open the `2020` folder…

...and then double-click again to open the `module1` folder.

You are now able to access the assignment files.

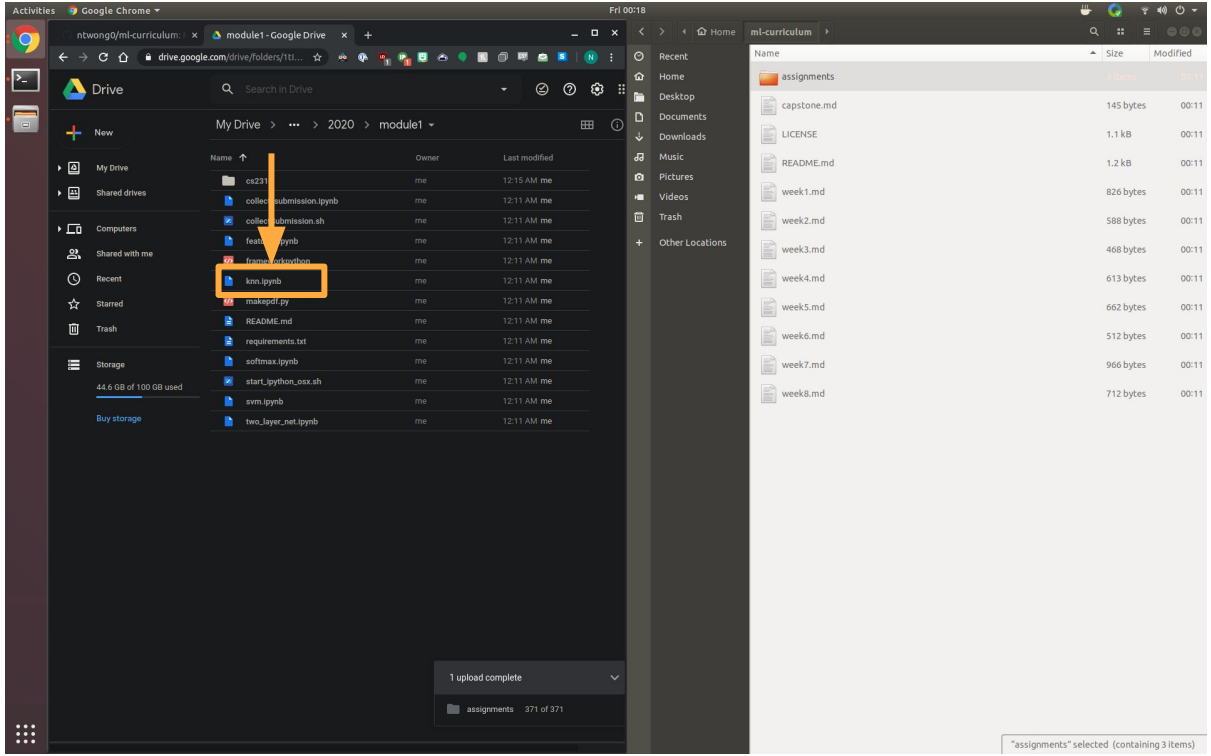


ml-curriculum

For each assignment, you'll need to modify both the `.ipynb` notebook file and the underlying `.py` implementation files.

First, we'll discuss the notebook file.

Right-click the `knn.ipynb` file...



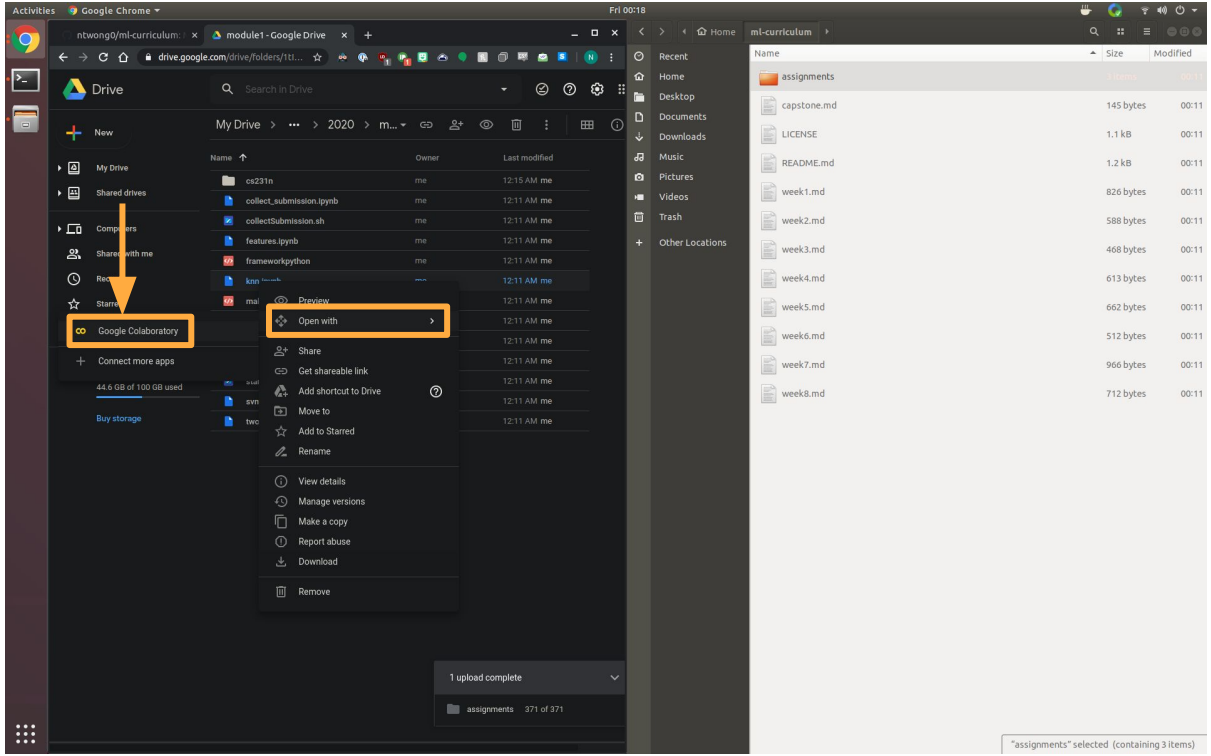
ml-curriculum

For each assignment, you'll need to modify both the `.ipynb` notebook file and the underlying `.py` implementation files.

First, we'll discuss the notebook file.

Right-click the `knn.ipynb` file...

...then hover over "Open with", and select "Google Colaboratory".



ml-curriculum

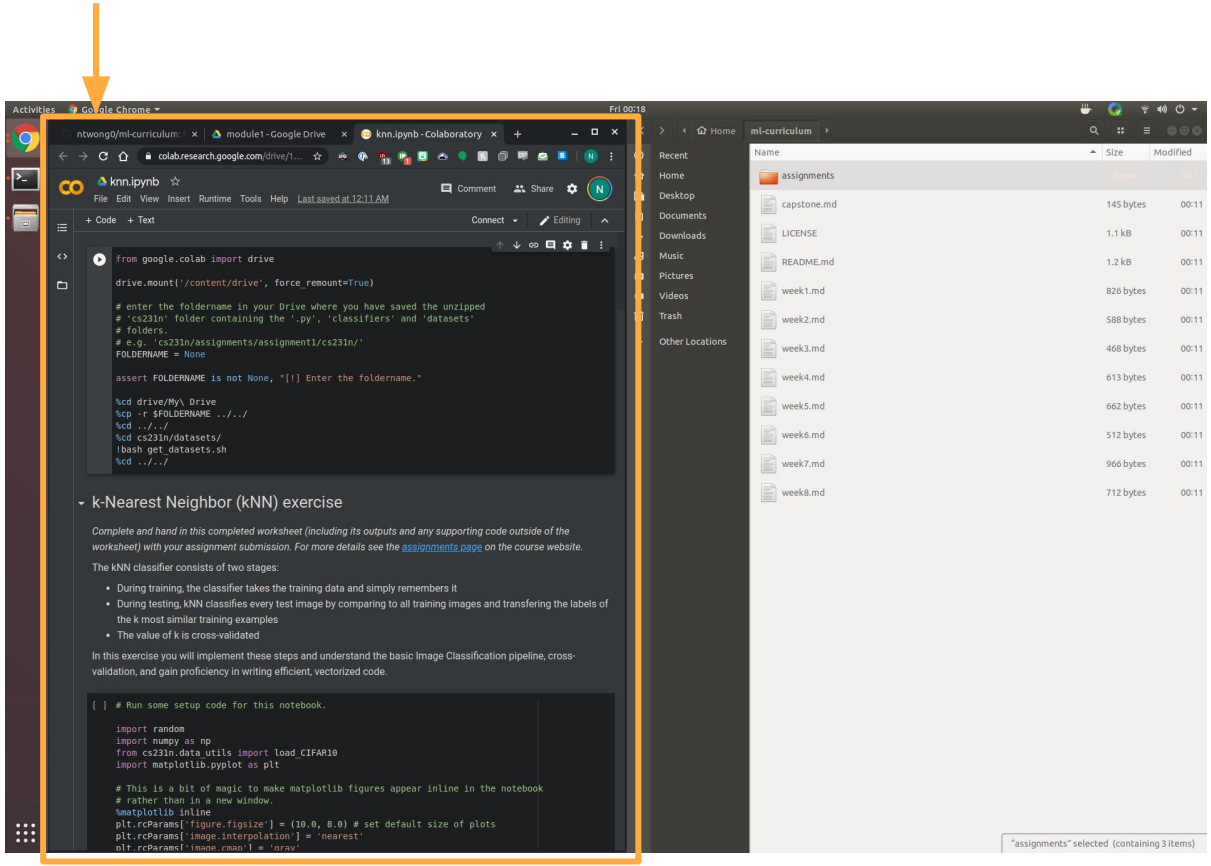
For each assignment, you'll need to modify both the `.ipynb` notebook file and the underlying `.py` implementation files.

First, we'll discuss the notebook file.

Right-click the `knn.ipynb` file...

...then hover over "Open with", and select "Google Colaboratory".

