Git Tutorial - Class takeaways

We use git (the versioning tool) and Github (or other similar platforms) in order to easily collaborate with other developers. It is a useful way to track the progress of your project, it allows you to do:

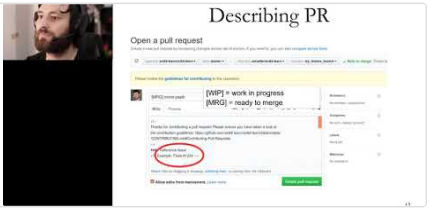
- Track bugs, feature implementations, and discuss project-related topics
- Host closed and open source projects
- Share your work with others
- etc.

You can use this YouTube tutorial taught by one of scikit-learn's core developers as an additional reference:

Scikit-learn sprint instructions

These are instructions for how to participate in a scikit-learn sprint and how to start working on a new open source project!Full transcript for this video i...

https://youtu.be/5OL8XoMMOfA?t=196



Basics of Git

- **Concepts**:
  - Local - The version of the repository that is in your computer
  - Origin - The remote version of the project you created to which you will be making commits directly
  - Upstream - The main remote version of the repository
- Clone a repository: `git clone <repo URL>`
- Add a new file to a repository: `git add <file>`
- Add a commit: `git commit -m "<Commit message here>"`
- Push the commits from local to remote: `git push`
- Get changes in the remote repo: `git pull`

Basic set up

Initially fork the repo. It means you create a copy of the original repo in your own account. Assume that the url of the original repo is `https://github.com/IMS-ML-Lab/whatever.git`. The url of your fork is `https://github.com/your-account/whatever.git`. Clone your fork. It means you create a local copy of your fork.

```
git clone https://github.com/your-account/whatever.git
```

Create two pointers. One for your fork called `origin`. It is automatically created since you cloned from your fork. Another one for the original repo called `upstream`.

```
git remote add upstream https://github.com/IMS-ML-Lab/whatever.git
```

Make sure both of them are present. It should return info about `origin` and `upstream`

```
git remote -v
```

Update your local repo:

```
git pull --rebase upstream master
```

Update your fork:

```
git push -u origin master
```

Contributing

Add a file (optional if you're just modifying existing files):

```
git add <filepath>
```

Commit the changes (if you used git add, passing the file names is not necessary):

```
git commit -m "<Commit message here>" <file changed 1> <file changed 2>
```

Pushing the changes:

Create a Pull Request

List existing branches:

```
git branch
```

Create a branch:

```
git branch your-awesome-branch
```

Switch to that branch:

```
git checkout your-awesome-branch
```

Make your commit and push to your new branch.

```
$ git add .
$ git commit -m 'initial commit'
$ git push origin your-awesome-branch
```

Manage the rest of the PR via GitHub.

Sync a remote fork on Github

1. Open your fork on GitHub.
2. Click on Pull Requests.
3. Click on New Pull Request. By default, GitHub will compare the original with your fork, and there shouldn't be anything to compare if you didn't make any changes.
4. Click on Try `switching the base`. Now GitHub will compare your fork with the original, and you should see all the latest changes.
5. Click on Click to create a pull request for this comparison and assign a predictable name to your pull request (e.g., Update from original).
6. Click on Send pull request.
7. Scroll down and click Merge pull request and finally Confirm merge. If your fork didn't have any changes, you will be able to merge it automatically.