

holoclean is a algorithm for data cleaning and in this “How to” we will review the packages and methods that implement it in Python.

0- Installation	2
0.1 Download and using Anaconda	2
0.2 Downloading Git Repo	4
0.3 Create a New MySQL User and Database	5
0.4 Installing Required Packages	8
0.5 Installing Pytorch	8
1- Session	9
2.1 Download drivers for MySQL	
2- Preliminaries	9
1.1 Hello Git!	9
1.2 Branching	13
1.3 Merge codes from a direction	15
1.4 Push your code to the repo	19
1.5 Push your code to the repo	23
1.6 clone a specific branch	23
1.7 Set SSH on Github	30
1.1 Generating a new SSH key and adding it to the ssh-agent	31
1.2 Adding your SSH key to the ssh-agent	32
1.3 Adding a new SSH key to your GitHub account	34
1.4 How use SSH with github	37
3- Project	39
4- Reading data	39
5- Parsing	39
6- Learning	39
7- Interaction and feedback	39

0- Installation

0.1 Download and using Anaconda

For Ubuntu:

- For the 32 bits version:
Open a terminal and write the following commands
 1. **wget**
`https://3230d63b5fc54e62148e-c95ac804525aac4b6dba79b00b39d1d3.ssl.cf1.rackcdn.com/Anaconda-2.3.0-Linux-x86.sh`
 2. **`bash Anaconda-2.3.0-Linux-x86.sh`**
- For the 64 bits:
Open a terminal and write the following commands
 1. **wget**
`https://3230d63b5fc54e62148e-c95ac804525aac4b6dba79b00b39d1d3.ssl.cf1.rackcdn.com/Anaconda-2.3.0-Linux-x86_64.sh`
 2. **`bash Anaconda-2.3.0-Linux-x86_64.sh`**

For MacOS:

- Follow the instructions on <https://conda.io/docs/user-guide/install/macos.html> to install Anaconda for MacOS

Installing on macOS

1. Download the installer:

- [Miniconda installer for macOS.](#)
- [Anaconda installer for macOS.](#)

2. Install:

- Miniconda—In your Terminal window, run:

```
bash Miniconda3-latest-MacOSX-x86_64.sh
```

- Anaconda—Double-click the `.pkg` file.

3. Follow the prompts on the installer screens.

If you are unsure about any setting, accept the defaults. You can change them later.

4. To make the changes take effect, close and then re-open your Terminal window.

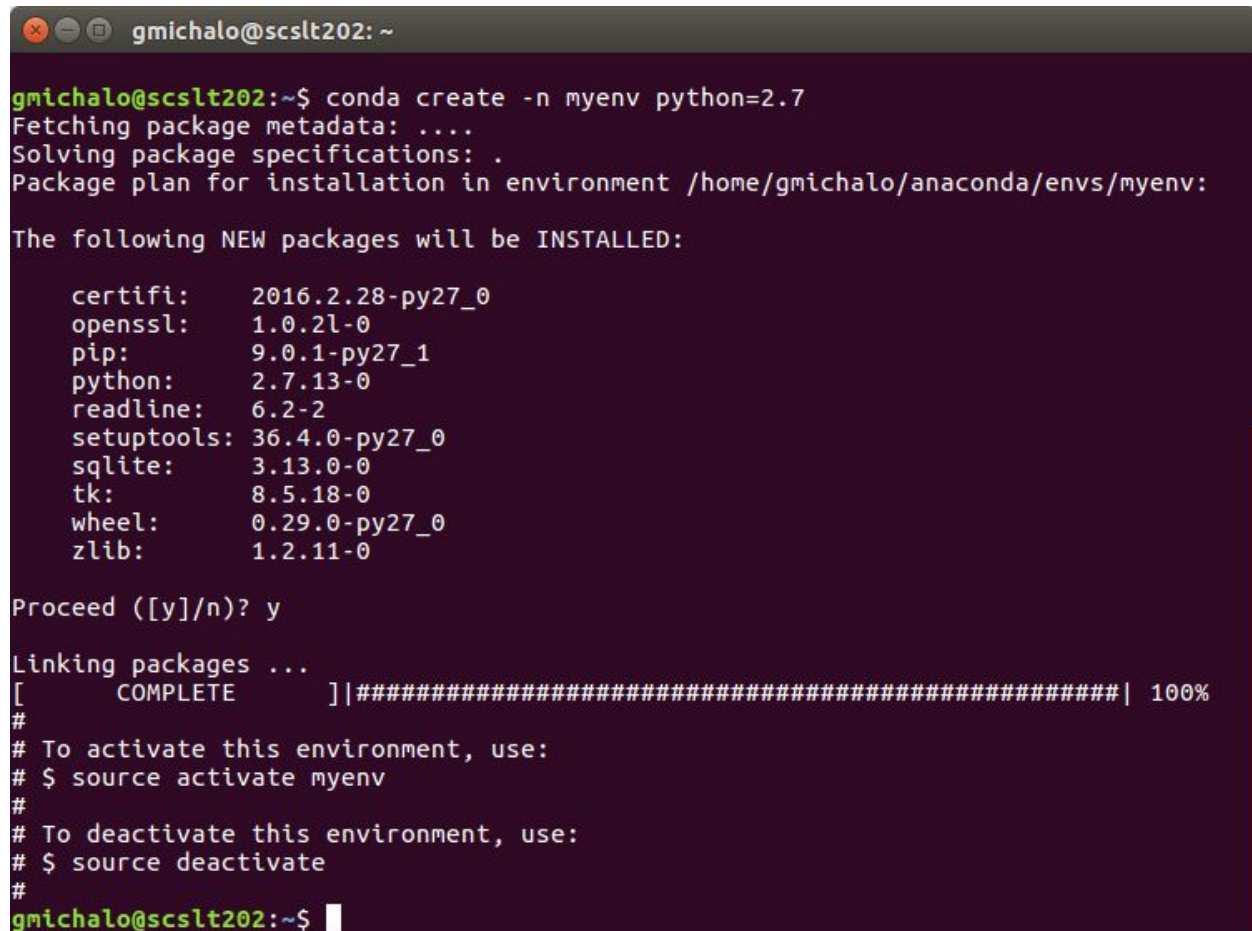
5. [Test your installation.](#)

Now you have Anaconda you can create a new environment using Python 2.7 where you run the all the other programs

We can create a new environment by writing in a terminal

conda create -n myenv python=2.7

Where myenv is the name of your environment. If everything is fine you should see this message on your screen.

A terminal window with a dark purple background and light green text. The window title is 'gmichalo@scs1t202: ~'. The user has entered the command 'conda create -n myenv python=2.7'. The terminal output shows the process of fetching metadata, solving specifications, and displaying a list of packages to be installed: certifi, openssl, pip, python, readline, setuptools, sqlite, tk, wheel, and zlib. It then asks for confirmation to proceed, which is answered with 'y'. Finally, it shows the linking of packages at 100% completion and provides instructions on how to activate and deactivate the environment.

```
gmichalo@scs1t202: ~$ conda create -n myenv python=2.7
Fetching package metadata: ....
Solving package specifications: .
Package plan for installation in environment /home/gmichalo/anaconda/envs/myenv:

The following NEW packages will be INSTALLED:

certifi:      2016.2.28-py27_0
openssl:      1.0.2l-0
pip:          9.0.1-py27_1
python:       2.7.13-0
readline:     6.2-2
setuptools:   36.4.0-py27_0
sqlite:       3.13.0-0
tk:           8.5.18-0
wheel:        0.29.0-py27_0
zlib:         1.2.11-0

Proceed ([y]/n)? y

Linking packages ...
[      COMPLETE      ]|#####| 100%
#
# To activate this environment, use:
# $ source activate myenv
#
# To deactivate this environment, use:
# $ source deactivate
#
gmichalo@scs1t202: ~$
```

In order to activate this environment you should use the command:

Source activate myenv

And if you want to deactivate this environment you should use:

Source deactivate

0.2 Download the git repo

Clone the git repo

```
git clone https://github.com/HoloClean/HoloClean-v0.01
```

```
james@james-All-Series: ~/Desktop/Holoclean
(myenv)james@james-All-Series:~/Desktop/Holoclean$ git clone https://github.com/HoloClean/HoloClean-v0.01
Cloning into 'HoloClean-v0.01'...
Username for 'https://github.com': j48zheng
Password for 'https://j48zheng@github.com':
remote: Counting objects: 3105, done.
remote: Compressing objects: 100% (310/310), done.
remote: Total 3105 (delta 248), reused 352 (delta 148), pack-reused 2639
Receiving objects: 100% (3105/3105), 55.21 MiB | 532.00 KiB/s, done.
Resolving deltas: 100% (1613/1613), done.
Checking connectivity... done.
(myenv)james@james-All-Series:~/Desktop/Holoclean$
```