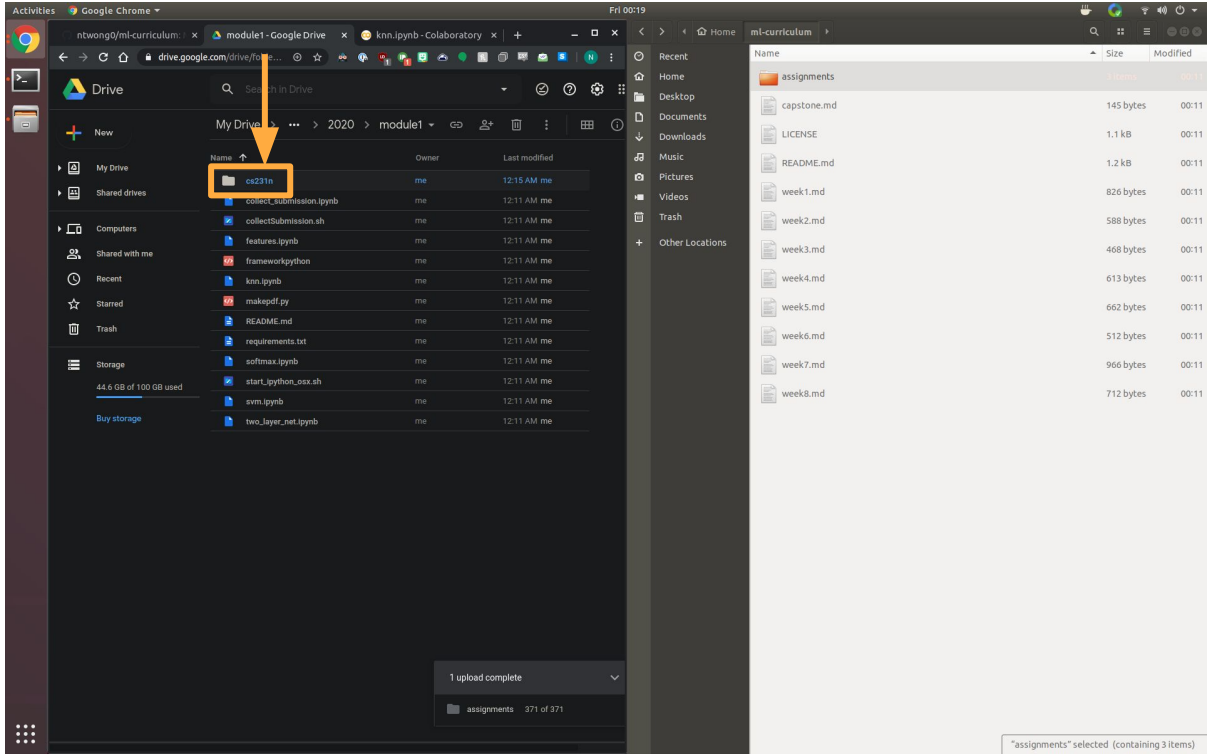
The notebook is now open via Google Colaboratory (colab), and you can proceed with the assignment here.



ml-curriculum

Next, we'll discuss the Python implementation file.

Back in the `module1` Google Drive folder, double-click to open `cs231n`...

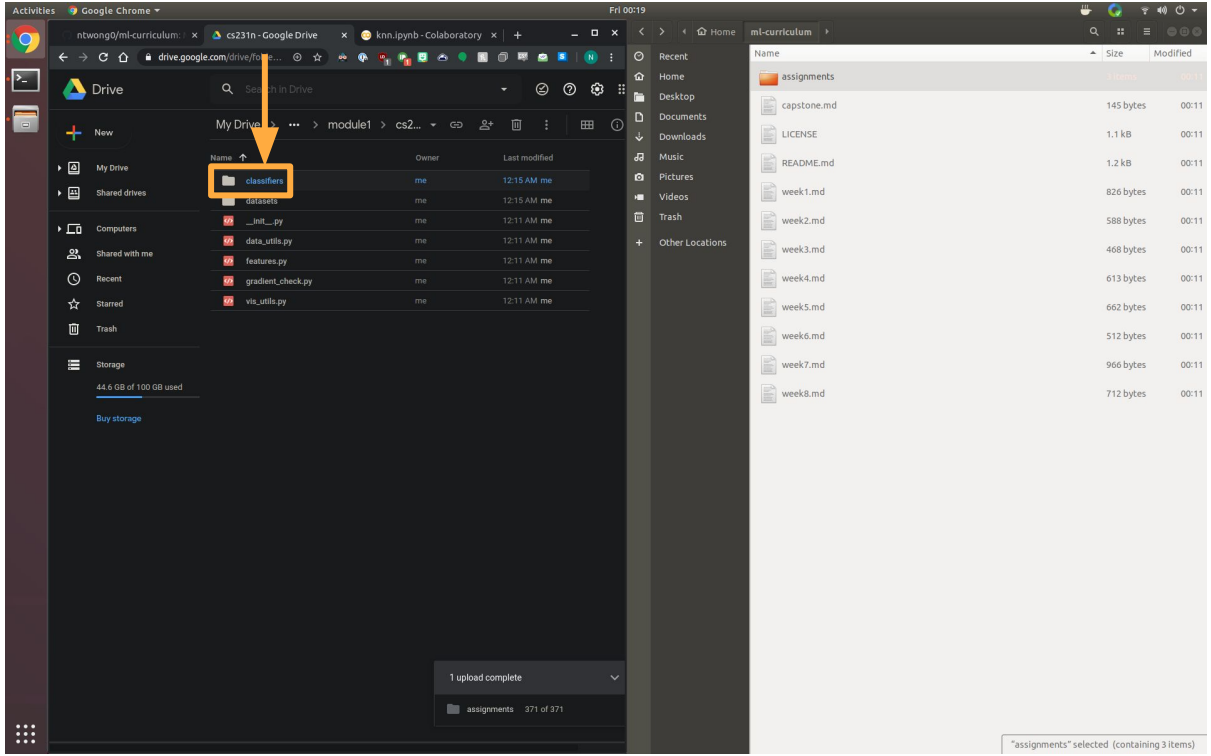


ml-curriculum

Next, we'll discuss the Python implementation file.

Back in the `module1` Google Drive folder, double-click to open `cs231n`...

...and double-click to open `classifiers`...



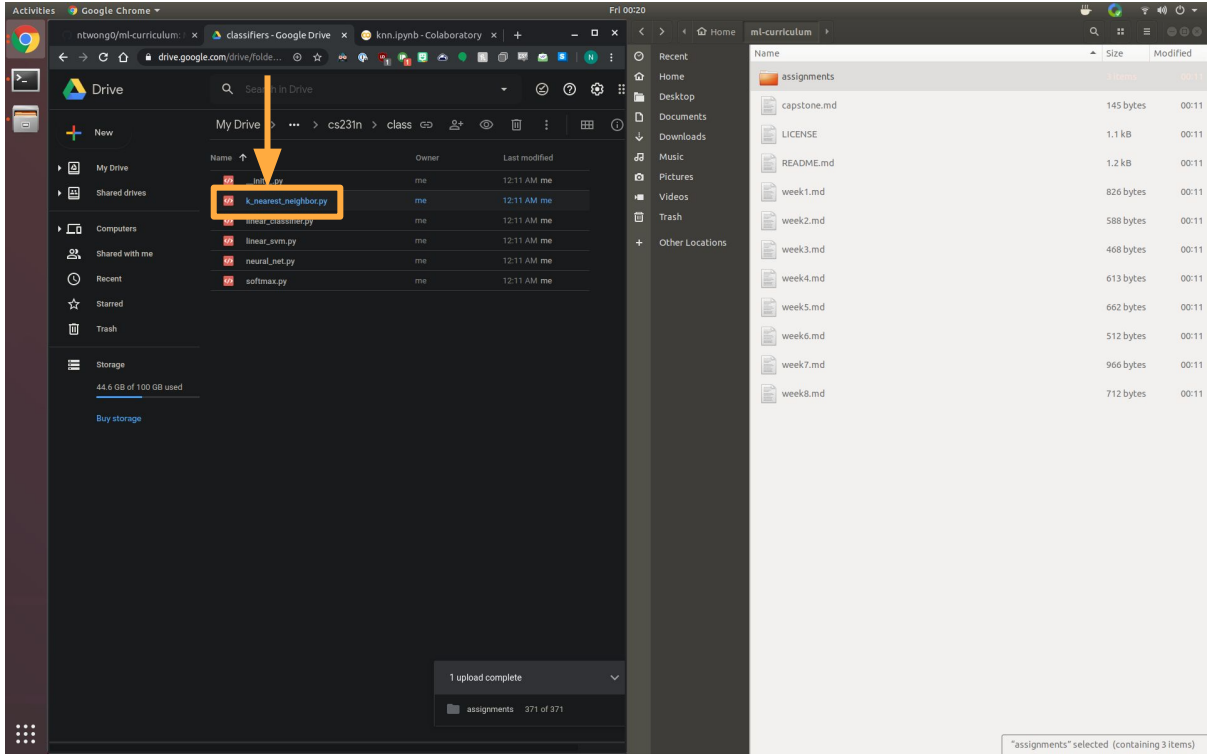
ml-curriculum

Next, we'll discuss the Python implementation file.

Back in the `module1` Google Drive folder, double-click to open `cs231n`...

...and double-click to open `classifiers`...

...and right-click on the implementation file of interest. Here, we have `k_nearest_neighbor.py`.



ml-curriculum

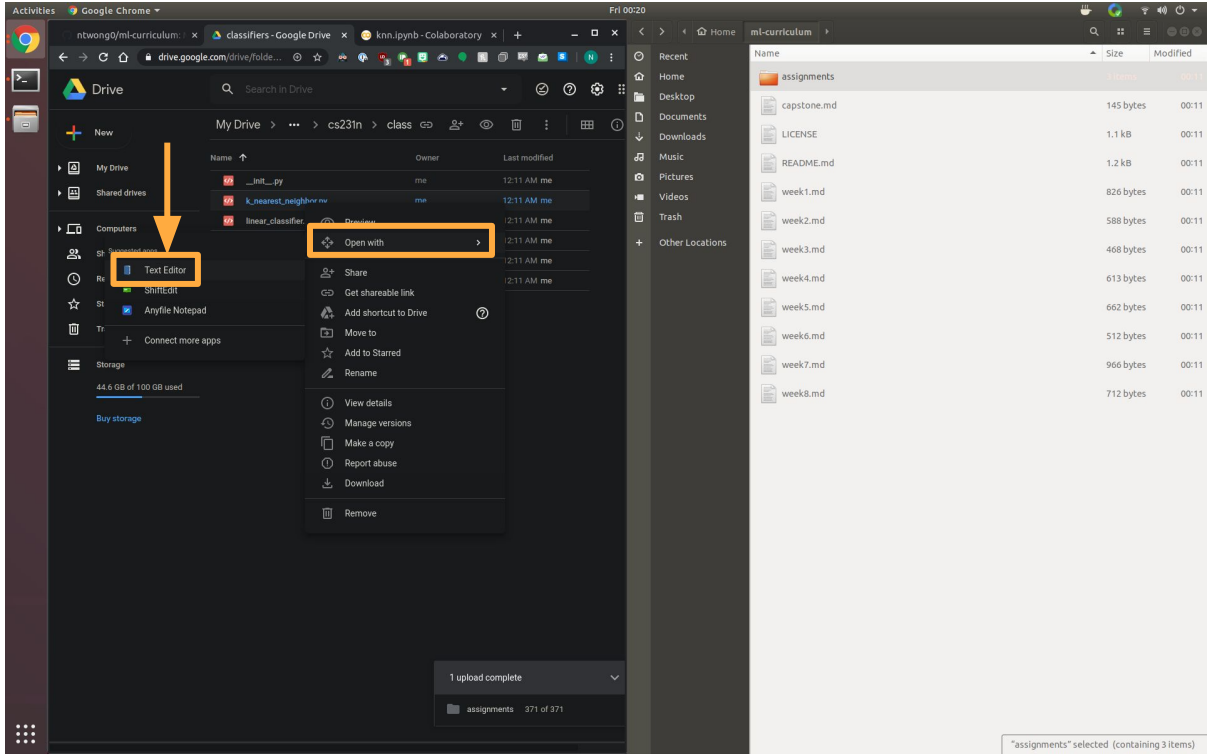
Next, we'll discuss the Python implementation file.

Back in the `module1` Google Drive folder, double-click to open `cs231n`...

...and double-click to open `classifiers`...

