
GIT REPOSITORY

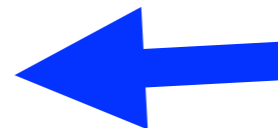
COME LAVORARE CON IL CODICE DI GRUPPO DI ML
10 MAGGIO 2023
S. SPAGNOLO

REPOSITORY

- Repository centrale, in cui far confluire tutti gli sviluppi e in cui salvare versioni specifiche del codice
- <https://github.com/LecceGroup/DNNkit>

GITHUB MANUALE

- <https://docs.github.com/en>
 - Or
 - <https://www.atlassian.com/git/tutorials>
 - <https://git-scm.com/>
- **Come collaborare allo sviluppo di codice di gruppo con GitHub ?**
 - <https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/getting-started/about-collaborative-development-models>
 - Esistono due modi:
 - **Fork and pull model**
 - Shared repository model



Scegliamo questo

COME LAVORARE

Fork and pull model

- 1) ottenere account GitHub (<https://github.com/> -> Sign up) e accesso via ssh
 - <https://docs.github.com/en/authentication/connecting-to-github-with-ssh>
- 2) creare una fork della repository di gruppo <https://github.com/stefaniaspagnolo/DNNkit>
- 3) scaricare il codice in locale dalla propria fork e creare il link alla repository remota di gruppo
- 4) creare una propria branch locale e remota
- 5) sviluppare codice nella propria branch
 - Sequenza di editing, test e commit (cioe' salvataggi delle nuove versioni del codice in una repository locale) - la branch remota rimane stabile
- 6) aggiornare il codice nella branch remota con gli sviluppi aggiornati nella repository locale - ogni volta che c'e' un progresso significativo o che si vuole discutere (e quindi far vedere) il proprio codice a colleghi
- 7) quando la versione del codice nella branch remota ha raggiunto uno sviluppo significativo e desiderato, richiedere un merge della propria branch remota con la branch principale della repository di gruppo, *dopo aver fatto un merge con la branch main della repository di gruppo*
- 8) quando il manager della repository di gruppo accetta il merge, localmente spostarsi sulla main, aggiornarla con la versione attuale della branch principale della repository di gruppo, rimuovere la propria branch locale e la sua versione remota
- 9) per il prossimo sviluppo, ripartire da 4)