Check out the pytorch branch

```
git checkout pytorch
```

```
james@james-All-Series: ~/Desktop/Holoclean/HoloClean-v0.01
(myenv)james@james-All-Series:~/Desktop/Holoclean$ git clone https://github.com/HoloClean/HoloClean-v0.01
Cloning into 'HoloClean-v0.01'...
Username for 'https://github.com': j48zheng
Password for 'https://j48zheng@github.com':
remote: Counting objects: 3105, done.
remote: Compressing objects: 100% (310/310), done.
remote: Total 3105 (delta 248), reused 352 (delta 148), pack-reused 2639
Receiving objects: 100% (3105/3105), 55.21 MiB | 532.00 KiB/s, done.
Resolving deltas: 100% (1613/1613), done.
Checking connectivity... done.
(myenv)james@james-All-Series:~/Desktop/Holoclean$ ls
Anaconda-2.3.0-Linux-x86_64.sh  HoloClean-v0.01  spark-2.2.1-bin-hadoop2.7.tgz
derby.log                    metastore_db
Holoclean How to.docx        spark-2.2.1-bin-hadoop2.7
(myenv)james@james-All-Series:~/Desktop/Holoclean$ cd HoloClean-v0.01/
(myenv)james@james-All-Series:~/Desktop/Holoclean/HoloClean-v0.01$ ls
README.md
(myenv)james@james-All-Series:~/Desktop/Holoclean/HoloClean-v0.01$ git checkout holospark
Branch holospark set up to track remote branch holospark from origin.
Switched to a new branch 'holospark'
(myenv)james@james-All-Series:~/Desktop/Holoclean/HoloClean-v0.01$ ls
book-data  Holoclean_demo.ipynb  python-package-requirement.txt  script.py  Tutorial
docs       holofusion            README.md                        set_env.sh
holoclean  mysql_script.sh       run.sh                          test
(myenv)james@james-All-Series:~/Desktop/Holoclean/HoloClean-v0.01$
```

0.3 Setup MySql Server

For Ubuntu:

First update and upgrade your apt-get

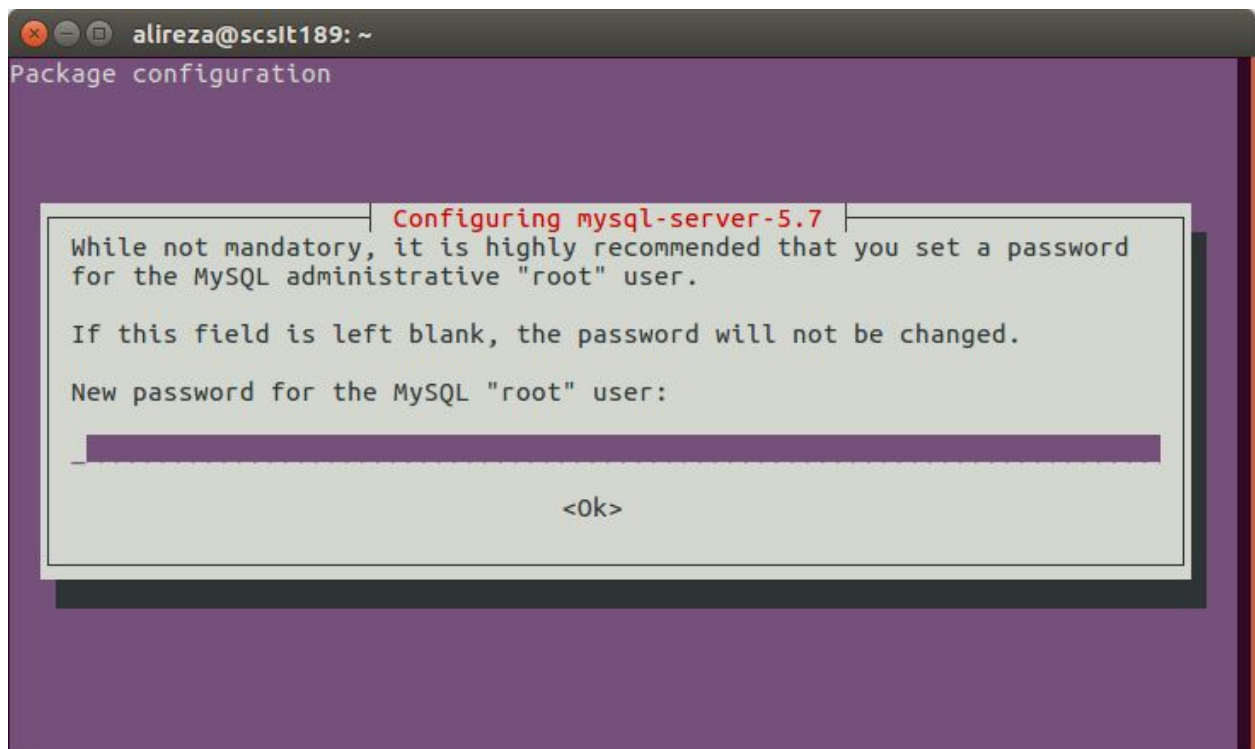
```
sudo apt-get update
```

```
sudo apt-get upgrade
```

Then you can Install MySQL

```
sudo apt-get install mysql-server
```

The you should set password for your server



After that you repeat the password again and the installation will continue


```

alireza@scsIt189: ~
Preparing to unpack .../mysql-server-5.7_5.7.19-0ubuntu0.16.04.1_amd64.deb ...
Unpacking mysql-server-5.7 (5.7.19-0ubuntu0.16.04.1) ...
Selecting previously unselected package libhtml-template-perl.
Preparing to unpack .../libhtml-template-perl_2.95-2_all.deb ...
Unpacking libhtml-template-perl (2.95-2) ...
Selecting previously unselected package mysql-server.
Preparing to unpack .../mysql-server_5.7.19-0ubuntu0.16.04.1_all.deb ...
Unpacking mysql-server (5.7.19-0ubuntu0.16.04.1) ...
Processing triggers for man-db (2.7.5-1) ...
Processing triggers for libc-bin (2.23-0ubuntu9) ...
Processing triggers for ureadahead (0.100.0-19) ...
Processing triggers for systemd (229-4ubuntu19) ...
Setting up mysql-server-core-5.7 (5.7.19-0ubuntu0.16.04.1) ...
Setting up libevent-core-2.0-5:amd64 (2.0.21-stable-2ubuntu0.16.04.1) ...
Setting up mysql-server-5.7 (5.7.19-0ubuntu0.16.04.1) ...
update-alternatives: using /etc/mysql/mysql.cnf to provide /etc/mysql/my.cnf (my.cnf) in auto mode
Renaming removed key_buffer and myisam-recover options (if present)
Setting up libhtml-template-perl (2.95-2) ...
Setting up mysql-server (5.7.19-0ubuntu0.16.04.1) ...
Processing triggers for libc-bin (2.23-0ubuntu9) ...
Processing triggers for systemd (229-4ubuntu19) ...
Processing triggers for ureadahead (0.100.0-19) ...
alireza@scsIt189:~$

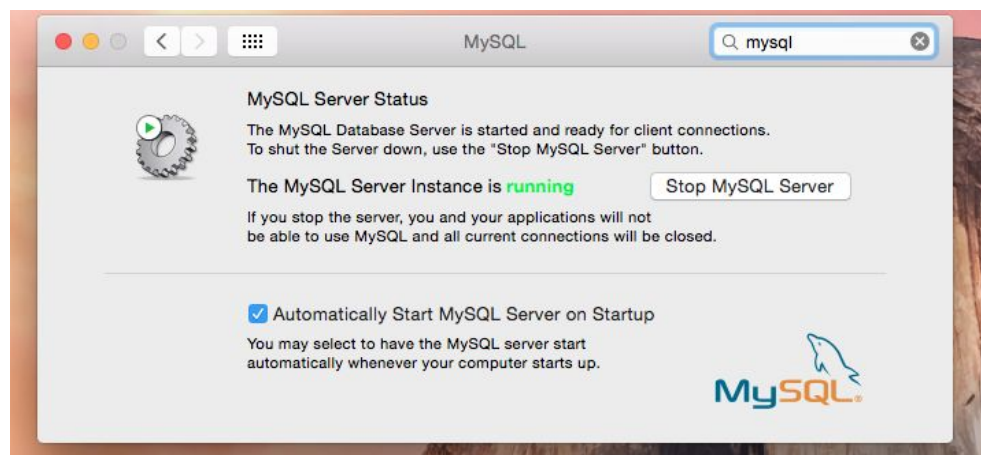
```

After this part we should install the service for client (python as client)

```
sudo apt-get install python-dev libmysqlclient-dev
```

For MacOS:

Install and run the .dmg file from <https://dev.mysql.com/downloads/mysql/>
 After the installation is finished, open system preferences and click on the MySQL icon and make sure the MySQL Server Instance is running

