**Nick Domenico**

**Probability and Applied Statistics GitHub Assignment**

Git is a type of version control system (or repository management system), which is a tool to help track changes made to code over time by multiple developers collaborating on a project. Version control systems track every change that is made to code in a repository, that works like a timeline of changes. If a change is made that results in a bug or error, developers can revert back to an earlier version of the project in order to speed up the time it takes to fix the problems. This improves the overall speed of development and allows multiple developers to work on the same project with little risk of messing up code that somebody else wrote. Git, specifically, is the most used version control system by developers today.

Git workflow is the process by which a team will collaborate on the project. Having a consistent workflow helps developers to increase productivity and keeps everyone on the same page for how they should be pushing their code. There are several different types of Git workflows that have been created as outlines, but they are not concrete rules that are set in stone, as you can mix and match different aspects from different workflows to better suit your team. One example of a Git workflow is the centralized workflow, where each developer has a local copy of the central repository (the main branch) that they make and commit changes to at their own pace before pushing those changes to the central repository. Some more examples of Git workflows include feature branching (similar to centralized except each feature of the program has a different branch instead of everything being in main), gitflow workflow (which assigns roles to branches and how they interact), and forking workflow (where each developer has a local repository and a server-side repository).

A commit is a way of saving your code. When thinking about Git as a timeline of different states of your code, a commit is an individual “snapshot” or save of your code at a certain point of time. This commit is then stored on the timeline and can be called back to if needed. Unlike with other version control systems, with Git, you can commit code to both your local repository and to the central repository, which has some advantages; namely being that you can make multiple commits in your local repo and make sure that everything is working properly before committing to the central repo. Another difference is that Git commits are like “snapshots” of your code, while in other version control systems, commits only keep track of the differences between previous versions, not the entire thing. This makes loading commits in Git significantly faster, as you don’t need to rebuild the code from all of the differences.

A push is the command to upload code from a local repository to a remote repository. It is most often used when collaborating on a project to send commits from a local repository to the central repository for the rest of the team to see. Pushing normally does not allow you to overwrite commits from other developers, but you can use the “--force” flag to push the code regardless.

A merge is the command for taking multiple branches of code and combining them into a single branch. When thinking of Git as a timeline, sometimes the changes that a developer makes will diverge from the main timeline, like a branching path. The merge command takes the branching path and merges it back onto the original path.

A merge conflict can happen when multiple developers try to change the same code. If two people edit the same line of code, or if someone deletes a file that another person is working on, and someone tries to execute a merge, Git doesn’t know which developer was correct. When this happens, the merge is stopped, and the developer that called the merge has to fix the issue.

A repository is what allows you to save different versions of your code and call back to earlier version when needed. Repositories are created using the “git init” command, which will create a new “.git” repository. You can also clone other repositories, for example, to make a local copy of a central repository to make changes without directly changing the central repo. In order to save changes to a repository, you need to use the commit command, and the push command is used to upload changes from one repository to another.