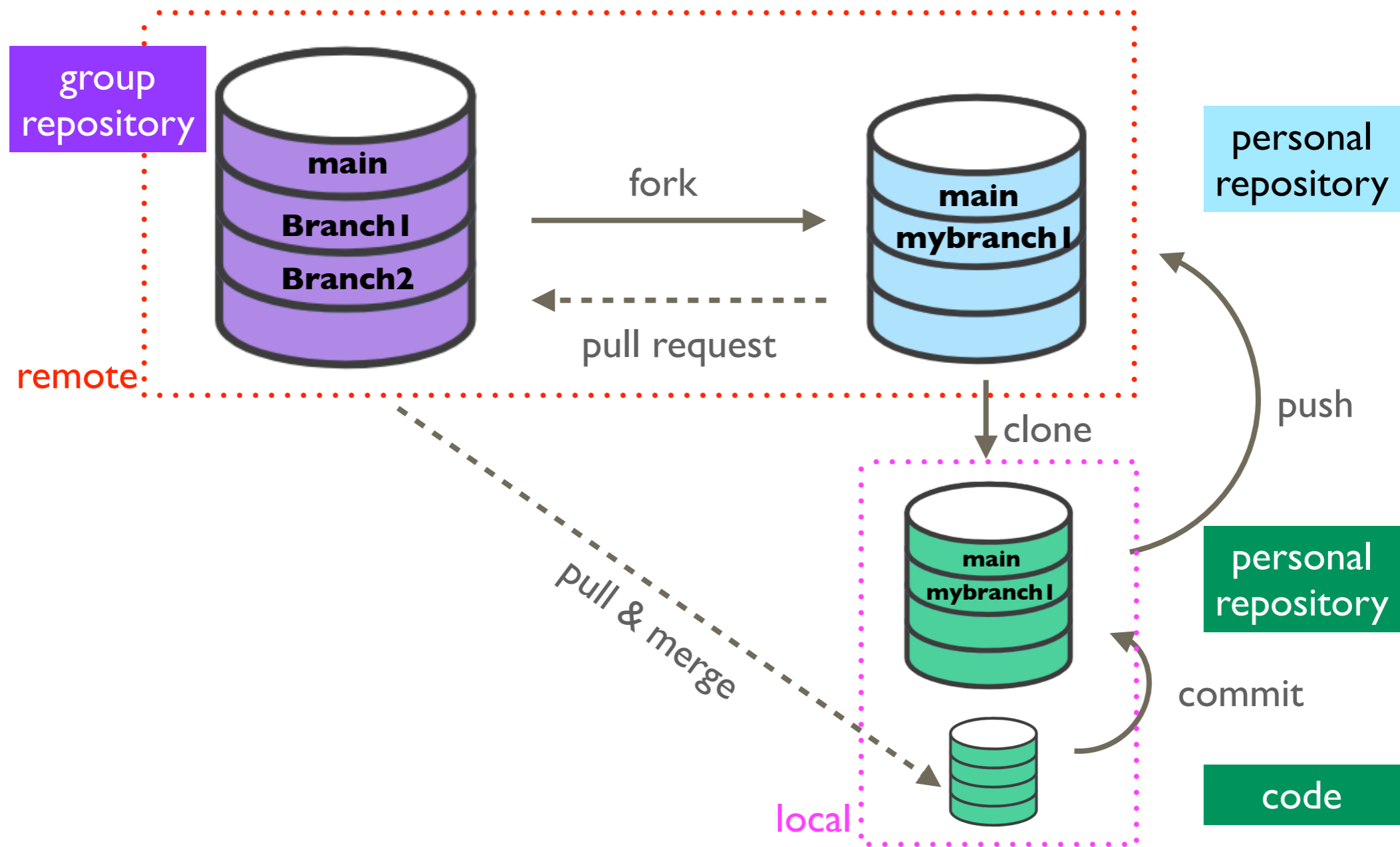
COME LAVORARE

<https://www.atlassian.com/git/tutorials/comparing-workflows/forking-workflow>

1. A developer 'forks' an 'official' server-side repository. This creates their own server-side copy.
2. The new server-side copy is cloned to their local system.
3. A Git remote path for the 'official' repository is added to the local clone.
4. A new local feature branch is created.
5. The developer makes changes on the new branch.
6. New commits are created for the changes.
7. The branch gets pushed to the developer's own server-side copy.
8. The developer opens a pull request from the new branch to the 'official' repository.
9. The pull request gets approved for merge and is merged into the original server-side repository