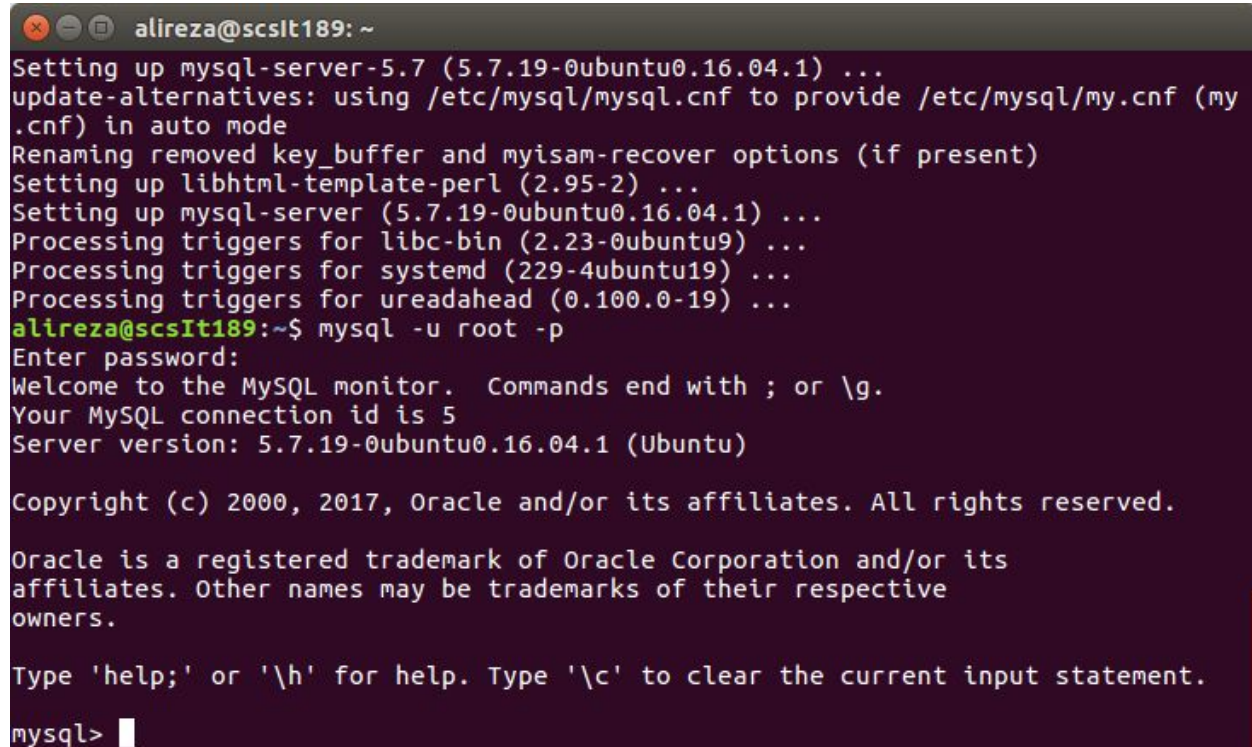


Then open terminal and run:

```
sudo usr/local/mysql/bin/mysql_secure_installation
```

The standard tool for interacting with MySQL is the mysql client, which installs with the mysql-server package.

```
mysql -u root -p
```

A terminal window titled 'alireza@scsIt189: ~' showing the installation of MySQL server. The output includes: 'Setting up mysql-server-5.7 (5.7.19-0ubuntu0.16.04.1) ...', 'update-alternatives: using /etc/mysql/mysql.cnf to provide /etc/mysql/my.cnf (my.cnf) in auto mode', 'Renaming removed key_buffer and myisam-recover options (if present)', 'Setting up libhtml-template-perl (2.95-2) ...', 'Setting up mysql-server (5.7.19-0ubuntu0.16.04.1) ...', 'Processing triggers for libc-bin (2.23-0ubuntu9) ...', 'Processing triggers for systemd (229-4ubuntu19) ...', 'Processing triggers for ureadahead (0.100.0-19) ...'. The user then runs 'mysql -u root -p', enters a password, and is greeted with 'Welcome to the MySQL monitor. Commands end with ; or \g. Your MySQL connection id is 5. Server version: 5.7.19-0ubuntu0.16.04.1 (Ubuntu)'. Copyright and Oracle trademark notices are displayed. The prompt 'mysql>' is shown at the bottom.

```
alireza@scsIt189: ~
Setting up mysql-server-5.7 (5.7.19-0ubuntu0.16.04.1) ...
update-alternatives: using /etc/mysql/mysql.cnf to provide /etc/mysql/my.cnf (my
.cnf) in auto mode
Renaming removed key_buffer and myisam-recover options (if present)
Setting up libhtml-template-perl (2.95-2) ...
Setting up mysql-server (5.7.19-0ubuntu0.16.04.1) ...
Processing triggers for libc-bin (2.23-0ubuntu9) ...
Processing triggers for systemd (229-4ubuntu19) ...
Processing triggers for ureadahead (0.100.0-19) ...
alireza@scsIt189:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 5
Server version: 5.7.19-0ubuntu0.16.04.1 (Ubuntu)

Copyright (c) 2000, 2017, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

Create a New MySQL User and Database

run script create the MySQL server

```
./mysql_script.sh
```

```
james@james-All-Series: ~/Desktop/Holoclean
Server version: 5.7.21-0ubuntu0.16.04.1 (Ubuntu)

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> create database holo;
Query OK, 1 row affected (0.00 sec)

mysql> create user 'holocleanUser'@'localhost' identified by 'abcd1234'
-> ;
Query OK, 0 rows affected (0.00 sec)

mysql> grant all on holo.* to 'holocleanUser'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql> select host, user from mysql.user;
+-----+-----+
| host      | user          |
+-----+-----+
| localhost | debian-sys-maint |
| localhost | holocleanUser  |
| localhost | mysql.session  |
| localhost | mysql.sys      |
| localhost | root           |
+-----+-----+
5 rows in set (0.00 sec)

mysql> 
```

Now you can access to the database by your user setting

0.4- Installing required packages

Install required packages by running:

```
pip install python-package-requirement.txt
```

0.5- Install pytorch

Follow instructions at:

<http://pytorch.org/>

To install pytorch for your OS

Make sure to install version 0.3.0 or later

1- Session

Holoclean use Spark session to be executed over large data

1.1 Download drivers for MySQL

In order to use spark and Mysql, we need to download the drivers from:

<https://dev.mysql.com/downloads/connector/j/5.1.html>

Afterwards we untar the file and when we start a spark session in holoclean we put as an argument the whole path to the jar file. For example:

