STA 445 Special Topics - GitHub

Dr. Robert Buscaglia

November 15, 2023

Git and GitHub

Today's lecture will introduce you to GitHub. GitHub is a web-based platform that is used primarily for version control and collaboration in software development projects. GitHub allows you to work together with collaborators to update work together in real-time, tracking changes to versions and the code base, and allows for efficient management of software versions. Git specifically is the distributed version control system and is affiliated with any particular service like GitHub. Github is a hosting system that allows for functionality of Git to be done through a server-based cloud with easy to use interfaces. Some of the main ideas behind the use of GitHub (Git) include:

1. Version Control
2. Repositories
3. Collaboration
4. Branching and Merging
5. Pull Requests
6. Issue Tracking
7. Community and Open Source
8. Security and protection

GitHub is not all about software management though. GitHub is used frequently for other elements of work including developing documentation, performing data analysis, and storing and sharing data. In future chapters, we will try to incorporate GitHub to host R packages that can be used to share functions and data easily with others. This is something I do commonly with students and computer-oriented collaborators.

Getting Started with GitHub

We will start off with the basics of signing up for GitHub. Lets work to get you setup with your own GitHub for use in this course or your own personal work. Preparing a strong GitHub account can be vital to landing a job in any data or software position, and is a great way to store your own work somewhere that is easily accessed elsewhere.

Exercise 1 - Prepare a GitHub Account

Please visit https://github.com/ and prepare an account. The steps for doing so are below and we will take in-class time to prepare this.

1. Navigate to https://github.com/ homepage
2. Click `Sign-up` on the upper right.
3. You will be brought to an interesting GPT-like interface - enter your email

- When entering an email, choose what account you want your GitHub to be linked to carefully.
- I use my own personal email in case the work is not related to NAU
- I have no requirements on this side - other courses might!

4. Enter a password.
5. Create a username

- Usernames can be changed after creating an account.
- Changing a username though can be very difficult or detrimental if others use your work.
- I tried changing my username and almost lost several working repositories!

6. Want ads? (I said no)
7. Solve a puzzle.
8. Submit your account and verify by email.

- You will need to verify the account by email before using.

Congratulations! You should now have a GitHub account ready for use. If you previously had a GitHub account and would like to continue using it, please take the time while we are setting up to ensure you can sign-in and use your GitHub account.

After account creation, GitHub has modernized with many prompts to help you get the most out of the experience. This is something you can choose to tailor yourself. I for instance was offered an upgraded account due to my teaching status. You may receive offers as students. Please do not purchase anything for this class! A free account is more than enough to accomplish the tasks ahead.

User Profile

Lets take a little longer to ensure our profiles are useful and updated. A user profile can be important for getting noticed. GitHub may prompt you to continue setting up your profile, or select 'Your Profile' from the drop down bar from the upper right corner (your GitHub icon). Lets take a moment to get this updated.

Exercise 2 - Update your GitHub profile information

Please take a few moments to update your information. I strongly encourage adding a picture at some point, this is how you get recognized.

Creating a Repository

A **repository** refers to a location where files and the entire history of changes to those files are stored. It acts as a central hub for a project, containing all the resources necessary for the project's development. We will use repositories to store some of the upcoming work in STA 445. There are many aspects to a repository, which you will work on below. For now, lets work on setting up our first repository together.

Exercise 3 - Create your first Repository

Please setup a repository named 'MyFirstRepository' following the steps below. You may also want to refer to the GitHub tutorial found here (GitHub QuickStart Guide). You will be doing more with this later.

1. From your profile page select Repositories at the top of the page.

- If you had other repositories, you may see them here. They can be sorted and searched using the search bar at the top of the page.

2. Select 'New' from the right of the search bar.
3. Name the repository 'MyFirstRepository'.
4. Add a short description.
5. Ensure the profile is set to Public.
6. Initialize the repository with a README file.
7. We have no .gitignore or license needs (ignore these options).
8. Review the rest of the page before choosing 'Create Repository'.

When complete you will be brought to the front page of the repository and see a README file. This is a markdown file and works with very similar syntax to our R-markdown files. You may use latex within the documents and can write important descriptions about your package.

Commits and Branches

When you work on the exercise below you will encounter many of the aspects of working within a distributed version control system (DVCS). After following the prompts below, you will be asked to **commit** your changes to the repository. Committing changes refers to saving the current state of modified files in a repository. When you commit changes, you are essentially creating a checkpoint that records the modifications made to files since the last commit. Each commit represents a snapshot of the project at a specific point in time.

When we commit the changes below, we will save our changes to the `main` branch. **Branches** are separate lines of development within a repository that diverge from the `main` codebase. They allow developers to work on features, bug fixes, or future content experiments without affecting the main code until changes are ready to be merged. Branching allows for independent or parallel workflows and development from collaborators or within teams, feature development outside of the published code, and isolating major code changes within identified branches.