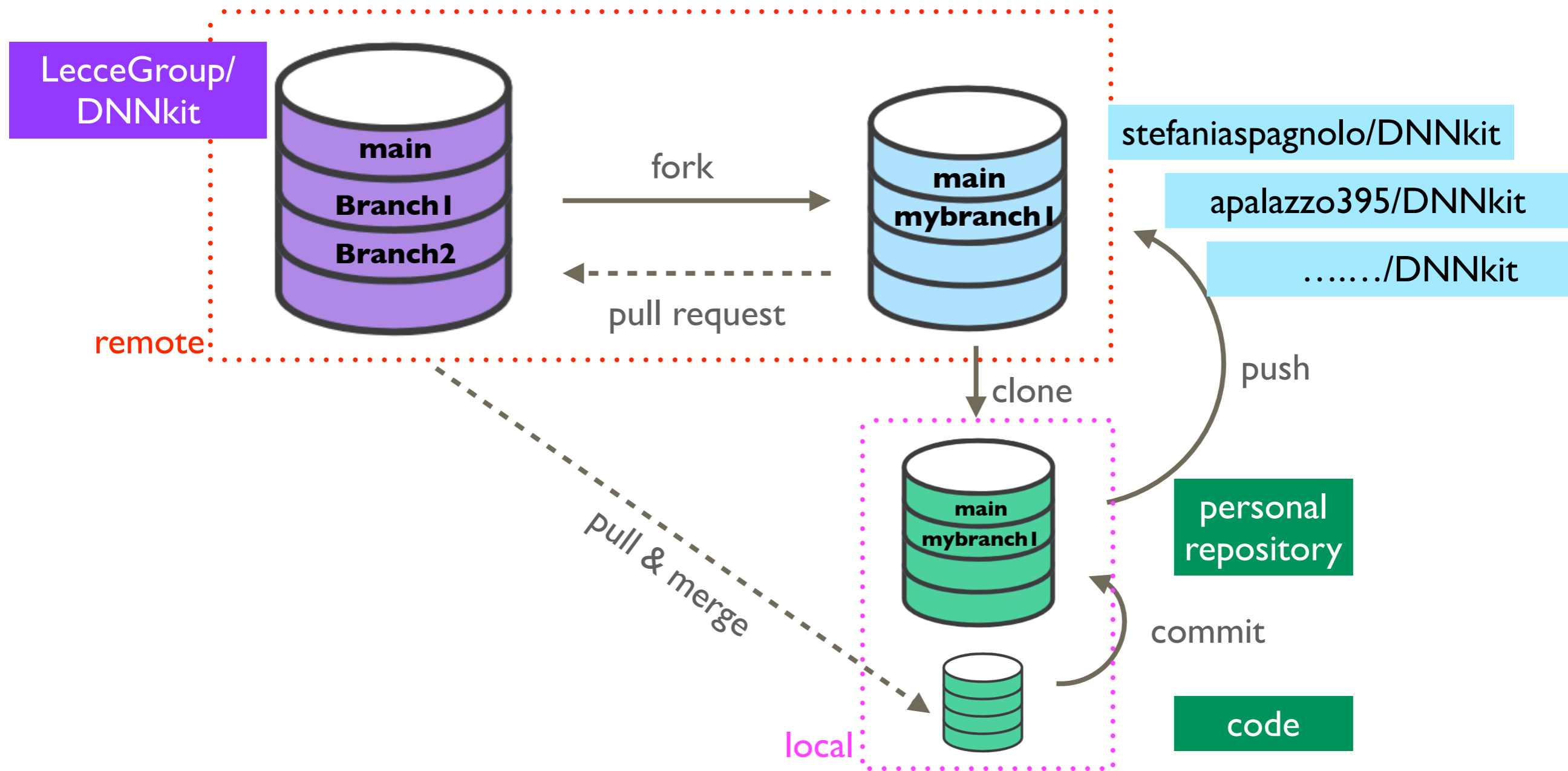
COME LAVORARE

<https://www.atlassian.com/git/tutorials/comparing-workflows/forking-workflow>



COME LAVORARE - NEL NOSTRO CASO

<https://www.atlassian.com/git/tutorials/comparing-workflows/forking-workflow>



COME LAVORARE

Fork and pull model

- 1) ottenere account GitHub (<https://github.com/> -> Sign up) e accesso via ssh *web+linux shell*
 - <https://docs.github.com/en/authentication/connecting-to-github-with-ssh>
- 2) creare una fork della repository di gruppo <https://github.com/stefaniaspagnolo/LecceDNN> *web*
- 3) scaricare il codice in locale dalla propria fork e creare il link alla repository remota di gruppo
- 4) creare una propria branch locale e remota *linux shell* *From web to shell*
- 5) sviluppare codice nella propria branch *linux shell*
 - Sequenza di editing, test e commit (cioè salvataggi delle nuove versioni del codice in una repository locale) - la branch remota rimane stabile
- 6) aggiornare il codice nella branch remota con gli sviluppi aggiornati nella repository locale - ogni volta che c'è un progresso significativo o che si vuole discutere (e quindi far vedere) il proprio codice a colleghi *From shell to web*
- 7) quando la versione del codice nella branch remota ha raggiunto uno sviluppo significativo e desiderato, richiedere un merge della propria branch remota con la branch principale della repository di gruppo, *dopo aver fatto un merge con la branch main della repository di gruppo* *From shell to web*
- 8) quando il manager della repository di gruppo accetta il merge, localmente spostarsi sulla main, aggiornarla con la versione attuale della branch principale della repository di gruppo, rimuovere la propria branch locale e la sua versione remota *web*
- 9) per il prossimo sviluppo, ripartire da 4)