```
holoclean_se._start_spark_session("/home/user/Downloads/mysql-connector-java-5.1.44/mysql-connector-java-5.1.44-bin.jar")
```

2- Preliminaries

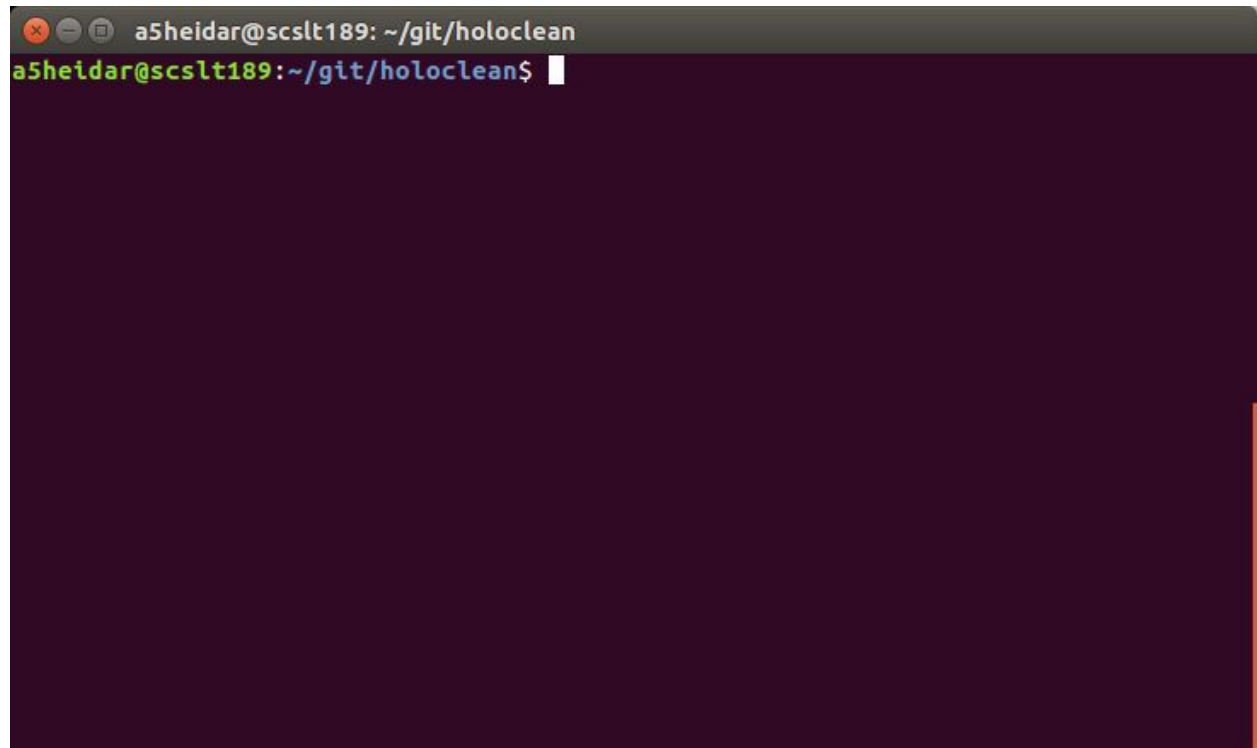
2.1 Hello Git!

To be part of team you should know how to communicate with them as you might know programmer language is code and their social network is Github so in this section we would learn how to connect to git.

Requirement :

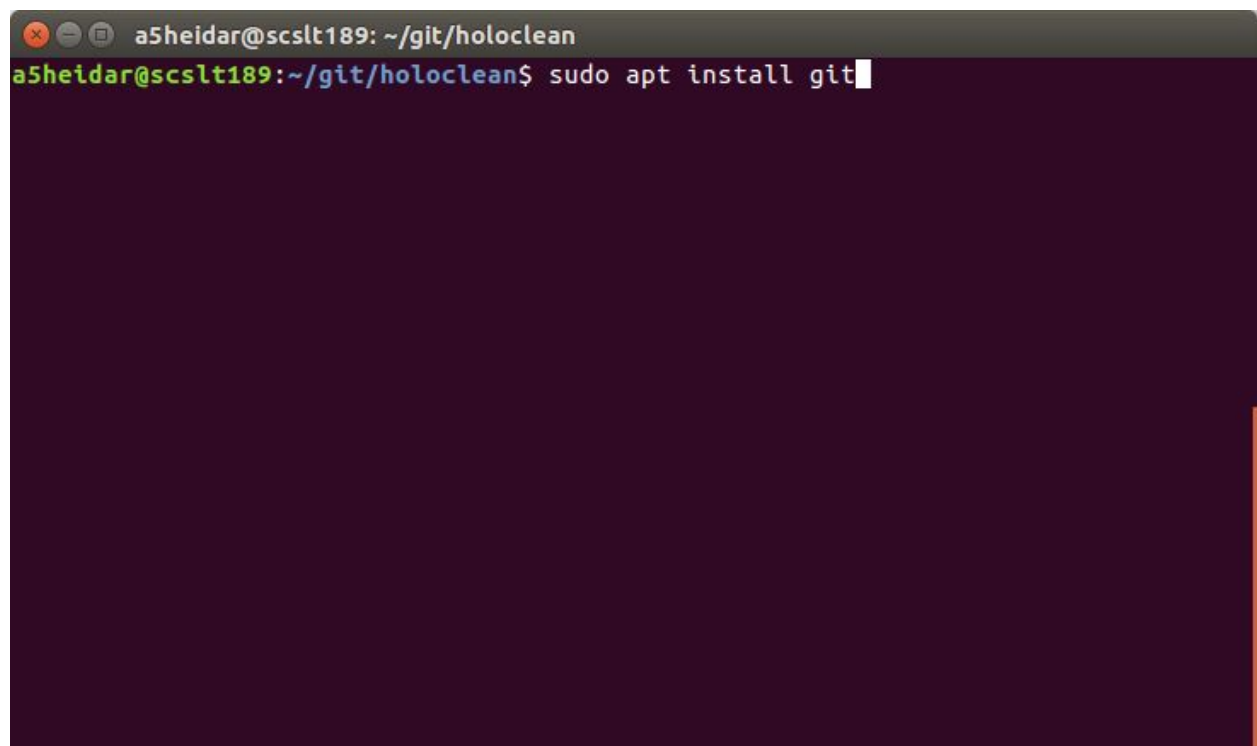
- Linux
- Github Account
- Internet access

Open your terminal

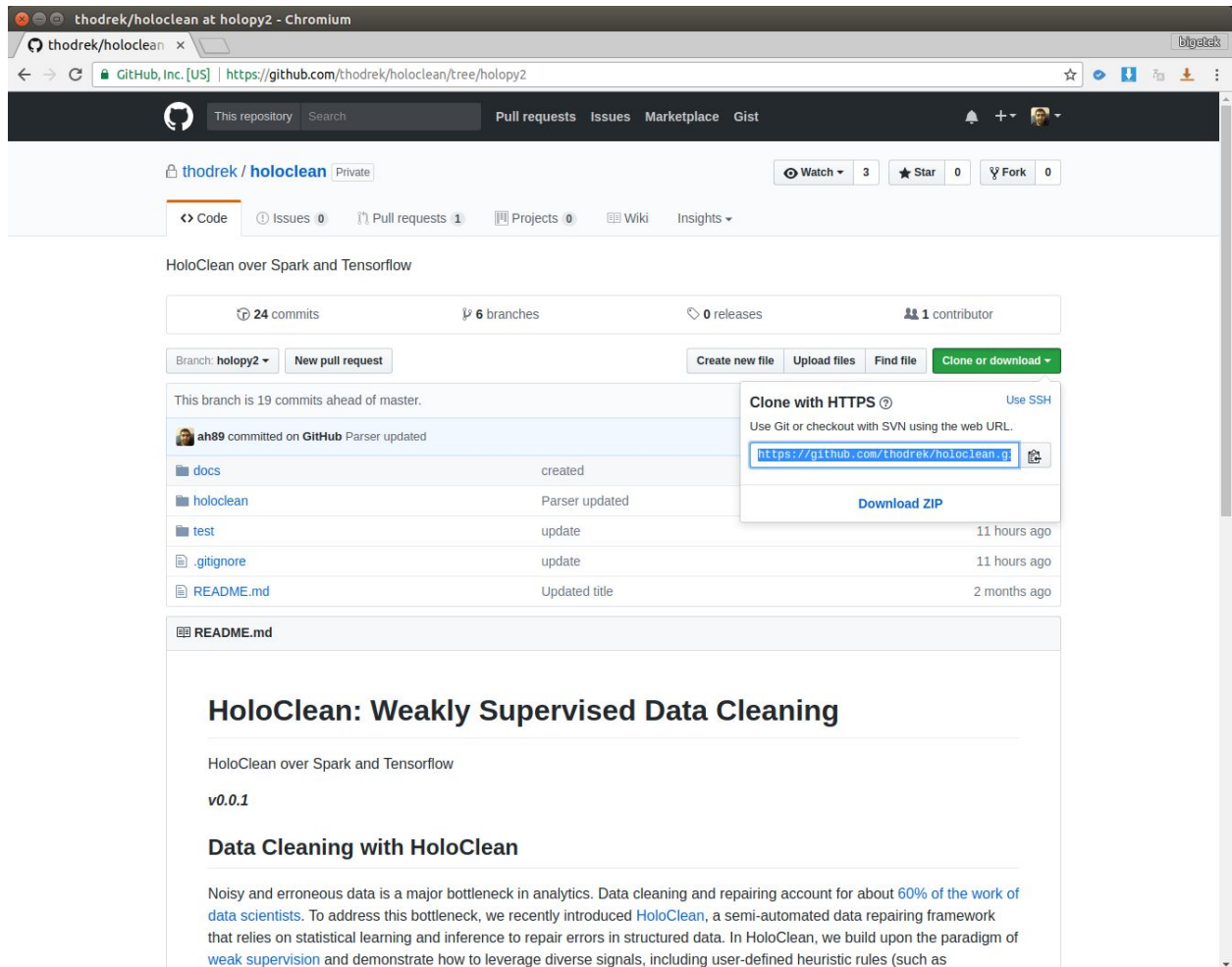
A terminal window with a dark purple background. The title bar shows 'a5heidar@scslt189: ~/git/holoclean'. The prompt 'a5heidar@scslt189:~/git/holoclean\$' is followed by a white cursor.

For step you need to **install git** for this you can use this command

`sudo apt install git`

A terminal window with a dark purple background. The title bar shows 'a5heidar@scslt189: ~/git/holoclean'. The prompt 'a5heidar@scslt189:~/git/holoclean\$' is followed by the command 'sudo apt install git' and a white cursor.

And after that you can access to the git directory that given to you or you can get it after invited to a github project. For getting clone address go to your dashboard in your github account and on the top right click on download and clone and get the address clone



Copy that address and keep it!

For a project first you need to make directory in your local machine so we made folder with name 'git' on my machine and then go in that direction we want to clone the data from that git that we get it's address from our dashboard as you can see in following:

git clone https://github.com/HoloClean/HoloClean-v0.01

After that it will ask your Github username and password and then you have git files on your local machine.

```

james@james-All-Series: ~/Desktop/Holoclean
(myenv)james@james-All-Series:~/Desktop/Holoclean$ git clone https://github.com/HoloClean/HoloClean-v0.01
Cloning into 'HoloClean-v0.01'...
Username for 'https://github.com': j48zheng
Password for 'https://j48zheng@github.com':
remote: Counting objects: 3105, done.
remote: Compressing objects: 100% (310/310), done.
remote: Total 3105 (delta 248), reused 352 (delta 148), pack-reused 2639
Receiving objects: 100% (3105/3105), 55.21 MiB | 532.00 KiB/s, done.
Resolving deltas: 100% (1613/1613), done.
Checking connectivity... done.
(myenv)james@james-All-Series:~/Desktop/Holoclean$

```