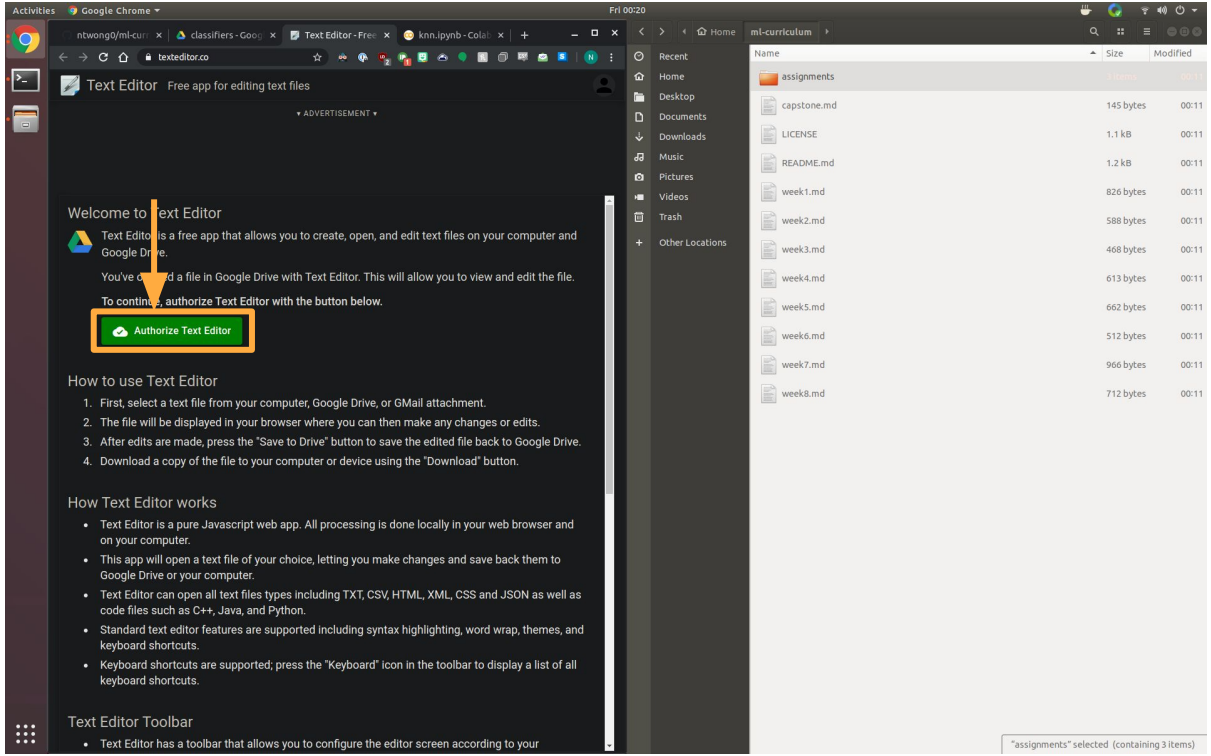
...and right-click on the implementation file of interest. Here, we have `k_nearest_neighbor.py`.

Hover over "Open with", and select "Text Editor".



ml-curriculum

You'll need to authorize Text Editor to proceed. Select "Authorize Text Editor"

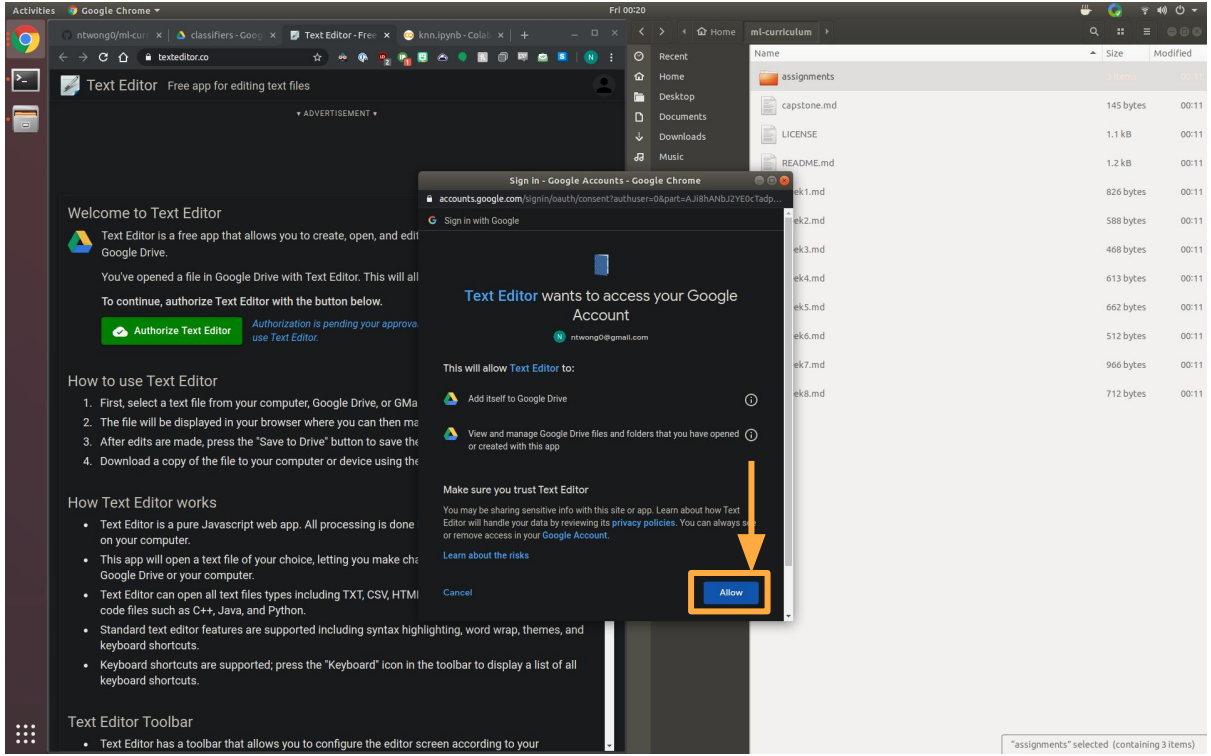


ml-curriculum

You'll need to authorize Text Editor to proceed. Select "Authorize Text Editor"

Then, select your account.

Then, select "Allow".



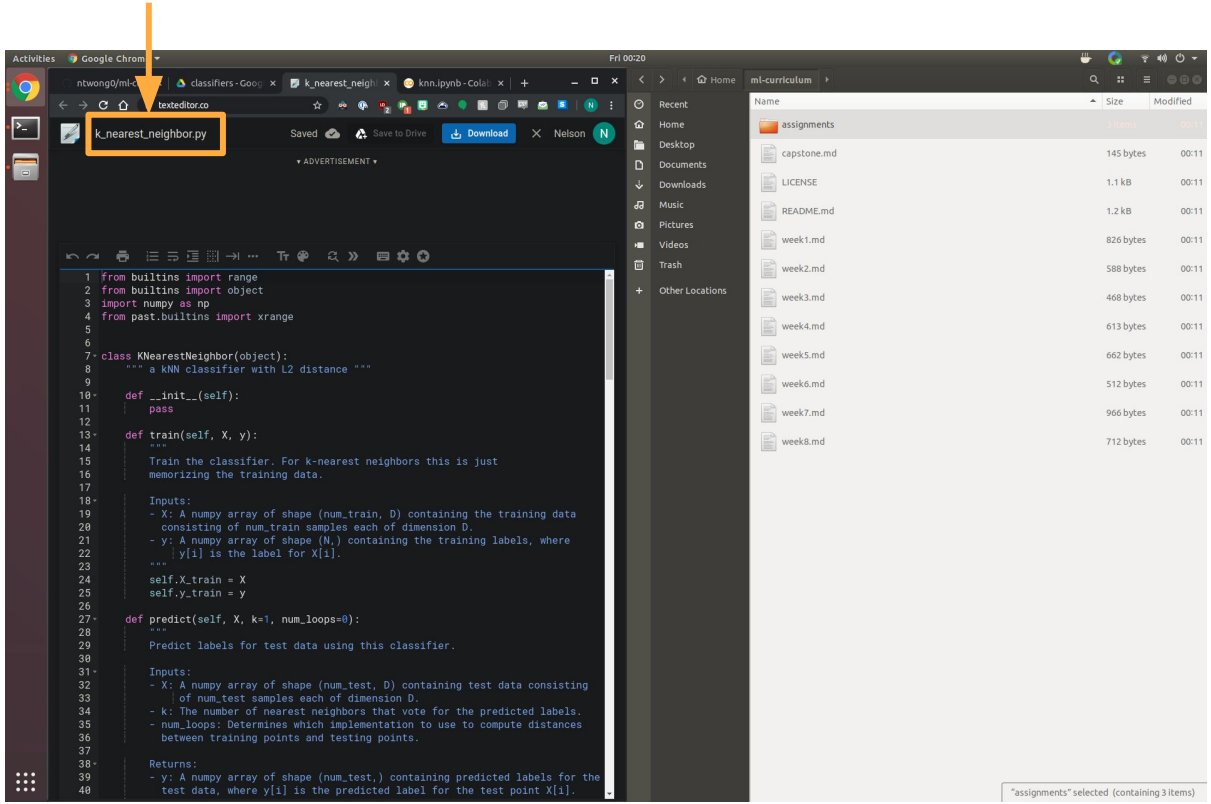
ml-curriculum

You'll need to authorize Text Editor to proceed. Select "Authorize Text Editor"

Then, select your account.

Then, select "Allow".

You are now able to edit `k_nearest_neighbor.py`.



ml-curriculum

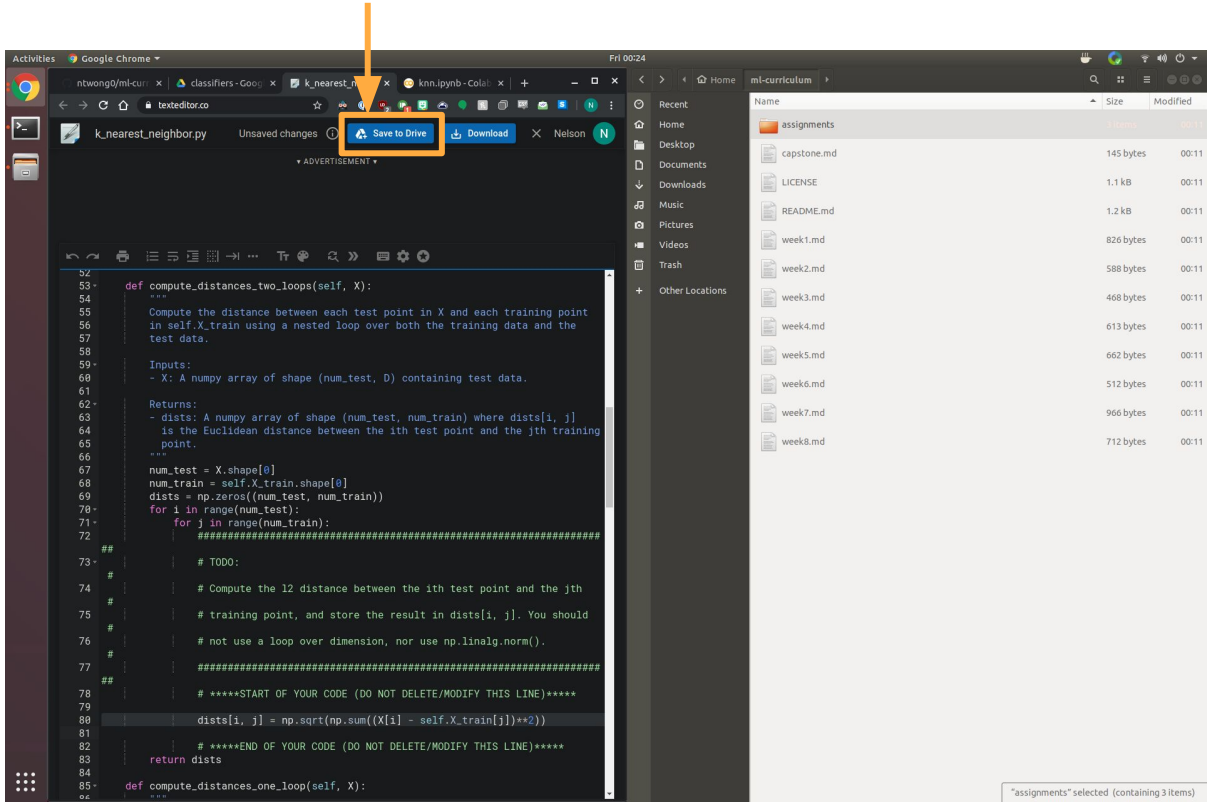
You'll need to authorize Text Editor to proceed. Select "Authorize Text Editor"

Then, select your account.