COME LAVORARE

Fork and pull model

- 3) scaricare il codice in locale dalla propria fork e creare il link alla repository remota di gruppo
 - `git clone git@github.com:utentegithub/DNNkit.git`
 - `git remote -v` ora mostra
 - origin git@github.com:utentegithub/DNNkit.git (fetch)
 - origin git@github.com:utentegithub/DNNkit.git (push)
 - Commento: La mia repository remota (fork di quella di gruppo) risulta repository remota di origine (origin) della mia copia del codice con la sua repository locale (definita dal contenuto della directory locale `.git`). Da questa repository posso prendere codice (fetch), prendere e aggiornare codice locale (pull) e su questa repository remota posso scrivere codice (push)
 - `git remote add upstream git@github.com:LecceGroup/DNNkit.git`
 - `git remote -v` ora mostra
 - origin git@github.com:utentegithub/DNNkit.git (fetch)
 - origin git@github.com:utentegithub/DNNkit.git (push)
 - upstream git@github.com:LecceGroup/DNNkit.git (fetch)
 - ppstream git@github.com:LecceGroup/DNNkit.git (push)
 - Commento: adesso due repository remote sono collegate alla mia repository locale, origin (la mia fork della repository di gruppo) e upstream (la repository di gruppo)

COME LAVORARE

Fork and pull model

- 4) creare una propria branch (chiamata dev/stefania) **locale** e **remota**
 - **git checkout -b dev/stefania**
 - oppure
 - **git branch dev/stefania**
 - **git checkout dev/stefania**
 - In entrambi i casi *git branch* mi mostra tutte le branch disponibili localmente e mi indica (con un asterisco) quella in uso, posso muovermi da una branch a un'altra (chiamata otherBranch) con *git checkout otherBranch*
 - **git push origin dev/stefania** oppure **git push --set-upstream origin dev/stefania**
 - Copia nella repository remota personale (origin) la branch locale dev/stefania
- 5) sviluppare codice nella propria branch
 - **git status** mostra la branch in uso, I files modificati localmente (rispetto alla repository locale)
 - **git add myFile** prenota la copia del file locale (nuovo o modificato) nella repository locale
 - **git commit -m "minimo commento" myFile** copia il file locale (nuovo o modificato) nella repository locale

SU WEB

<https://github.com/stefaniaspagnolo/DNNkit>

stefaniaspagnolo / DNNkit Public

forked from LecceGroup/DNNkit

Pull requests Actions Projects Wiki Security Insights Settings

main 2 branches 0 tags

Go to file Add file Code

Switch branches/tags

Find or create a branch...

Branches Tags

main default

dev/stefania

View all branches

testScripts

updates to scripts last week

About

pDNN code from Dinos to Enrico/Martino/Alessandra

Readme

0 stars

0 watching

1 fork

Releases

No releases published

COME LAVORARE

Fork and pull model

- 6) **git push origin dev/stefania**
 - aggiorna il codice nella branch remota (dev/stefania) con gli sviluppi aggiornati nella repository locale
- 7)
 - **git pull upstream main** *Vedi alternativa*
 - Aggiorna la branch locale main con il contenuto della branch remota upstream/main (ossia della branch main della repository di gruppo)
 - **git merge main**
 - Fa un merge della branch attuale (dev/stefania) con la branch main
 - E' possibile che sia necessario risolvere conflitti
 - **git commit -m "..."**
 - **git push**
 - Da web generare una pull request
 - richiede un merge della propria branch remota con la branch principale della repository di gruppo