```

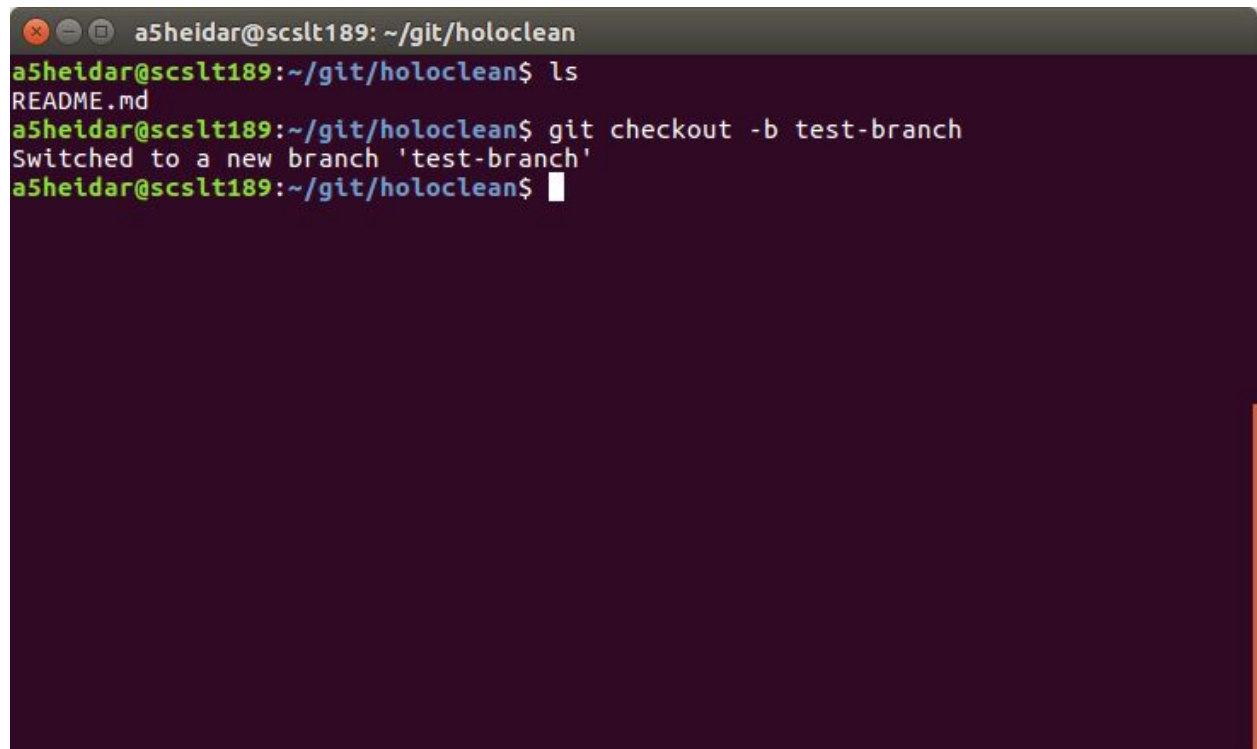
james@james-All-Series: ~/Desktop/Holoclean/HoloClean-v0.01
(myenv)james@james-All-Series:~/Desktop/Holoclean$ git clone https://github.com/HoloClean/HoloClean-v0.01
Cloning into 'HoloClean-v0.01'...
Username for 'https://github.com': j48zheng
Password for 'https://j48zheng@github.com':
remote: Counting objects: 3105, done.
remote: Compressing objects: 100% (310/310), done.
remote: Total 3105 (delta 248), reused 352 (delta 148), pack-reused 2639
Receiving objects: 100% (3105/3105), 55.21 MiB | 532.00 KiB/s, done.
Resolving deltas: 100% (1613/1613), done.
Checking connectivity... done.
(myenv)james@james-All-Series:~/Desktop/Holoclean$ ls
Anaconda-2.3.0-Linux-x86_64.sh  HoloClean-v0.01  spark-2.2.1-bin-hadoop2.7.tgz
derby.log                     metastore_db
Holoclean How to.docx         spark-2.2.1-bin-hadoop2.7
(myenv)james@james-All-Series:~/Desktop/Holoclean$ cd HoloClean-v0.01/
(myenv)james@james-All-Series:~/Desktop/Holoclean/HoloClean-v0.01$ ls
README.md
(myenv)james@james-All-Series:~/Desktop/Holoclean/HoloClean-v0.01$ git checkout holospark
Branch holospark set up to track remote branch holospark from origin.
Switched to a new branch 'holospark'
(myenv)james@james-All-Series:~/Desktop/Holoclean/HoloClean-v0.01$ ls
book-data  Holoclean_demo.ipynb  python-package-requirement.txt  script.py  Tutorial
docs       holofusion            README.md                       set_env.sh
holoclean  mysql_script.sh       run.sh                          test
(myenv)james@james-All-Series:~/Desktop/Holoclean/HoloClean-v0.01$

```

2.2 Branching

Git branches are gloriously excellent for safely making and testing changes. We can make branch by following command:

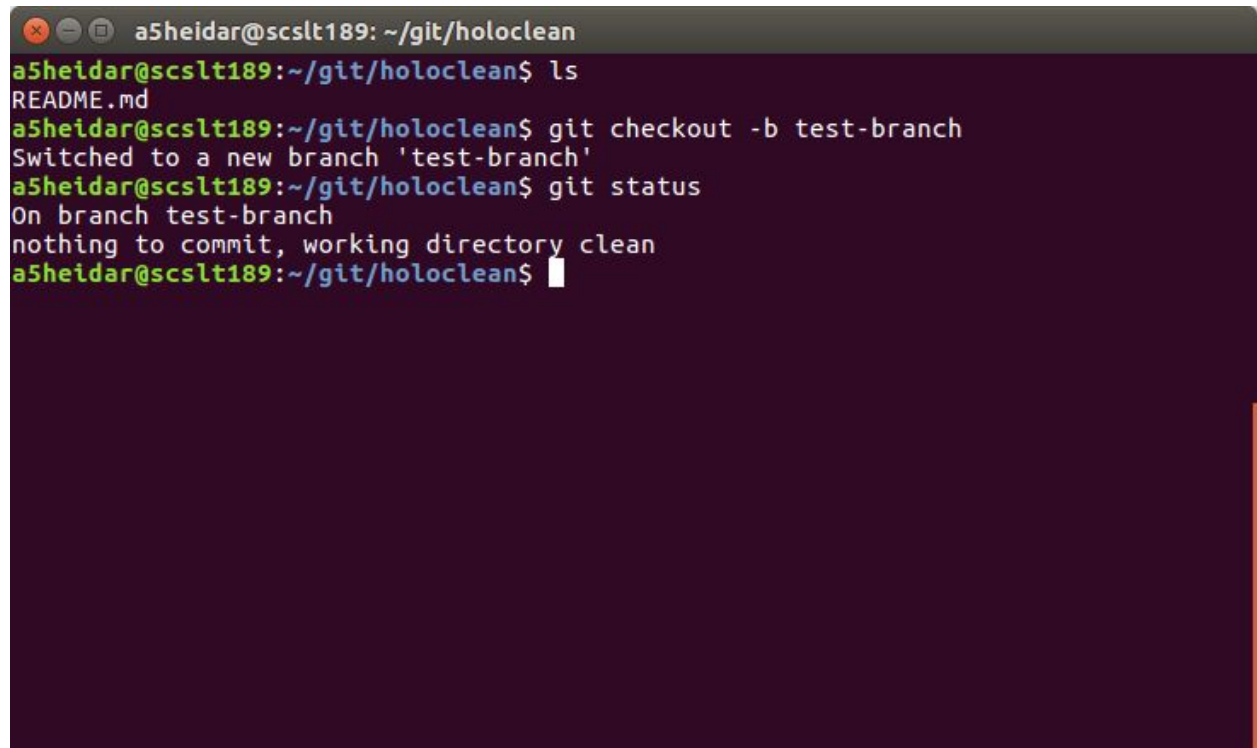
`git checkout -b test-branch`

A terminal window with a dark purple background. The title bar shows 'a5heidar@scslt189: ~/git/holoclean'. The terminal content shows the user running 'ls' and seeing 'README.md'. Then they run 'git checkout -b test-branch' and receive the message 'Switched to a new branch 'test-branch''. The prompt returns to the shell.

```
a5heidar@scslt189: ~/git/holoclean
a5heidar@scslt189:~/git/holoclean$ ls
README.md
a5heidar@scslt189:~/git/holoclean$ git checkout -b test-branch
Switched to a new branch 'test-branch'
a5heidar@scslt189:~/git/holoclean$
```

If you want to see your branch you can use the following commands:

`git status`

A terminal window with a dark purple background and a title bar that reads 'a5heidar@scslt189: ~/git/holoclean'. The terminal shows the following commands and output:

```
a5heidar@scslt189:~/git/holoclean$ ls
README.md
a5heidar@scslt189:~/git/holoclean$ git checkout -b test-branch
Switched to a new branch 'test-branch'
a5heidar@scslt189:~/git/holoclean$ git status
On branch test-branch
nothing to commit, working directory clean
a5heidar@scslt189:~/git/holoclean$
```

2.3 Merge codes from a direction

If you want to start from somewhere that another branch is (not from scratch beginning) you can merge the code from there on your own branch

In here we made a new branch 'testBranch'

