Teaching Courses with GitHub and Jupyter Notebooks

Introduction
It is commonplace among industry data science and software development teams to use a version control system such as Git and GitHub to manage their work. By contrast, in most academic programs the university’s Learning Management System (LMS) is the primary tool used for distribution and collection of student work on assignments and projects. In order to provide students exposure to the “tools of the trade” and establish a level of comfort in working with a version control system, GitHub can be used as an alternative method to an LMS for managing student work. GitHub provides a service called GitHub Classroom which makes it easy for educators to set up their course to make use of GitHub. The primary function of GitHub Classroom is to manage the automated distribution, collection and return of assignments to each student on a class roster.

There are multiple ways to utilize GitHub Classroom together with an automated grading system, for example by setting up continuous integration with a tool such as Travis CI. Among data science instructors, creating assignments in Jupyter Notebooks is a popular option. A tool called nbgrader can be used to automate grading of student notebooks.

The following instructions will walk through setting up a course using GitHub Classroom for assignment distribution/collection together with nbgrader for automated grading of assignments. A package called abc-classroom is used to further automate the connection between GitHub Classroom and nbgrader.

The high-level steps in the process are:
1) Set up the course as a GitHub master organization (which persists across semesters)
2) Set up a GitHub Classroom for the particular semester of the course, containing the student roster for that semester
3) Create assignments using nbgrader with test cells for auto-grading
4) Distribute assignments via GitHub Classroom so that each student receives their own private repo with the assignment and can manage their own version control
5) Automated collection via GitHub Classroom of each student’s assignment repo at the assignment deadline
6) Automated grading of assignments with nbgrader
7) Return of graded assignments to each student using GitHub Classroom

Initial Set Up Steps (FIRST TIME ONLY)
1. Register for GitHub (https://github.com) and GitHub Classroom (https://classroom.github.com)
2. If you have a Mac, ensure you have Command Line Tools Installed (which includes Git)
   a. $ xcode-select --install
3. Install nbgrader
   a. $ conda install -c conda-forge nbgrader
4. Install abc-classroom
a. `$ pip install abc-classroom`
b. Check out the docs at https://abc-classroom.readthedocs.io/en/latest/get-started.html

5. Create a GitHub personal access token and store locally in file `.abc-classroom.tokens.tml`
a. `$ abc-init`

6. Follow directions at https://help.github.com/en/github/authenticating-to-github/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent to set up a GitHub SSH key, add it to the SSH agent, and then add it to your GitHub account
a. If using Mac, write down your passphrase for key somewhere as backup

Set up a new course
1. On GitHub
   a. First step is to set up a master “organization” for the course on GitHub (e.g. test-course)
      i. This will be for the overall course, and then you can create a new “classroom” (see below) for each semester of the course that you teach which pulls from the master organization (each classroom has its own roster of students). Select a name that relates to the course, not the semester (e.g. “CS101”). It should belong to your personal account
   b. FIRST TIME ONLY - Go to https://education.github.com/discount_requests/new and request the educational discount for the “organization” (NOT individual) to be able to create unlimited private repos for the organization
   c. FIRST TIME ONLY - Once it is approved, go to https://education.github.com/toolbox/offers/github-org-upgrades to upgrade your new master course organization so you can create unlimited private repos (in order to keep all student assignment submissions private)
   d. Go to https://github.com/settings/connections/applications/64a051cf1598b9f0658f and allow GitHub Classroom access to the master course organization

2. On GitHub Classrooms at classroom.github.com
   a. Create a new classroom for section of course (e.g. CS101-fall2020)
   b. Connect classroom to course master organization

3. On local computer
   a. Cd to desired folder on computer
   b. Create course directory using abc-quickstart
      i. `$ abc-quickstart <course-name e.g. CS101 >`
   c. Cd to the new course directory `<course-name>`
   d. In it, create `<nbgrader_dir>` directory using nbgrader:
      i. `$ nbgrader quickstart <nbgrader_dir>`
      ii. In the “source” folder delete the ps1 folder (contains sample notebooks)
   e. Update the following in config.yml:
      i. Set name for GitHub organization to connect to
      ii. Change path to course materials to `<nbgrader_dir>`
iii. Optional – change README text which shows up for students in their assignment repos

**Add course syllabus and class materials**
1. In your master course organization on GitHub, create a new repo called something like “Course materials”. Either make it public, or make it private and add the students’ GitHub usernames as members to it
2. Upload your class materials into the repo on GitHub, or put it in local folder and push to GitHub
   a. To push it to GitHub from your local computer:
      i. Create a folder in your main `<course-name>` directory for your course docs (syllabus, slides etc)
      ii. Organize your materials as you like in that folder (can create sub-folders for each week if you wish)
      iii. In your GitHub course organization, create a public repo and name it the same as the folder you just created
      iv. On computer, cd to the folder you created
     v. Run the following to push the content to GitHub organization:
        1. $ git init
        2. $ git add –all
        3. $ git commit -m “update”
        4. $ git remote add origin <url-of-github-repo>
        5. $ git push -u origin master
   vi. You should now see all your course docs in the new repo you created

**Create roster for classroom**
1. In classroom, click on “Students” tab
2. Create list of student identifiers (use student names, university IDs or email addresses as the identifier for each student) from roster and upload as csv or input manually
   a. Alternatively, if you use a LMS (e.g. Sakai) you can connect GitHub Classroom with your LMS and automatically import your roster
3. When student receives link to an assignment for the first time, they will need to select their identifier from the classroom roster which maps their GitHub username to their identifier on your roster