COME LAVORARE

Fork and pull model

- 6) **git push origin dev/stefania**
 - aggiorna il codice nella branch remota (dev/stefania) con gli sviluppi aggiornati nella repository locale
- 7)
 - **git fetch upstream main; git checkout main; git merge upstream/main; git checkout dev/stefania**
 - Scarica la branch main di upstream; mi sposto su main locale; non faccio un merge con upstream/main; torno nella branch di sviluppo locale dev/stefania
 - **git merge main**
 - Fa un merge della branch attuale (dev/stefania) con la branch main
 - E' possibile che sia necessario risolvere conflitti
 - **git commit -m "..."**
 - **git push**
 - Da web generare una pull request
 - richiede un merge della propria branch remota con la branch principale della repository di gruppo

alternativa

COME LAVORARE

Fork and pull model

- quando il manager della repository di gruppo accetta il merge,
- 8)
 - `git checkout main`
 - `git pull upstream`
 - Spostrarsi localmente sulla branch main e aggiornarla alla versione sulla repository remota di gruppo
 - Da interfaccia web cancellare la branch di sviluppo dev/stefania sulla propria fork
 - `git branch -d dev/stefania`
 - Rimuove la versione locale della branch
 - Da interfaccia web sync della fork

DOVE SI TROVA IL CODICE

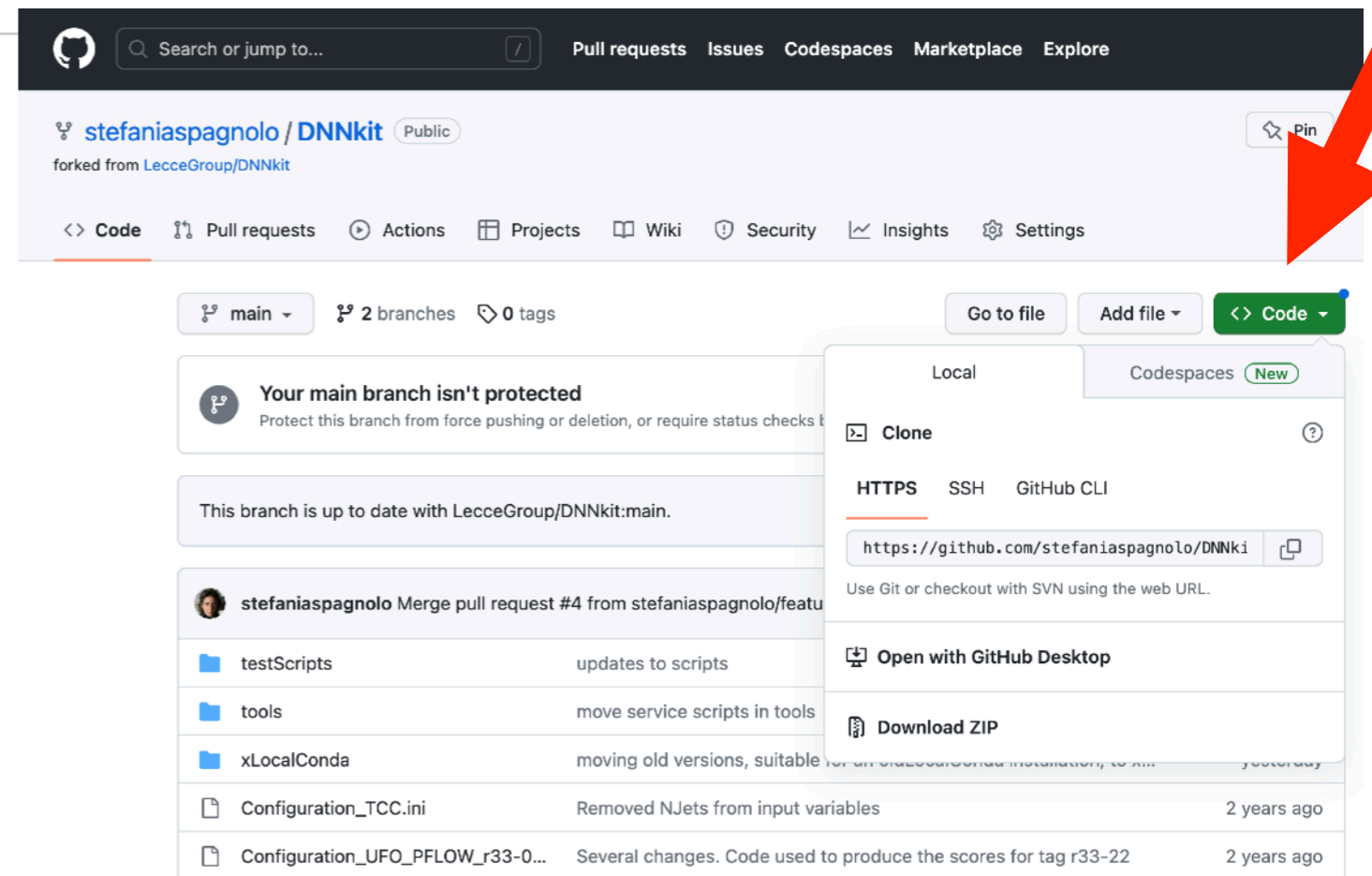
IL MODO PIU' COMODO DI CONSULTARE IL CODICE