```
alireza@scsIt189: ~/git/test/holoclean
Your branch is up-to-date with 'origin/master'.
nothing to commit, working directory clean
alireza@scsIt189:~/git/test/holoclean$ git branch
* master
alireza@scsIt189:~/git/test/holoclean$ cd ..
alireza@scsIt189:~/git/test$ git branch
* master
alireza@scsIt189:~/git/test$ git branch -a
* master
alireza@scsIt189:~/git/test$ cd holoclean/
alireza@scsIt189:~/git/test/holoclean$ git branch -a
* master
remotes/origin/HEAD -> origin/master
remotes/origin/IhabHolo
remotes/origin/ah89-patch-1
remotes/origin/code-skeleton
remotes/origin/dev
remotes/origin/holopy
remotes/origin/holopy2
remotes/origin/holopy3
remotes/origin/master
alireza@scsIt189:~/git/test/holoclean$ git checkout -b testBranch
Switched to a new branch 'testBranch'
alireza@scsIt189:~/git/test/holoclean$
```

i

Now as you can see in the image we want fill our branch with another branch code and start from where that branch is for this we need to merge our branch code with other. The command that can be used is :

`git merge <another branch code>`

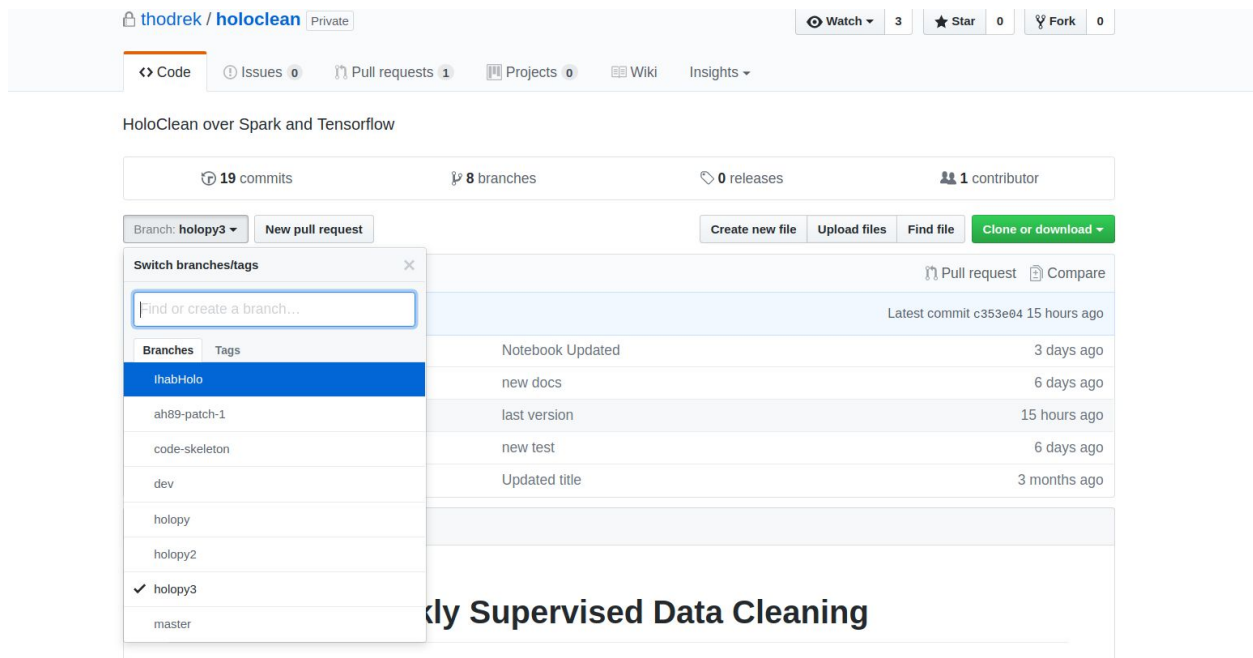
```
alireza@scsIt189: ~/git/test/holoclean
remotes/origin/ah89-patch-1
remotes/origin/code-skeleton
remotes/origin/dev
remotes/origin/holopy
remotes/origin/holopy2
remotes/origin/holopy3
remotes/origin/master
alireza@scsIt189:~/git/test/holoclean$ git checkout -b testBranch
Switched to a new branch 'testBranch'
alireza@scsIt189:~/git/test/holoclean$ git status
On branch testBranch
nothing to commit, working directory clean
alireza@scsIt189:~/git/test/holoclean$ git status
On branch testBranch
nothing to commit, working directory clean
alireza@scsIt189:~/git/test/holoclean$ git merge holopy3
merge: holopy3 - not something we can merge

Did you mean this?
    origin/holopy3
alireza@scsIt189:~/git/test/holoclean$ git merge origin/holopy3
Updating 0c8fcd6..c353e04
Fast-forward
.../.ipynb_checkpoints/Grounding-checkpoint.ipynb | 543 ++++++
```

Now if you list your file you can see that you have same files from your target branch on your own branch.

```
alireza@scsIt189: ~/git/test/holoclean
create mode 100644 holoclean/metastore_db/seg0/ca11.dat
create mode 100644 holoclean/metastore_db/seg0/ca21.dat
create mode 100644 holoclean/metastore_db/seg0/cb1.dat
create mode 100644 holoclean/metastore_db/seg0/cc0.dat
create mode 100644 holoclean/metastore_db/seg0/cd1.dat
create mode 100644 holoclean/metastore_db/seg0/ce1.dat
create mode 100644 holoclean/metastore_db/seg0/cf0.dat
create mode 100644 holoclean/metastore_db/service.properties
create mode 100644 holoclean/models/__init__.py
create mode 100755 holoclean/models/learning_framework.py
create mode 100644 holoclean/models/learning_framework.pyc
create mode 100755 holoclean/repairingEngine.py
create mode 100644 holoclean/test.csv
create mode 100755 holoclean/utilities/data_parser.py
create mode 100644 holoclean/utilities/data_parser.pyc
create mode 100644 test/learning_test.py
create mode 100644 test/prediction_test.py
create mode 100644 test/tests.txt
alireza@scsIt189:~/git/test/holoclean$ ls
docs holoclean README.md test Tutorial
alireza@scsIt189:~/git/test/holoclean$ git status
On branch testBranch
nothing to commit, working directory clean
alireza@scsIt189:~/git/test/holoclean$
```

As you can see you are in your own branch but this branch is on you local so cannot see it on the github website

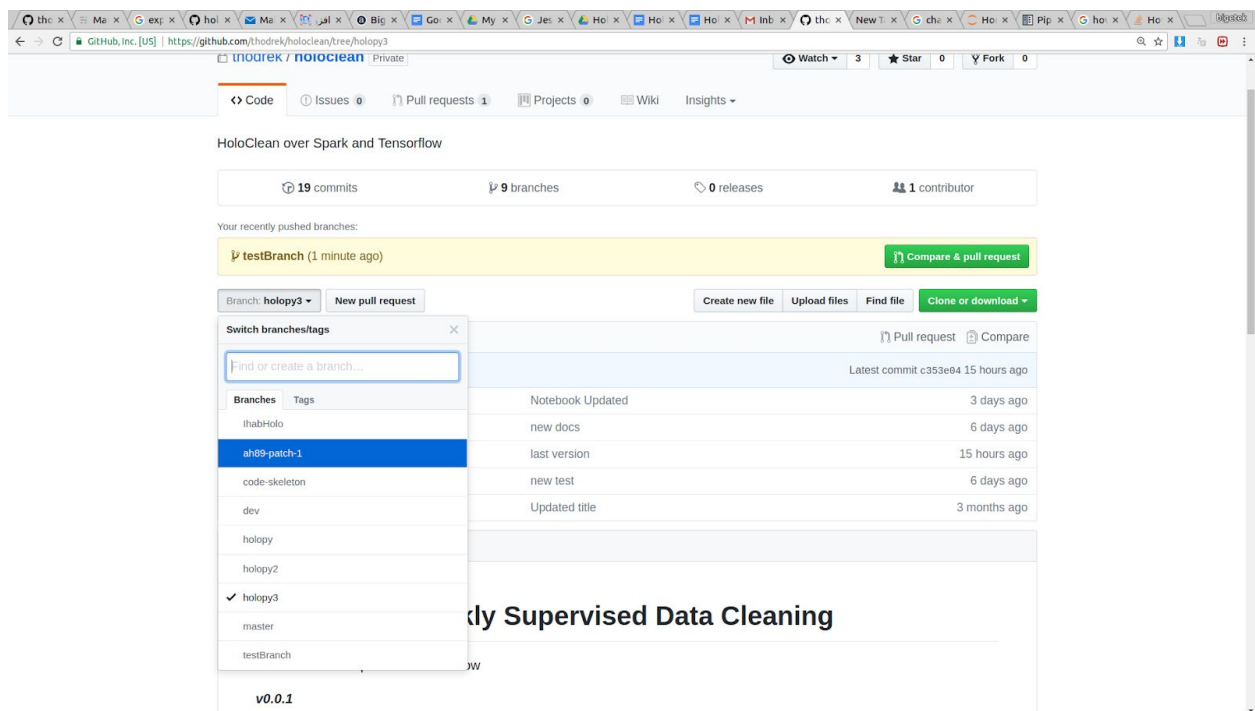


The last step is to push your code to the remote it can be done by