Then, select "Allow".

You are now able to edit `k_nearest_neighbor.py`.

Remember to save your edits to Drive!

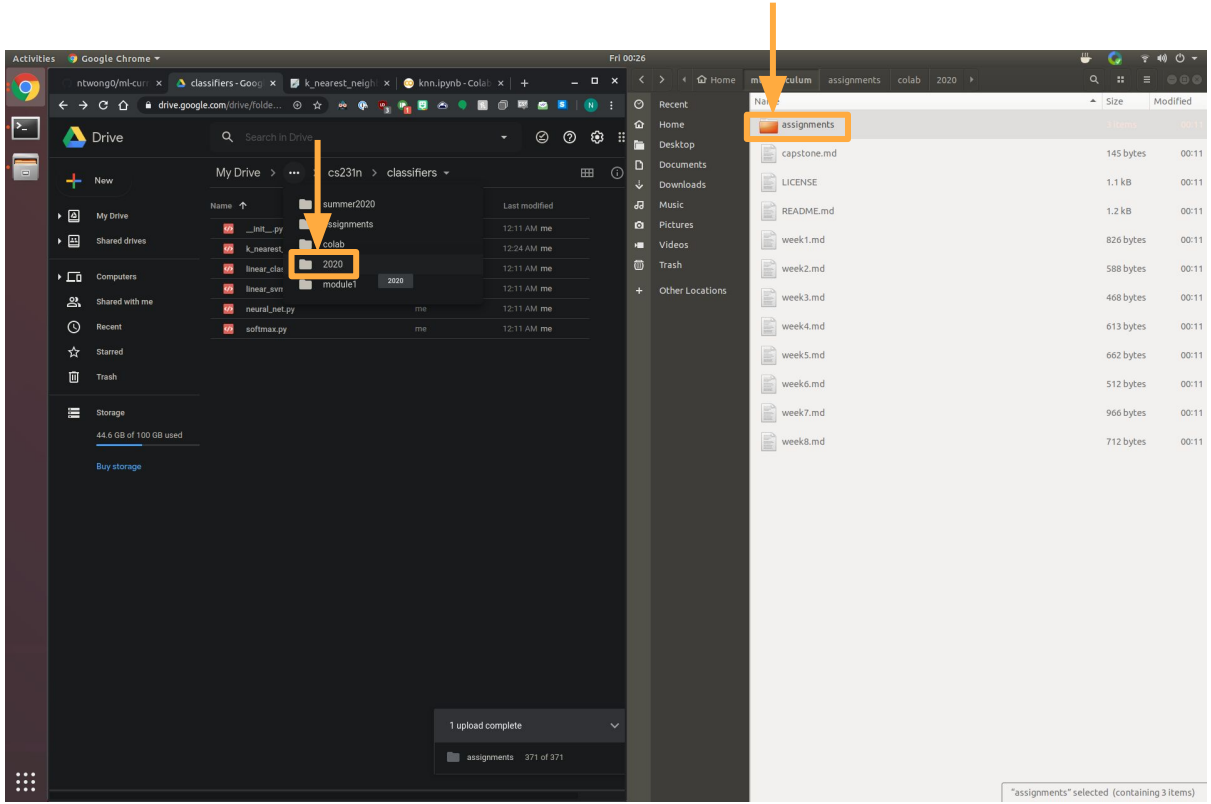


ml-curriculum

Once you're done editing the assignment, you'll wish to publish your work.

First, navigate back to the `2020` folder on Google Drive.

At the same time, double-click to open the `assignments` directory.

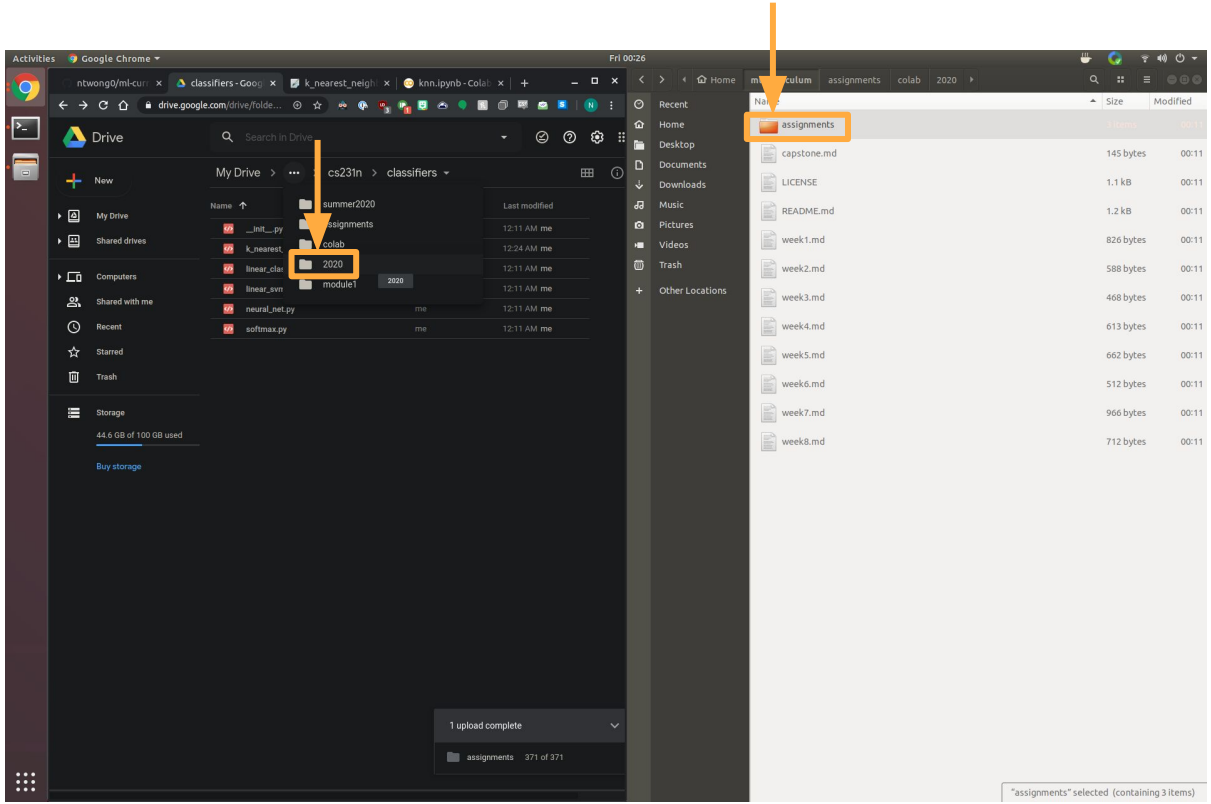


ml-curriculum

Once you're done editing the assignment, you'll wish to publish your work.

First, navigate back to the `2020` folder on Google Drive.

At the same time, double-click to open the `assignments` directory.



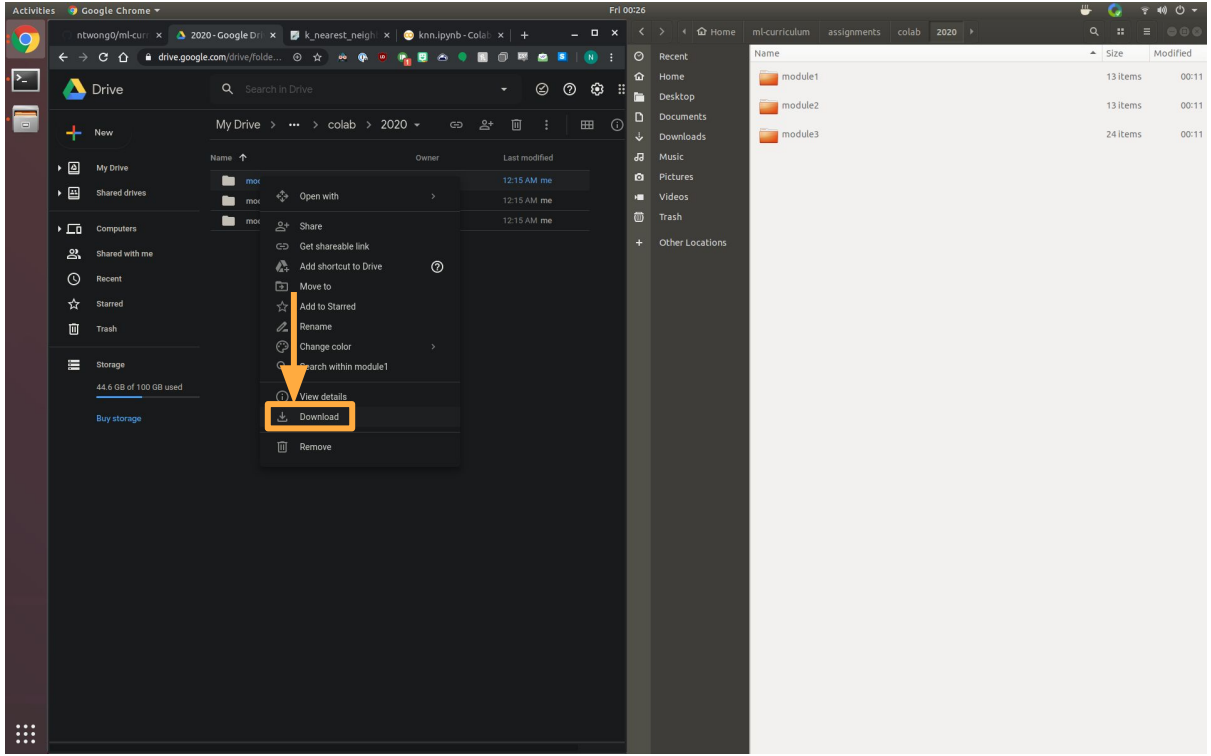
ml-curriculum

Once you're done editing the assignment, you'll wish to publish your work.

First, navigate back to the `2020` folder on Google Drive.

At the same time, double-click to open the `assignments` directory.