**Creating an Assignment**
1. Build the assignment locally on computer with nbgrader
   a. Build the assignment using nbgrader
      i. Create a `<nbgrader_dir>/source/<assignment name>` directory for the new assignment. For the assignment name, use only lowercase and no special characters
      ii. Create the jupyter notebook and use nbgrader tools by selecting View -> Cell Toolbar -> Create Assignment
iii. Instructions for nbgrader at
https://nbgrader.readthedocs.io/en/latest/user_guide/creating_and_grading_assignments.html
b. Generate student version of the assignment in the <nbgrader_dir>/release directory:
   i. Cd to <nbgrader_dir>
   ii. $ nbgrader generate_assignment <assignment name>

c. Add any other needed files to the student version of the assignment (e.g. csv files),
   found in the <nbgrader_dir>/release folder

2. Create the assignment template repo in GitHub master organization (so it can be reused
   later for additional classrooms / sections of the course you teach)
a. Create new assignment template repo
   i. Cd out to <course-name> directory and run the following
      1. $ abc-new-template <assignment name> --github

b. On GitHub
   i. Go to master-organization repo and navigate to assignment template repo
   ii. In Settings:
      1. Check “Template repository”
      2. Change visibility to private repo

3. Create assignment on classroom.github.com and link to template repo:
a. Create new assignment and name it with same <assignment name>. Make sure the
   “custom repository prefix exactly matches the <assignment name>
b. Add a deadline
   c. Select individual or group assignment
d. Select “private repos”
e. Click “add starter code” and link assignment to template repo you just created
   <master-course-organization>/<template-repo-name>
f. Choose “template repository” as import method
g. Send assignment link to students
   i. Students must pull assignment from GitHub repo, work on it, and push
      changes back to the repo before the deadline

How to Handle Large Data Files
1. If your data files are too large to store on GitHub, you will want to locate them in cloud
   storage and obtain a public link to download them (e.g. Box or Google Drive)
2. Add a code cell to the top of your Jupyter Notebook to download the files
3. Add a .gitignore file into your <nbgrader_dir>/source/<assignment name> repository so
   that the data files do not get uploaded to GitHub (e.g. so when the students upload their
   solutions, it does not try to upload the large data files with them)

Editing an Assignment Before Distribution to Students
1. Make your changes in the <nbgrader_dir>/source/<assignment name> directory
2. Generate student version of the assignment in the <nbgrader_dir>/release directory:
   a. Cd to <nbgrader_dir>
   b. $ nbgrader generate_assignment <assignment name>
3. Cd out to <course-name> directory and run the following
   a. $ abc-new-template <assignment name> --github --mode delete

Collecting & Grading Assignments
1. DO THIS FOR FIRST ASSIGNMENT ONLY
   a. Navigate to the classroom on classroom.github.com, go to the “Students” tab and click “Download” to download the student roster. Place it as “roster.csv” (might need to change the file’s name) into your local <course-name> directory
   b. Add your GitHub ssh key to the ssh-agent and store the passphrase following the instructions at https://docs.github.com/en/github/authenticating-to-github/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent#adding-your-ssh-key-to-the-ssh-agent. This is used to authenticate with GitHub when cloning the repos back to your computer
2. After assignment deadline, run the following script from the <course-name> directory to collect all submissions. This first clones all student assignment GitHub repos into the clone_dir directory (specified in config.yml). It then copies all files into the <nbgrader_dir>/submitted directory (where it needs to be located to run the nbgrader autograder)
   a. $ abc-clone <assignment name-as-shown-in-student-repos> Note: the assignment name used for the student repos created by GitHub Classroom may be different than the originally created assignment name so be sure to check the name used in the students’ repos
3. Use nbgrader to autograde the assignment
   a. Cd to <nbgrader_dir> directory
   b. Run the following script:
      i. $ nbgrader autograde <assignment name> --force
4. Add any manual grading
   a. From the <nbgrader_dir> directory, launch Jupyter Notebooks ($ jupyter notebook) and go to Formgrader tab
   b. Open student notebooks from ‘autograded’ folder in <nbgrader_dir>
   c. Select “Manual Grading”, click on assignment, click on notebook, and complete manual grading for each student
5. Generate feedback for students
   a. Cd to <nbgrader_dir> directory and run the following
      i. $ nbgrader generate_feedback <assignment name>
6. Push feedback back to students via GitHub
   a. Cd to <course-name> directory and run the following:
      b. $ abc-feedback <assignment name> --github

Exporting Gradebook to a CSV
1. Navigate to <nbgrader_dir> local folder (where gradebook.db is located) and run:
   a. $nbgrader export
Student Workflow

4. Students must create their own GitHub account
5. Students receive link to first assignment from instructor
6. Students select the university identifier they correspond to from the list input by the instructor (either their name or email address, depending on what you used as identifier)
7. Student clicks “Accept this assignment”
8. To complete assignment:
   a. Create folder on local computer and cd into folder
   b. Clone assignment repo to local
      i. $ git clone <assignment-repo-url>
   c. Complete assignment on local computer using Jupyter Notebooks
   d. Push changes to remote assignment repo
      i. $ git add –all
      ii. $ git commit –m “<type-message>”
      iii. $ git push
   e. Students can update their submission as many times as they like prior to the deadline by repeating the above steps
9. After grading, feedback will be pushed to student’s repo
10. Student can pull to local computer or download the zip folder, and click on <notebook-name>.html to view the grades and feedback