```
git push --set-upstream origin <branchName>
```

```
alireza@scsIt189: ~/git/test/holoclean
alireza@scsIt189:~/git/test/holoclean$ git push --set-upstream origin testBranch
Username for 'https://github.com': ah89
Password for 'https://ah89@github.com':
Total 0 (delta 0), reused 0 (delta 0)
To https://github.com/thodrek/holoclean.git
 * [new branch]      testBranch -> testBranch
Branch testBranch set up to track remote branch testBranch from origin.
alireza@scsIt189:~/git/test/holoclean$
```

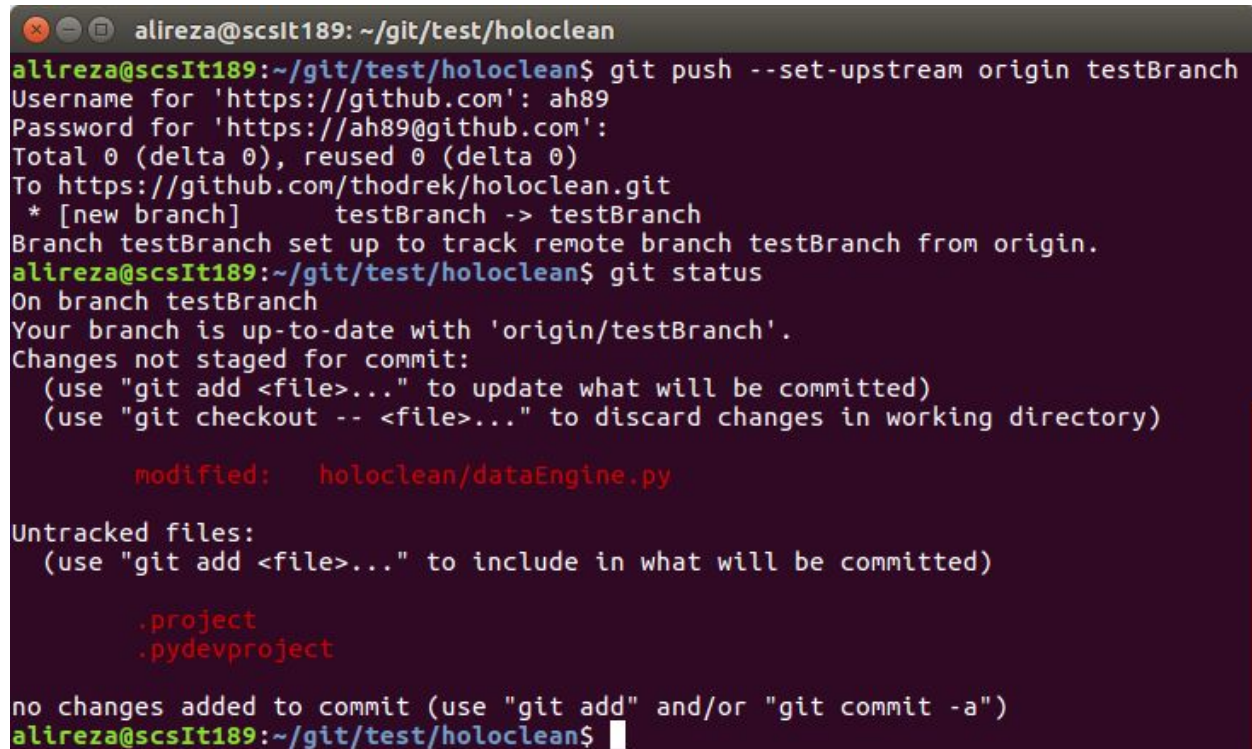
And after that you can see your branch name in the repo



2.4 Push your code to the repo

After you done with programming you can push it in your branch but a logical way is first to what changes have been made so check the status of your branch

`git status`

A terminal window with a dark background and light-colored text. The window title is 'alireza@scsIt189: ~/git/test/holoclean'. The user has executed 'git push --set-upstream origin testBranch', which prompts for a username and password, and then shows the push details. After that, the user runs 'git status', which shows that the branch is up-to-date and lists modified and untracked files.

```
alireza@scsIt189: ~/git/test/holoclean
alireza@scsIt189:~/git/test/holoclean$ git push --set-upstream origin testBranch
Username for 'https://github.com': ah89
Password for 'https://ah89@github.com':
Total 0 (delta 0), reused 0 (delta 0)
To https://github.com/thodrek/holoclean.git
 * [new branch]      testBranch -> testBranch
Branch testBranch set up to track remote branch testBranch from origin.
alireza@scsIt189:~/git/test/holoclean$ git status
On branch testBranch
Your branch is up-to-date with 'origin/testBranch'.
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   holoclean/dataEngine.py

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        .project
        .pydevproject

no changes added to commit (use "git add" and/or "git commit -a")
alireza@scsIt189:~/git/test/holoclean$
```

You can see the files that you have by add them to the git queue you can be sure they will update the remote repo in next push

`git add <file name>`

```
alireza@scsIt189: ~/git/test/holoclean
Username for 'https://github.com': ah89
Password for 'https://ah89@github.com':
Total 0 (delta 0), reused 0 (delta 0)
To https://github.com/thodrek/holoclean.git
* [new branch]      testBranch -> testBranch
Branch testBranch set up to track remote branch testBranch from origin.
alireza@scsIt189:~/git/test/holoclean$ git status
On branch testBranch
Your branch is up-to-date with 'origin/testBranch'.
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   holoclean/dataEngine.py

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        .project
        .pydevproject

no changes added to commit (use "git add" and/or "git commit -a")
alireza@scsIt189:~/git/test/holoclean$ git add holoclean/dataEngine.py
alireza@scsIt189:~/git/test/holoclean$
```

After you added this file you can make comment about your added file by using commit

`git commit -m "Your comment come here!"`

```
alireza@scsIt189: ~/git/test/holoclean
To https://github.com/thodrek/holoclean.git
* [new branch]      testBranch -> testBranch
Branch testBranch set up to track remote branch testBranch from origin.
alireza@scsIt189:~/git/test/holoclean$ git status
On branch testBranch
Your branch is up-to-date with 'origin/testBranch'.
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   holoclean/dataEngine.py

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        .project
        .pydevproject

no changes added to commit (use "git add" and/or "git commit -a")
alireza@scsIt189:~/git/test/holoclean$ git add holoclean/dataEngine.py
alireza@scsIt189:~/git/test/holoclean$ git commit -m "This is just a test"
[testBranch fe1c742] This is just a test
1 file changed, 2 insertions(+)
alireza@scsIt189:~/git/test/holoclean$
```

Now your code has been added to git queue. You might have some other changes and want to add them to with proper comment after that you need to push the code to your remote repo you can do it by using

`git push`


```

alireza@scsIt189: ~/git/test/holoclean
To https://github.com/thodrek/holoclean.git
* [new branch]      testBranch -> testBranch
Branch testBranch set up to track remote branch testBranch from origin.
alireza@scsIt189:~/git/test/holoclean$ git status
On branch testBranch
Your branch is up-to-date with 'origin/testBranch'.
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   holoclean/dataEngine.py

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        .project
        .pydevproject

no changes added to commit (use "git add" and/or "git commit -a")
alireza@scsIt189:~/git/test/holoclean$ git add holoclean/dataEngine.py
alireza@scsIt189:~/git/test/holoclean$ git commit -m "This is just a test"
[testBranch fe1c742] This is just a test
 1 file changed, 2 insertions(+)
alireza@scsIt189:~/git/test/holoclean$ git push

```

And you can see the push has been done successfully.

```

alireza@scsIt189: ~/git/test/holoclean
git config --global push.default simple

When push.default is set to 'matching', git will push local branches
to the remote branches that already exist with the same name.

Since Git 2.0, Git defaults to the more conservative 'simple'
behavior, which only pushes the current branch to the corresponding
remote branch that 'git pull' uses to update the current branch.

See 'git help config' and search for 'push.default' for further information.
(the 'simple' mode was introduced in Git 1.7.11. Use the similar mode
'current' instead of 'simple' if you sometimes use older versions of Git)

Username for 'https://github.com': ah89
Password for 'https://ah89@github.com':
Counting objects: 4, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 390 bytes | 0 bytes/s, done.
Total 4 (delta 3), reused 0 (delta 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To https://github.com/thodrek/holoclean.git
   c353e04..fe1c742  testBranch -> testBranch
alireza@scsIt189:~/git/test/holoclean$

```

2.5 Push your code to the repo

Sometimes you might need to ask merge your code to master branch which is the original branch but before that you need to ask for review in that condition you should make a pull request. Pull request need the branch name that you want to merge and the target branch name and directory it's command scheme comes in following:

```
git request-pull <start> <url> [<end>]
```

For example if you want to feel this scheme it would be like:

```
git request-pull test-branch https://github.com/thodrek/holoclean.git master
```

In this command make a pull request to merge our test branch to the master branch with given url.

2.6 clone a specific branch

For starting programming you need to start to continue a specific branch codes so at first we need list of branches , got to a git direction and use following command :