Right-click on the module1 folder, and select "Download".



ml-curriculum

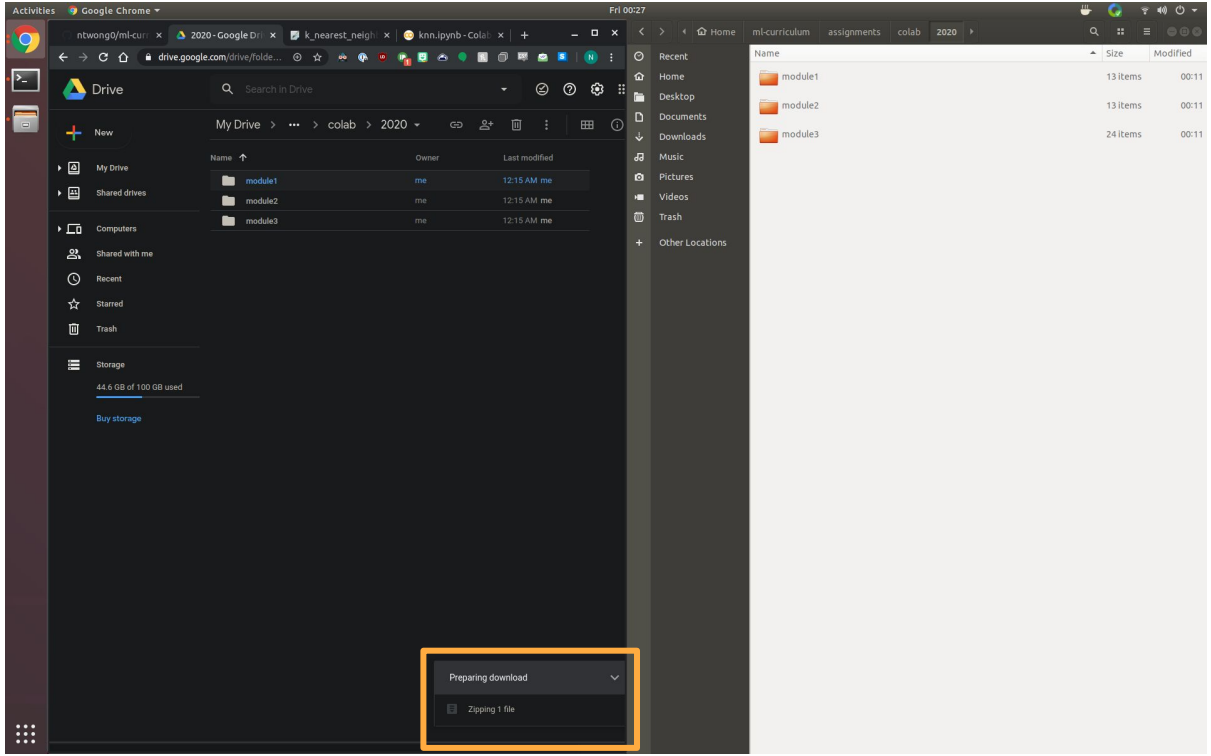
Once you're done editing the assignment, you'll wish to publish your work.

First, navigate back to the `2020` folder on Google Drive.

At the same time, double-click to open the `assignments` directory.

Right-click on the module1 folder, and select "Download".

Wait for the zip file to generate…



ml-curriculum

Once you're done editing the assignment, you'll wish to publish your work.

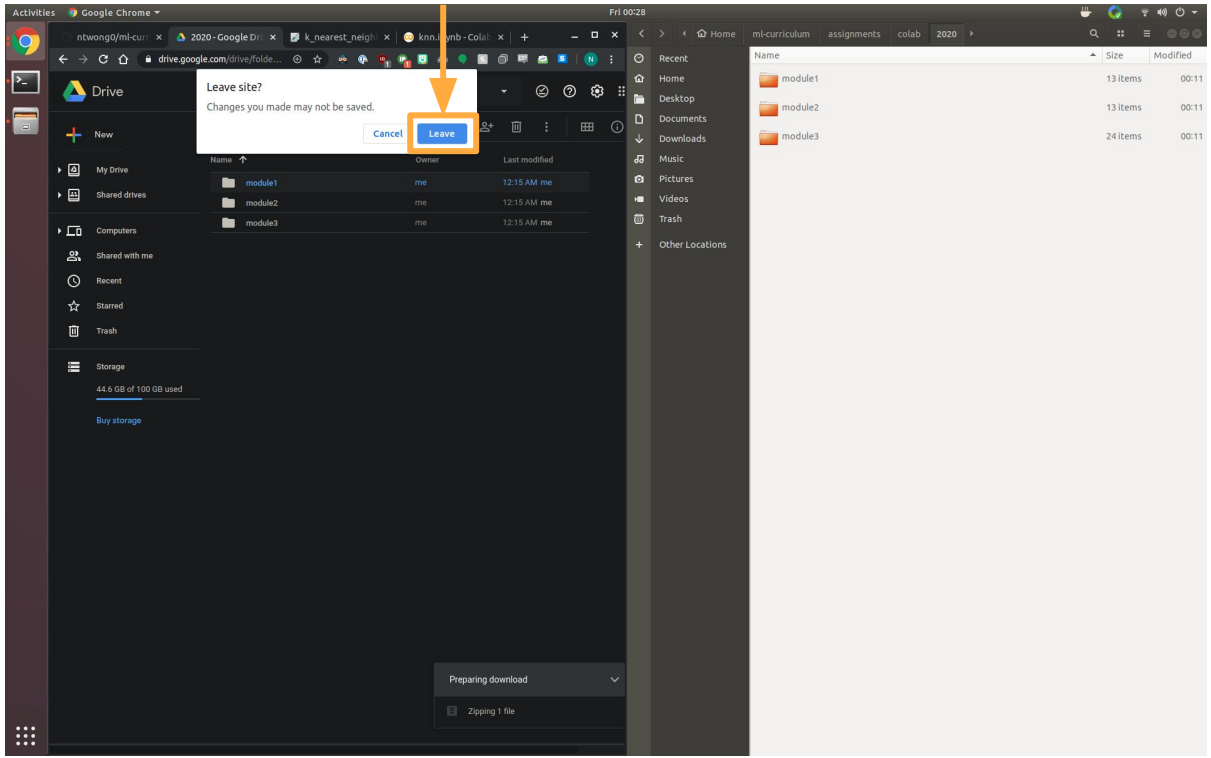
First, navigate back to the `2020` folder on Google Drive.

At the same time, double-click to open the `assignments` directory.

Right-click on the module1 folder, and select "Download".

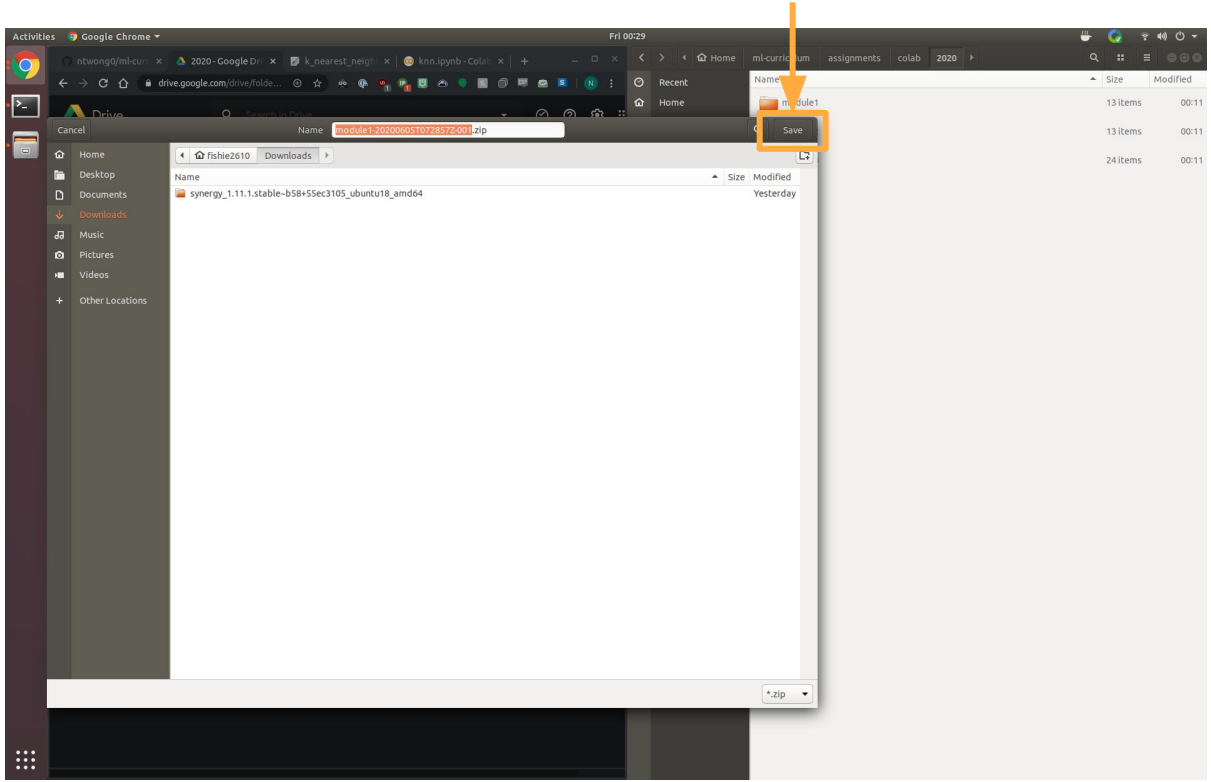
Wait for the zip file to generate…

...then select "Leave" to open the file download dialog.



ml-curriculum

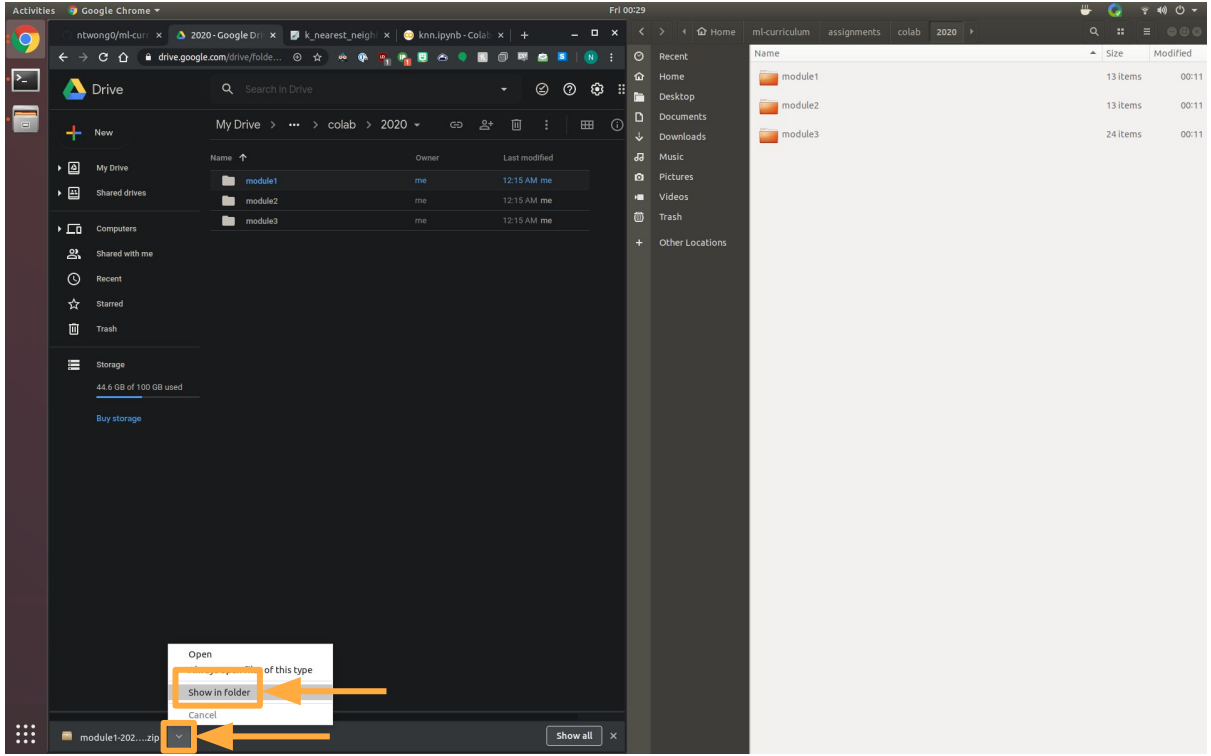
In the following save file dialog box, select the “Save” button.



ml-curriculum

In the following save file dialog box, select the “Save” button.