The screenshot shows the GitHub repository page for `LecceGroup/DNNkit`. The repository is public and has 172 commits. The main content area displays a list of files and folders, including `testScripts`, `tools`, `xLocalConda`, and various `Configuration` files. The sidebar on the right provides information about the repository, including the number of stars (0), watchers (1), and forks (1). A blue arrow points to the 'Fork' button in the top right corner of the repository page.

File/Folder	Description	Last Commit
testScripts	updates to scripts	last week
tools	move service scripts in tools	yesterday
xLocalConda	moving old versions, suitable for an oldLocalConda installation, to x...	yesterday
Configuration_TCC.ini	Removed NJets from input variables	2 years ago
Configuration_UFO_PFLOW_r33-0...	Several changes. Code used to produce the scores for tag r33-22	2 years ago
Configuration_UFO_PFLOW_r33-2...	Fixed typo in Configuration file	last year
Configuration_UFO_PFLOW_r33-2...	Working version for all channels	5 months ago
Configuration_r33-24.ini	new versions of pytjon scripts running in the lcg environment over cv...	last week
DSIDtoMass.txt	Added two DSID values that were missing for RSG	2 years ago
Functions.py	new versions of pytjon scripts running in the lcg environment over cv...	last week
README.md	moving old versions, suitable for an oldLocalConda installation, to x...	yesterday
buildDNN.py	Fixed bug in weighting function	2 years ago
buildDataset.py	new versions of pytjon scripts running in the lcg environment over cv...	last week
buildPDNN.py	minot updates gor the scripts to launch ML procedures	yesterday
computeSignificance.py	Working version for all channels	5 months ago
saveToPkl.py	new versions of pytjon scripts running in the lcg environment over cv...	last week
splitDataset.py	adding necessary import to splitDataset.py	last week

- <https://github.com/LecceGroup/DNNkit>
- Ore c'e' una sola fork, la mia
- **Da qui, produci la tua fork**