Exercise 4 - Edit the repository README

1. Select the pencil on the README.md file.
2. Keep the repository Header

- Be sure to put an extra line of space between sections, just like we do in RMD files.

3. Add your name as the sub-header (two #)
4. Add the course number (STA 445) as the sub-sub-header (three #)

5. Add any additional description you would like after the name of the course.
6. Select 'Commit changes…'

- Here you are allowed to write messages and description for the changes you produced. We are not changing anything critical at this time, but it can be important to give adequate descriptions to your changes. This can help both yourself or other collaborators. I have had several occasions where my descriptions helped me remember what tasks I had completed.

7. Use the default message and description.
8. Ensure you commit to the main branch.
9. Commit Changes.

A few files

In Chapter 16 we will make R packages, and we will work to try to make these packages downloadable directly from your own GitHub pages. For now, lets add a few files to our first repository so they aren't empty.

Exercise 5 - Add your Homework RMD and PDF files in two directories.

This exercise will require you to do some manipulation of the files on your computer. Please create two folders, one named `STA445_RMD` the other `STA445_PDF`. Within the `STA445_RMD` folder, please place any RMD file (or files) from this course. If you want to edit or mask any of your work you may do so. I will be looking that the file exists in this particular directory. Within the `STA445_PDF` file, place the corresponding PDF generated by the RMD files in the first directory.

To add the files to `MyFirstRepository` on GitHub

1. Select the repository you want to add the files.
2. Select 'Add file' from the top of the repository's page.
3. Select 'Upload files'.

- You can choose to write files directly within this environment. This can be helpful, such as when creating .css files for style and control of the GitHub page. I rarely use this feature though, as I write most code first in my preferred environments, then upload them to my GitHub.

4. I would suggest *dragging the two folders created above* directly on the select files area of the screen.

- You should see all items to be added, including the directories.

5. Add a commit change description. You do not have to add an extended description.
6. Commit changes to the main branch.

You should now had a README.md file along with the two requested directories above. Ensure that all files required are present in the right area.

A few comments on GitHub Desktop

All of the work we just did above is sometimes referred to as 'pushing' new information to the repository. Since we are the owners and creators, there was no real 'push' that occurred since we accepted the changes immediately. These terms are referred to frequently when using Git.

- **Push:** Sends your local changes to the remote repository.
- **Pull:** Retrieves changes from the remote repository to update your local repository.

For now, you can consider the files we uploaded as a 'push' of these directories to the repository, which we committed to the main branch. Handling this type of work can be done more efficiently through a GUI. I commonly use the GitHub Desktop application to handling my pushing, pulling (downloading the repository), and tracking my commit changes (branches, forks).

If you are using a personal machine, I encourage you to look into the GitHub Desktop app, found by clicking here GitHub Desktop. Since we are learning about GitHub and Git, all of the code used to create GitHub Desktop can be found here GitHub Desktop Git page.

GitHub desktop can then be linked to your GitHub account. Files can be pushed and pulled to repositories easily through the interface. I try to commit my working changes of a project when I finish any major updates. Then my work is successfully backed up to my GitHub profile.

Forking

The last task we will complete before doing one final tutorial on your own will be to fork a project. **Forking** refers to creating a personal copy (or clone) of another repository into your own account, thereby allowing you to freely experiment with the codebase without affecting the original project. This is often how you get started on working within a project, is to fork the `main` codebase from a Git and create changes to the files within.

Exercise 6 - Fork

We will fork the repository `DrBuscaglia/ForkMe`.

1. Navigate to the GitHub repository for `DrBuscaglia`.

- There are many ways to find different pages, you search for this user, you may type in a url, explore a bit on your own.

2. Select the `ForkMe` repository.
3. Select Fork. To the right of the name of the repository, you will see several options (Follow, Fork, Star).

- Watch - allows you to see changes to that repository on your GitHub homepage.
- Fork - what we're doing now; make a clone repository in our own profile.
- Star - highlights that repository as important

4. Keep all options as is, but you should see it moving to your GitHub profile.

5. Create fork.

The project should now be available within your own GitHub profile. You would then be ready to start making changes within your forked branch. When you had completed changes, you could *push* those changes back to the *main* branch for someone to review and *commit*.

Tutorial

We will finalize your understanding of the concepts above, and expand a bit beyond my descriptions, by completing the HelloWorld repository tutorial.

Exercise 7 - HelloWorld

Please complete the HelloWorld tutorial within the GitHub quick start guide. You can find the link here GitHub QuickStart Guide.

Conclusion

I hope this introduction helps you get started using GitHub. It is a very useful platform and a great way to show off how much you have done with data and code. I encourage you to expand your GitHub repositories during your NAU studies, where it could then be used as an important link on your resumes and job applications.

We will work with GitHub again in the next assignment!

Disclosure

This document was prepared with the aid of ChatGPT. OpenAI. (n.d.). ChatGPT (Version 3.5) [Online chatbot]. Retrieved from https://www.openai.com