After the file finishes downloading, select the down-arrow next to the zip file, and select “Show in Folder”

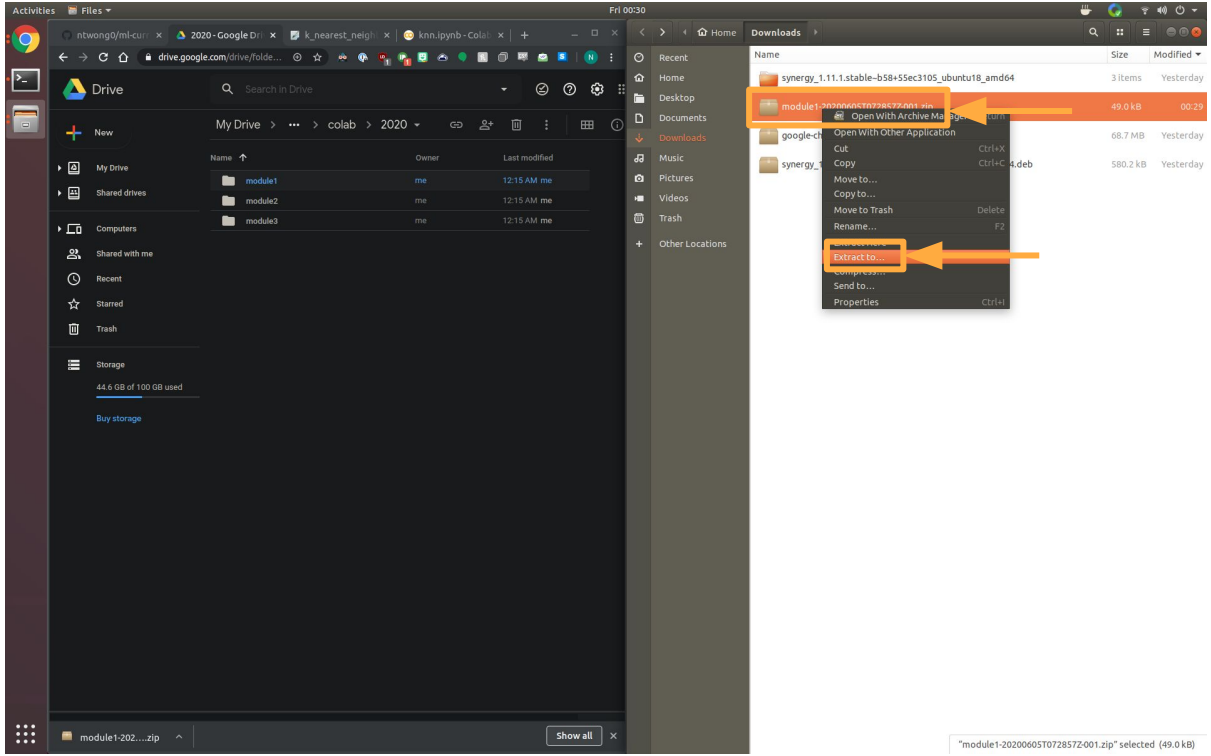


ml-curriculum

In the following save file dialog box, select the “Save” button.

After the file finishes downloading, select the down-arrow next to the zip file, and select “Show in Folder”

This opens up a new File Explorer window. Right-click the `module1....zip` file, and select “Extract to…”



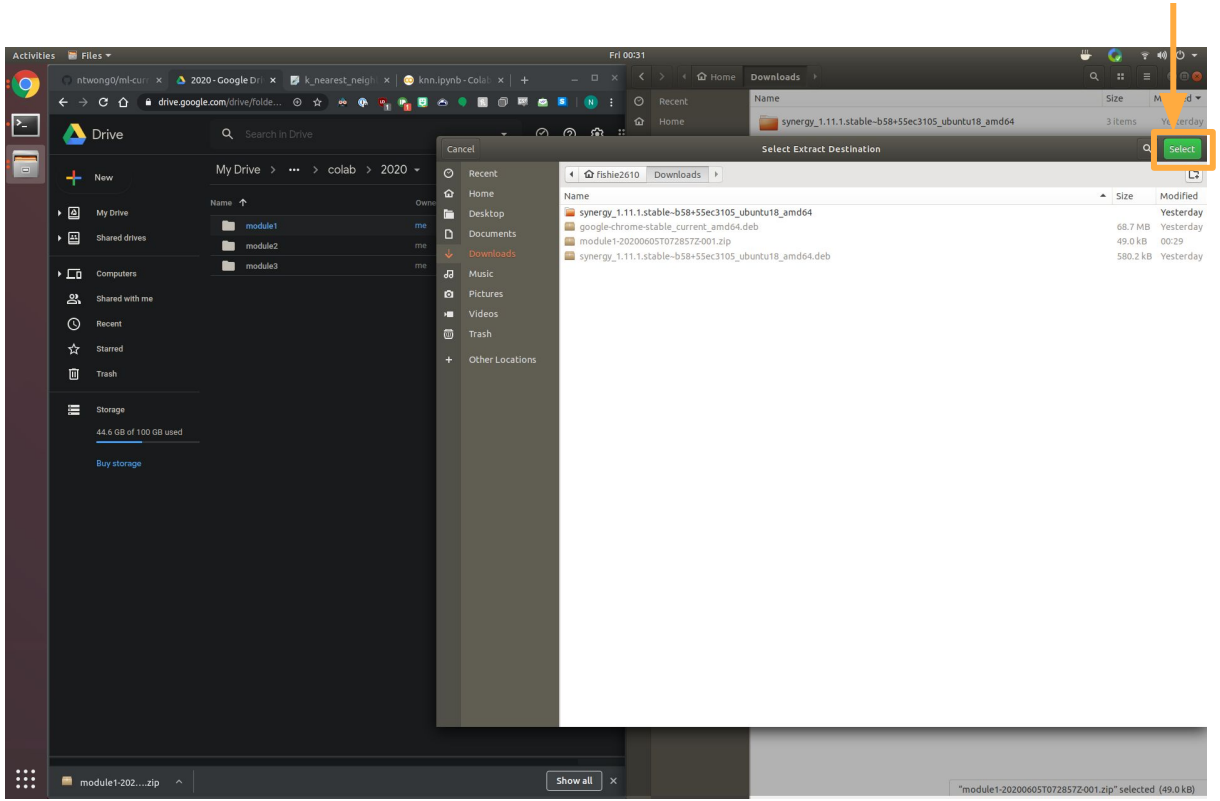
ml-curriculum

In the following save file dialog box, select the “Save” button.

After the file finishes downloading, select the down-arrow next to the zip file, and select “Show in Folder”

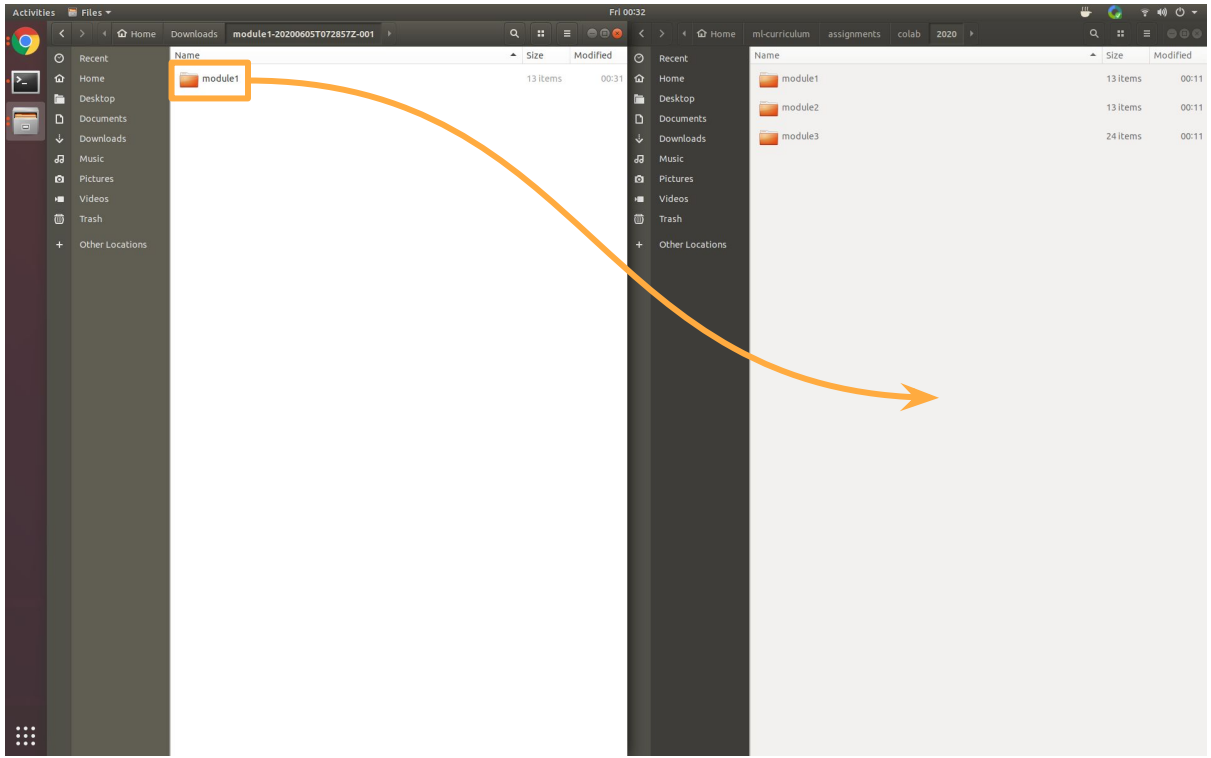
This opens up a new File Explorer window. Right-click the `module1....zip` file, and select “Extract to…”