COME SCARICARE IL CODICE

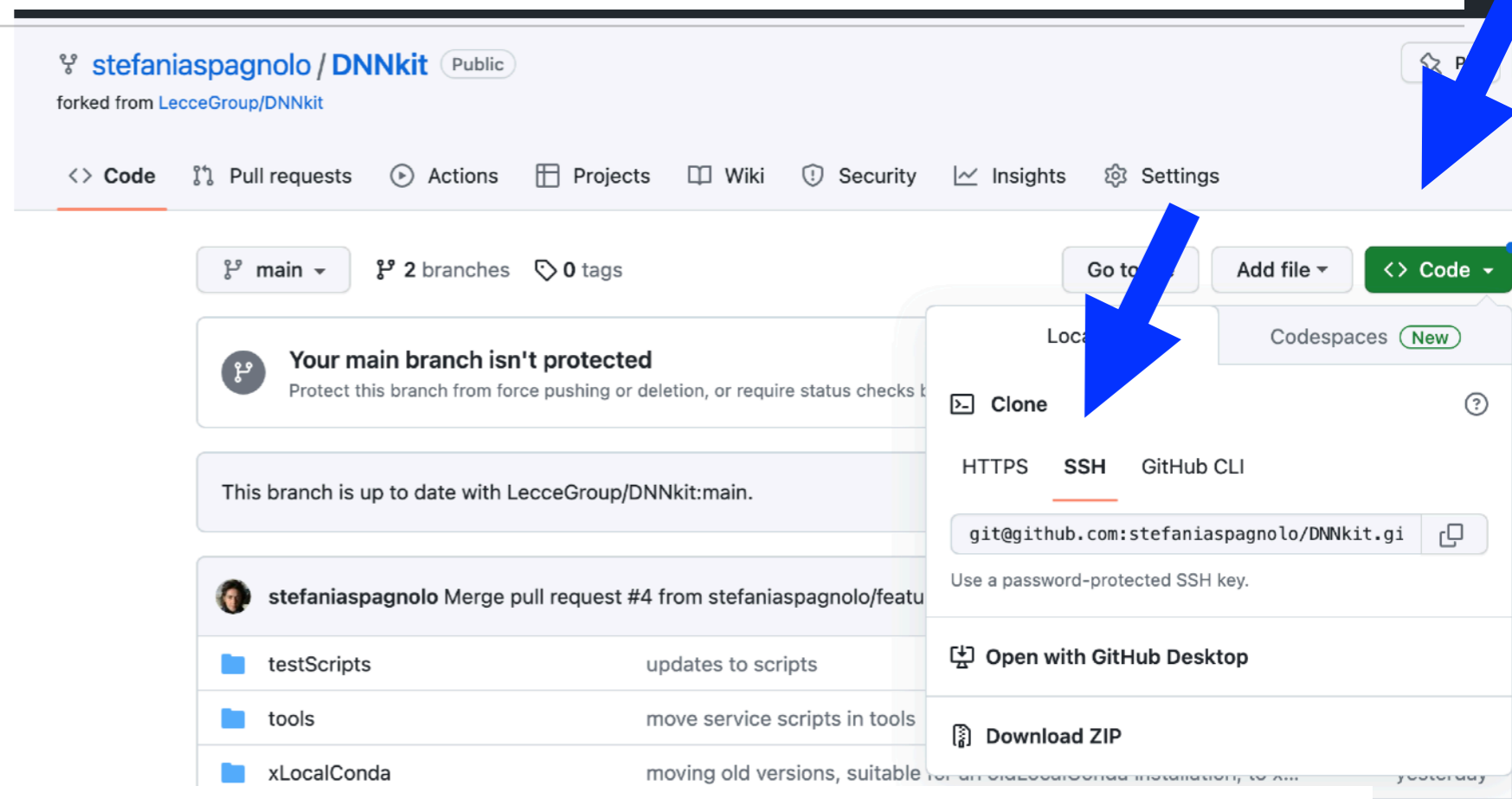


The screenshot shows the GitHub interface for the repository 'stefaniaspagnolo / DNNkit'. The 'Code' button is highlighted with a red arrow, and a dropdown menu is open showing options like 'Clone', 'Open with GitHub Desktop', and 'Download ZIP'. The repository is forked from 'LecceGroup/DNNkit'. The main branch is 'main' with 2 branches and 0 tags. A warning message states 'Your main branch isn't protected'. The repository is up to date with 'LecceGroup/DNNkit:main'. A pull request #4 is visible. The file list includes 'testScripts', 'tools', 'xLocalConda', 'Configuration_TCC.ini', and 'Configuration_UFO_PFLOW_r33-0...'. The file list shows the following details:

File Name	Description	Last Commit
testScripts	updates to scripts	
tools	move service scripts in tools	
xLocalConda	moving old versions, suitable for an on-premise installation, to x...	
Configuration_TCC.ini	Removed NJets from input variables	2 years ago
Configuration_UFO_PFLOW_r33-0...	Several changes. Code used to produce the scores for tag r33-22	2 years ago

- Da terminale
 - `git clone https://github.com/stefaniaspagnolo/DNNkit.git`
 - usando `https:` non richiede password / setup di sistemi di autenticazione, ma **NON** permette di produrre una propria versione su repository remota
 - **NON** e' una buona soluzione per collaborare nello sviluppo di codice

COME SCARICARE IL CODICE



- Da terminale
 - `git clone git@github.com:stefaniaspagnolo/DNNkit.git`
 - usando ssh: richiede password (e un po' di setup), ma permette di apportare le proprie modifiche a una propria versione su repository remota
 - **RACCOMANDATO** per collaborare nello sviluppo di codice