Week 6

GUI + Deployment + Summarization

Week 1	Prerequisite Learning
Week 2	Programming Fundamentals (Python) + Required Installation
Week 3	ML Specific
Week 4	Coding
Week 5	Coding + GUI
Week 6	**Deployment + Extension + Summarization**

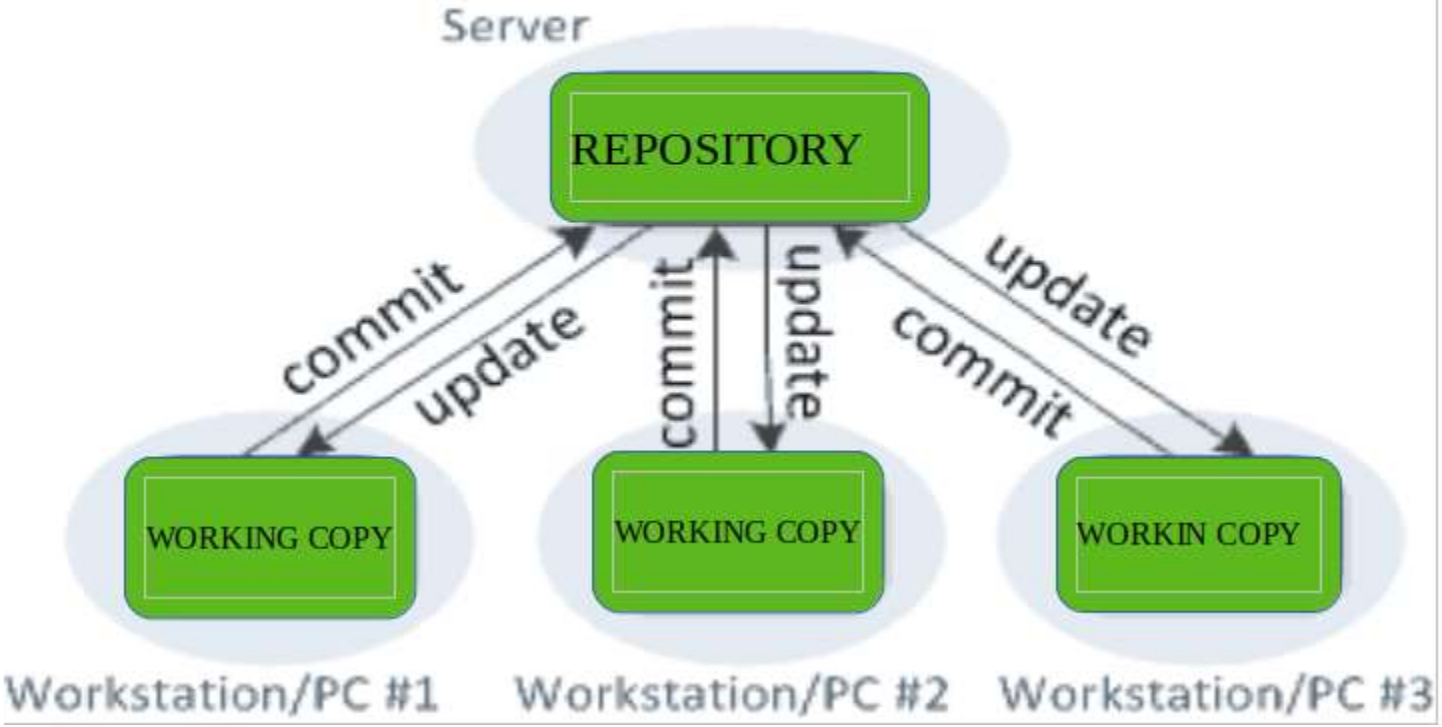
Git and GitHub

TalentBattle

- Git is an open-source, version control tool created in 2005 by developers working on the Linux operating system; GitHub is a company founded in 2008 that makes tools which integrate with git.

- You do not need GitHub to use git, but you cannot use GitHub without using git.

Centralized version control



Step 1: Create a local git repository

To initialize a git repository in the root of the folder, run the git init command:

Step 2: Add a new file to the repo

Add a new file to the project, using any text editor you like or running a **touch** command. `touch newfile.txt` just creates and saves a blank file named newfile.txt.

After creating the new file, you can use the **git status** command to see which files git knows exist.

Step 3: Add a file to the staging environment

Add a file to the staging environment using the git add command.

Step 4: Create a commit

Run the command git commit -m "Your message about the commit"

Step 5: Create a new branch

Say you want to make a new feature but are worried about making changes to the main project while developing the feature. This is where git branches come in.

Branches allow you to move back and forth between 'states' of a project. Official git docs describe branches this way: ‘A branch in Git is simply a lightweight movable pointer to one of these commits.’ For instance, if you want to add a new page to your website you can create a new branch just for that page without affecting the main part of the project. Once you're done with the page, you can merge your changes from your branch into the primary branch. When you create a new branch, Git keeps track of which commit your branch 'branched' off of, so it knows the history behind all the files.

Run git checkout -b <my branch name>.

This command will automatically create a new branch and then 'check you out' on it, meaning git will move you to that branch, off of the primary branch.

After running the above command, you can use the **git branch** command to confirm that your branch was created.

By default, every git repository's first branch is named `master` (and is typically used as the primary branch in the project).

Step 6: Create a new repository on GitHub

If you only want to keep track of your code locally, you don't need to use GitHub. But if you want to work with a team, you can use GitHub to collaboratively modify the project's code.

To create a new repo on GitHub, log in and go to the GitHub home page. You can find the “New repository” option under the “+” sign next to your profile picture, in the top right corner of the navbar.

Step 7: Push a branch to GitHub

Now we'll push the commit in your branch to your new GitHub repo. This allows other people to see the changes you've made. If they're approved by the repository's owner, the changes can then be merged into the primary branch.

To push changes onto a new branch on GitHub, you'll want to run git push origin yourbranchname. GitHub will automatically create the branch for you on the remote repository.

TalentBattle

Step 8: Create a pull request (PR)

A pull request (or PR) is a way to alert a repo's owners that you want to make some changes to their code. It allows them to review the code and make sure it looks good before putting your changes on the primary branch.

Step 9: Merge a PR

This will merge your changes into the primary branch.

Step 10: Get changes on GitHub back to your computer

Right now, the repo on GitHub looks a little different than what you have on your local machine.

For example, the commit you made in your branch and merged into the primary branch doesn't exist in the primary branch on your local machine.

In order to get the most recent changes that you or others have merged on GitHub, use the git pull origin master command (when working on the primary branch). In most cases, this can be shortened to “git pull”.


Thank you!


The best
way to predict
the future
is to create it.