Extract to the Downloads folder by clicking “Select”



ml-curriculum

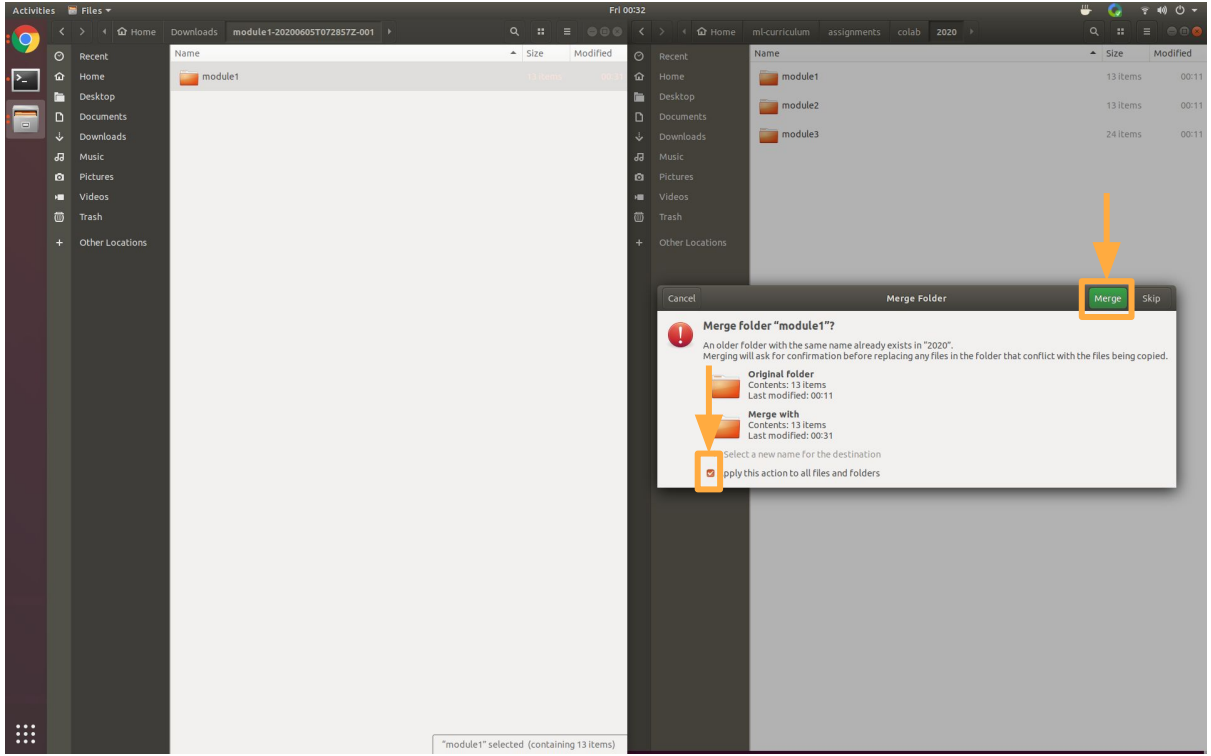
The extracted directory should now be revealed. Show the window side-by-side with our previous `2020` directory. Then, drag-and-drop the extracted module1 directory into the `2020` directory.



ml-curriculum

The extracted directory should now be revealed. Show the window side-by-side with our previous `2020` directory. Then, drag-and-drop the extracted module1 directory into the `2020` directory.

When presented with the Merge Folder prompt, check the box “Apply this action to all files and folders”, then select “Merge”

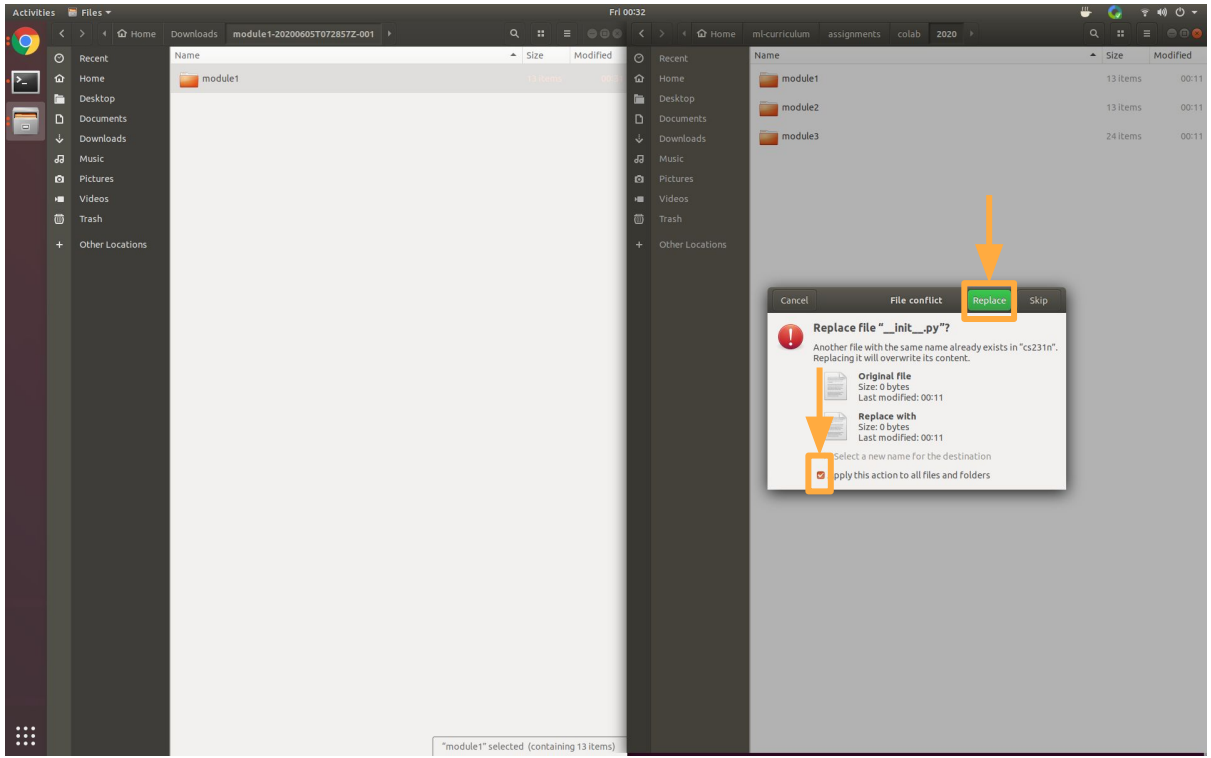


ml-curriculum

The extracted directory should now be revealed. Show the window side-by-side with our previous `2020` directory. Then, drag-and-drop the extracted module1 directory into the `2020` directory.

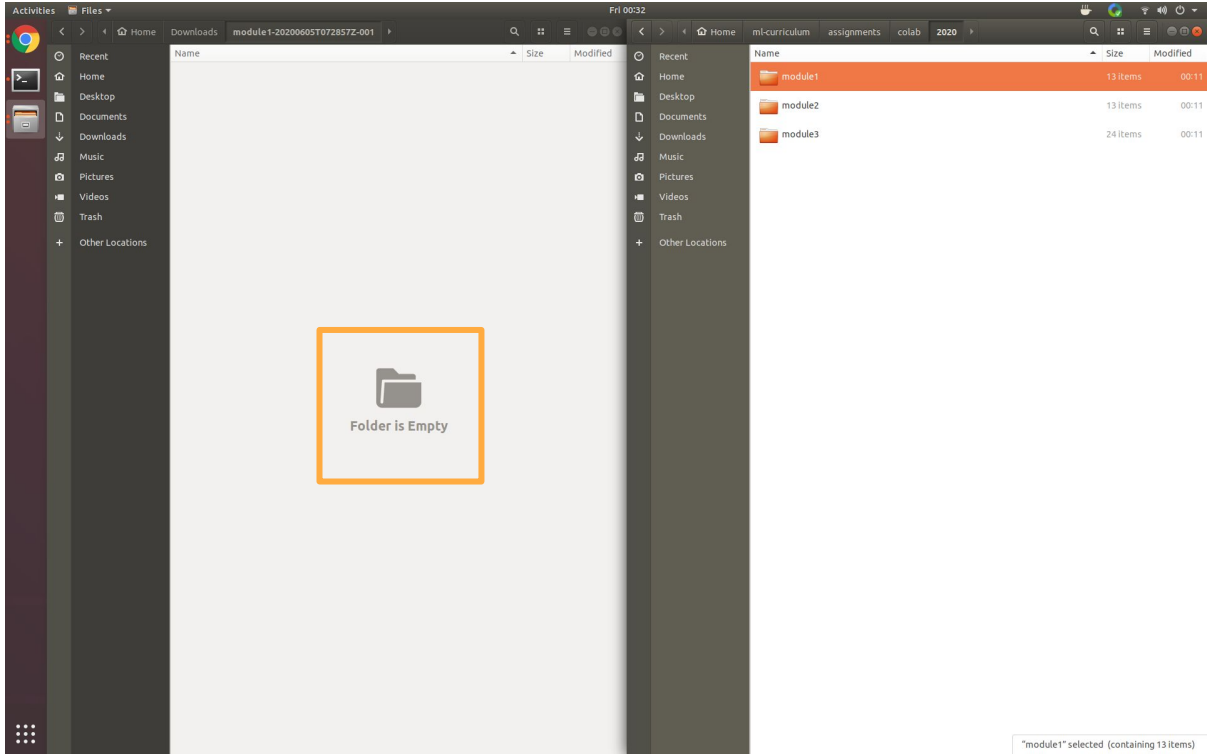
When presented with the Merge Folder prompt, check the box “Apply this action to all files and folders”, then select “Merge”

When presented with the File Conflict prompt, check the box “Apply this action to all files and folders”, then select “Replace”



ml-curriculum

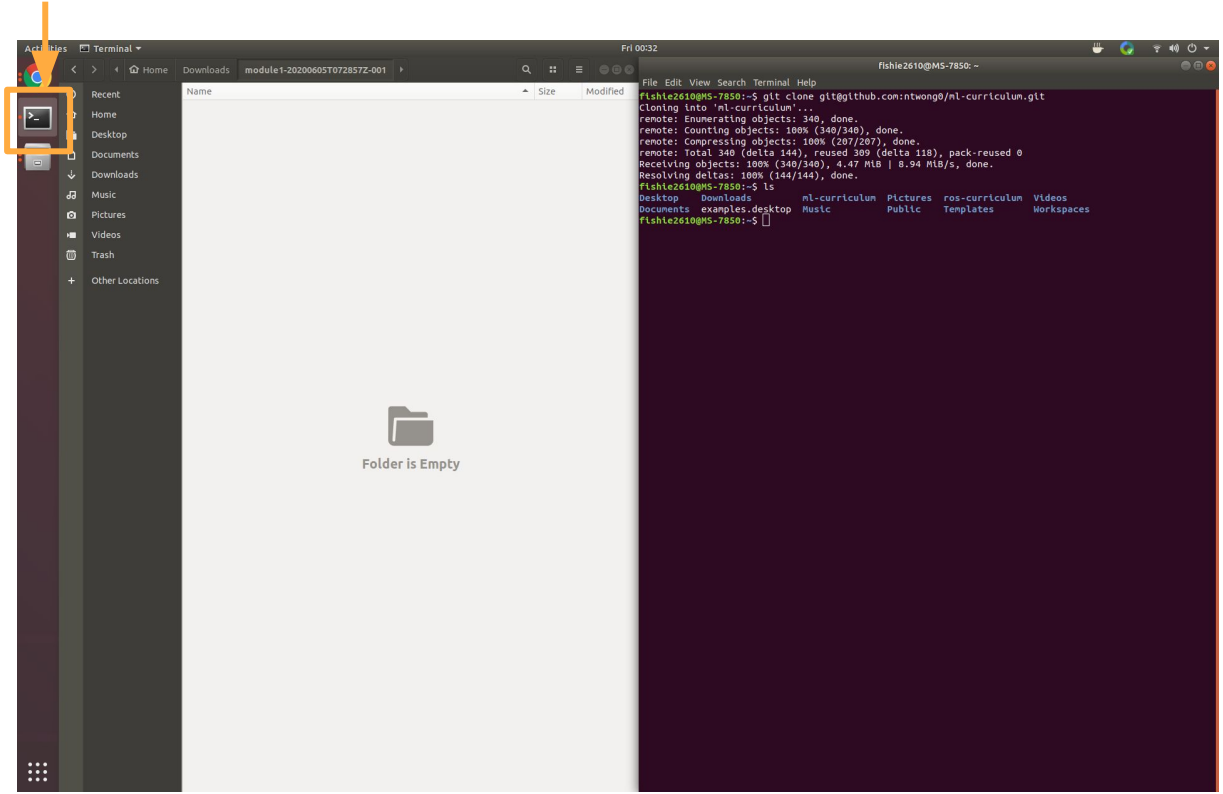
When the drag-and-drop operation is complete, the extracted directory will show as empty.



ml-curriculum

Let’s begin the process of uploading our files to Github.

Select the terminal window from before. If you closed it, open a new terminal window.

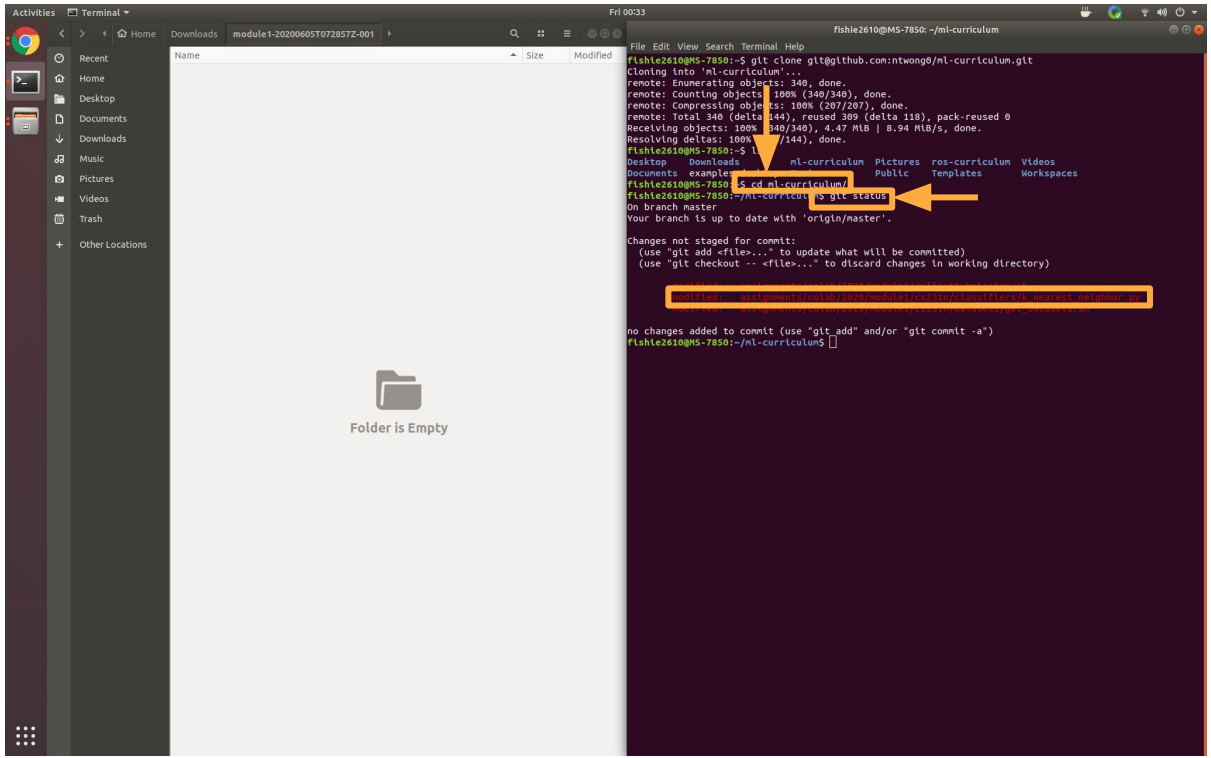


ml-curriculum

Let's begin the process of uploading our files to Github.

Select the terminal window from before. If you closed it, open a new terminal window.

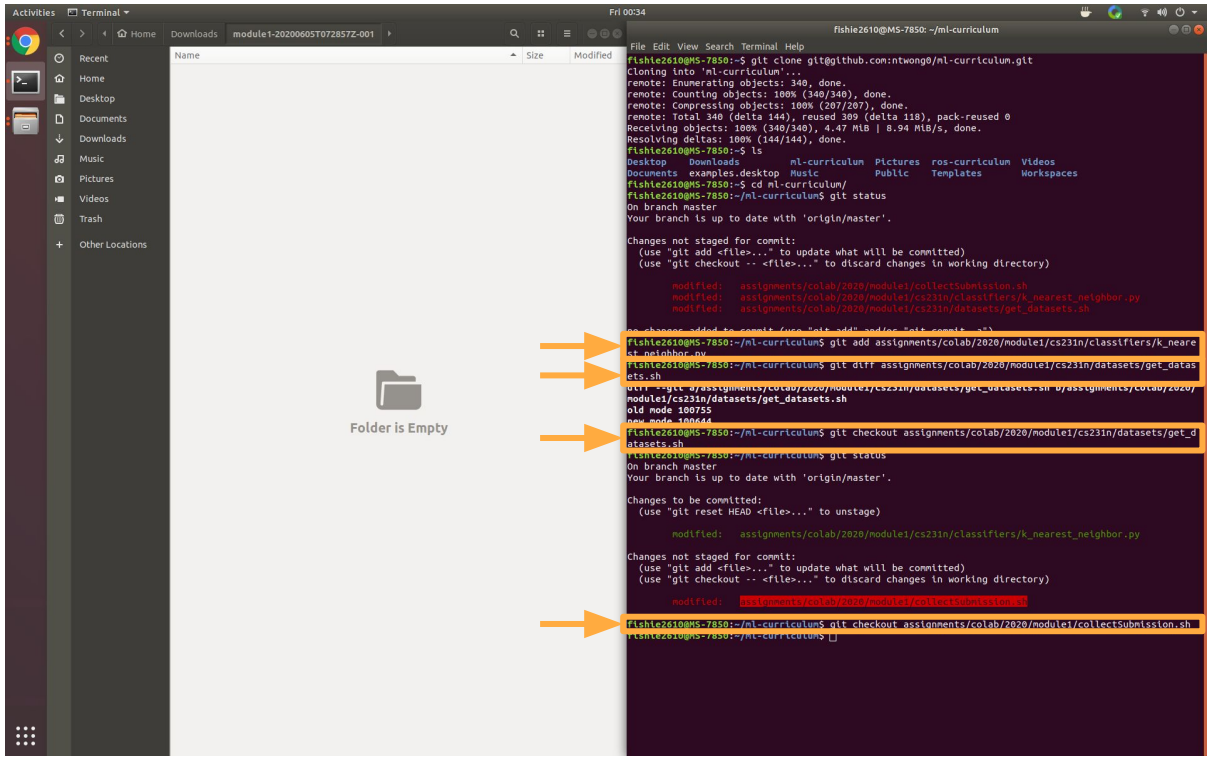
Change into the `ml-curriculum` directory with `cd ml-curriculum`, then use `git status` to reveal our edits. In this case, we see that our `k_nearest_neighbor.py` has changed.



ml-curriculum

If we use `git add`, we can stage our edit to `k_nearest_neighbor.py`

We can also use `git diff` to see our changes. In the case of the two `.sh` files, only the file attributes were changed, and we can reject those changes with `git checkout` to recall a previous version of those files.

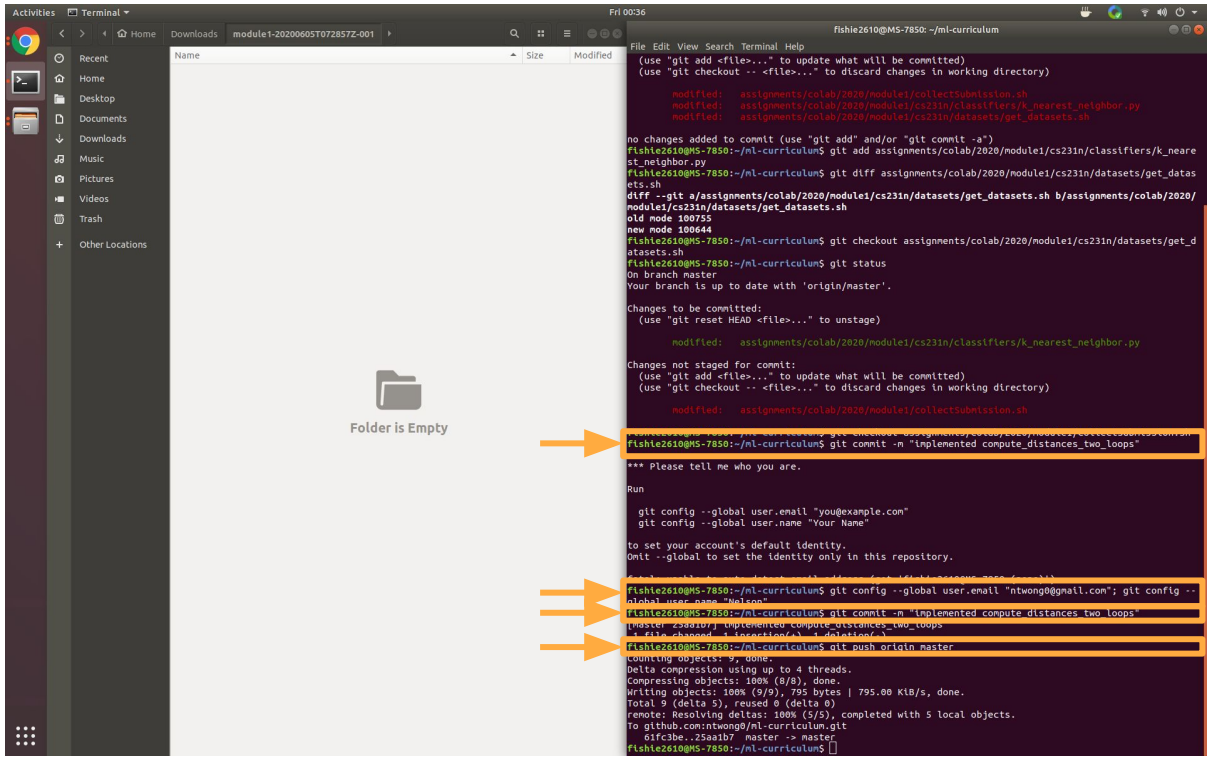


ml-curriculum

With all the changes staged, we are able to commit the changes using `git commit`.

If this is the first time you are committing a change, you'll need to identify your email and name with `git config`.

After you git commit, upload the commit to Github using `git push`



ml-curriculum

If you navigate back to your fork of the ml-curriculum repo, you will find your pushed commit on Github.