```
git branch -a
```



```
a5heidar@scslt189: ~/git/holoclean
Password for 'https://ah89@github.com':
remote: Counting objects: 493, done.
remote: Compressing objects: 100% (72/72), done.
remote: Total 493 (delta 43), reused 14 (delta 1), pack-reused 412
Receiving objects: 100% (493/493), 76.35 KiB | 0 bytes/s, done.
Resolving deltas: 100% (283/283), done.
Checking connectivity... done.
a5heidar@scslt189:~/git$ ls
holoclean
a5heidar@scslt189:~/git$ git branch -a
fatal: Not a git repository (or any of the parent directories): .git
a5heidar@scslt189:~/git$ cd holoclean/
a5heidar@scslt189:~/git/holoclean$ ls
README.md
a5heidar@scslt189:~/git/holoclean$ git branch -a
* master
  remotes/origin/HEAD -> origin/master
  remotes/origin/ah89-patch-1
  remotes/origin/code-skeleton
  remotes/origin/dev
  remotes/origin/holopy
  remotes/origin/holopy2
  remotes/origin/master
a5heidar@scslt189:~/git/holoclean$
```

The next step is to clone the code from specific branch to our local repo. This act can be done with following command :

`git checkout [name of branch]`

Before going further you need initial you git directory in your local machine with following command :

`git init`

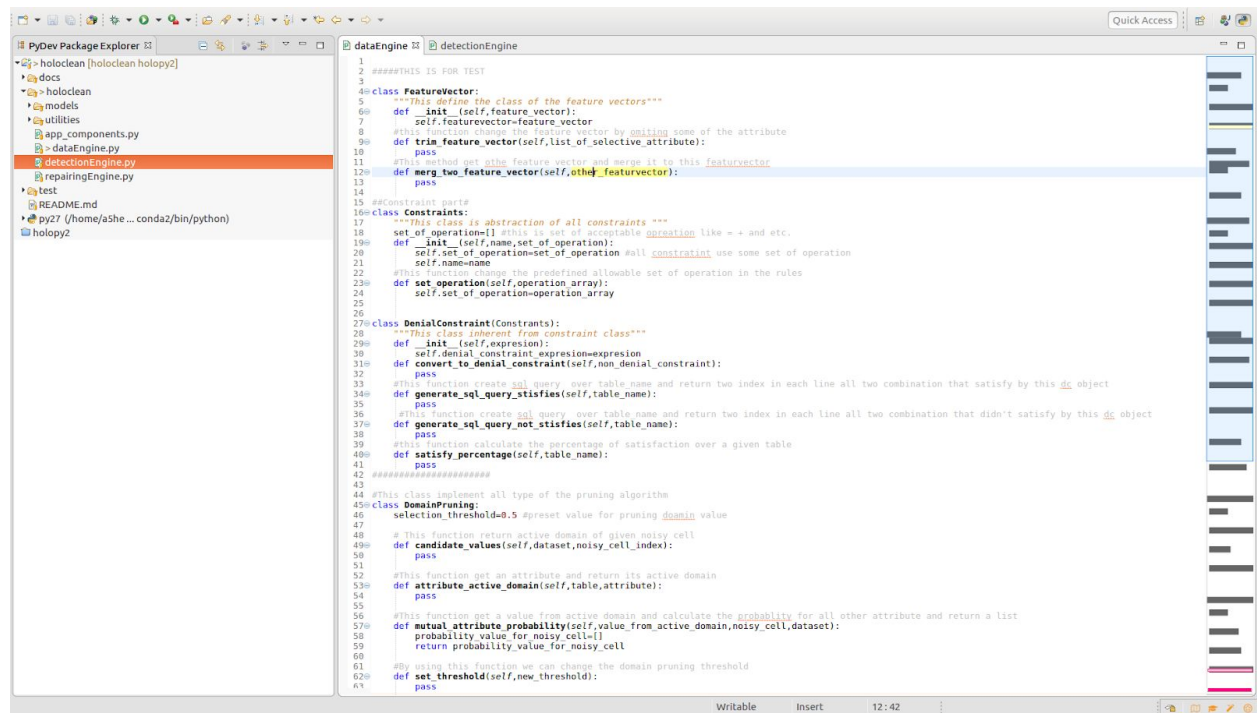
In our example we mean our /git directory as you can see in following screenshot

```
a5heidar@scslt189: ~/git
a5heidar@scslt189:~/git$ git checkout -b holopy2 remotes/origin/holopy
fatal: Not a git repository (or any of the parent directories): .git
a5heidar@scslt189:~/git$ git init
Initialized empty Git repository in /home/a5heidar/git/.git/
a5heidar@scslt189:~/git$
```

Now is the time to get codes of holopy2 branch from repo

```
a5heidar@scslt189: ~/git/holoclean
a5heidar@scslt189:~/git/holoclean$ git checkout holopy2
Branch holopy2 set up to track remote branch holopy2 from origin.
Switched to a new branch 'holopy2'
a5heidar@scslt189:~/git/holoclean$ git branch -a
* holopy2
  master
  remotes/origin/HEAD -> origin/master
  remotes/origin/ah89-patch-1
  remotes/origin/code-skeleton
  remotes/origin/dev
  remotes/origin/holopy
  remotes/origin/holopy2
  remotes/origin/master
a5heidar@scslt189:~/git/holoclean$ git status
On branch holopy2
Your branch is up-to-date with 'origin/holopy2'.
nothing to commit, working directory clean
a5heidar@scslt189:~/git/holoclean$ git pull
Username for 'https://github.com': ah89
Password for 'https://ah89@github.com':
Already up-to-date.
a5heidar@scslt189:~/git/holoclean$ ls
docs holoclean README.md test
a5heidar@scslt189:~/git/holoclean$
```

We changed something in the code and add comment on the first line of script:



```
1 2 #####THIS IS FOR TEST
3
4 class FeatureVector:
5     """This define the class of the feature vectors"""
6     def __init__(self, feature_vector):
7         self.feature_vector = feature_vector
8         #This function change the feature vector by omitting some of the attribute
9     def trim_feature_vector(self, list_of_selective_attribute):
10        pass
11    #This method get other feature vector and merge it to this feature vector
12    def merge_two_feature_vector(self, other_feature_vector):
13        pass
14
15 #Constraint part
16 class Constraints:
17     """This class is abstraction of all constraints"""
18     set_of_operation = [] #This is set of acceptable operation like = + and etc.
19     def __init__(self, name, set_of_operation):
20         self.set_of_operation = set_of_operation #All constraint use some set of operation
21         self.name = name
22     #This function change the predefined allowable set of operation in the rules
23     def set_operation(self, operation_array):
24         self.set_of_operation = operation_array
25
26
27 class DenialConstraint(Constraints):
28     """This class inherit from constraint class"""
29     def __init__(self, expression):
30         self.denial_constraint_expression = expression
31     def convert_to_denial_constraint(self, non_denial_constraint):
32         pass
33     #This function create sql query over table name and return two index in each line all two combination that satisfy by this dc object
34     def generate_sql_query_stisfies(self, table_name):
35         pass
36     #This function create sql query over table name and return two index in each line all two combination that didn't satisfy by this dc object
37     def generate_sql_query_not_stisfies(self, table_name):
38         pass
39     #This function calculate the percentage of satisfaction over a given table
40     def satisfy_percentage(self, table_name):
41         pass
42     #####
43
44 #This class implement all type of the pruning algorithm
45 class DomainPruning:
46     selection_threshold = 0.5 #preset value for pruning domain value
47
48     #This function return active domain of given noisy cell
49     def candidate_values(self, dataset, noisy_cell_index):
50         pass
51
52     #This function get an attribute and return its active domain
53     def attribute_active_domain(self, table, attribute):
54         pass
55
56     #This function give a new domain active domain and calculate the probability for all other attribute and return a list
57     def mutual_attribute_probability(self, value_from_active_domain, noisy_cell, dataset):
58         probability_value_for_noisy_cell = []
59         return probability_value_for_noisy_cell
60
61     #By using this function we can change the domain pruning threshold
62     def set_threshold(self, new_threshold):
63         pass
```

With status we see this changes in the command line :

git status

```
a5heidar@scslt189: ~/git/holoclean
a5heidar@scslt189:~/git/holoclean$ git status
On branch holopy2
Your branch is up-to-date with 'origin/holopy2'.
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   holoclean/dataEngine.py

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        .project
        .pydevproject
        .pydevproject.bak

no changes added to commit (use "git add" and/or "git commit -a")
a5heidar@scslt189:~/git/holoclean$
```

You get your status and you were in the right branch then you can push your changes to remote repo for this you first should make “commit” so by “add” you specify your file that you want to make commit for it and by commit the comment about that changes will attach.

git add [red file in the status for example holoclean/dataEngine.py]
git commit -m "some description like : Test for pushing"

```
a5heidar@scslt189: ~/git/holoclean
a5heidar@scslt189:~/git/holoclean$ git status
On branch holopy2
Your branch is up-to-date with 'origin/holopy2'.
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   holoclean/dataEngine.py

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        .project
        .pydevproject
        .pydevproject.bak

a5heidar@scslt189:~/git/holoclean$ git add holoclean/dataEngine.py
a5heidar@scslt189:~/git/holoclean$ git commit -m "Test for pushing"
[holopy2 9b81848] Test for pushing
 1 file changed, 2 insertions(+), 2 deletions(-)
a5heidar@scslt189:~/git/holoclean$
```

After that you can push all commits that you made in that branch you got in you status
git push


```
a5heidar@scslt189: ~/git/holoclean
.pydevproject.bak

a5heidar@scslt189:~/git/holoclean$ git add holoclean/dataEngine.py
a5heidar@scslt189:~/git/holoclean$ git commit -m "Test for pushing"
[holopy2 9b81848] Test for pushing
 1 file changed, 2 insertions(+), 2 deletions(-)
a5heidar@scslt189:~/git/holoclean$ git push
warning: push.default is unset; its implicit value has changed in
Git 2.0 from 'matching' to 'simple'. To squelch this message
and maintain the traditional behavior, use:

    git config --global push.default matching

To squelch this message and adopt the new behavior now, use:

    git config --global push.default simple

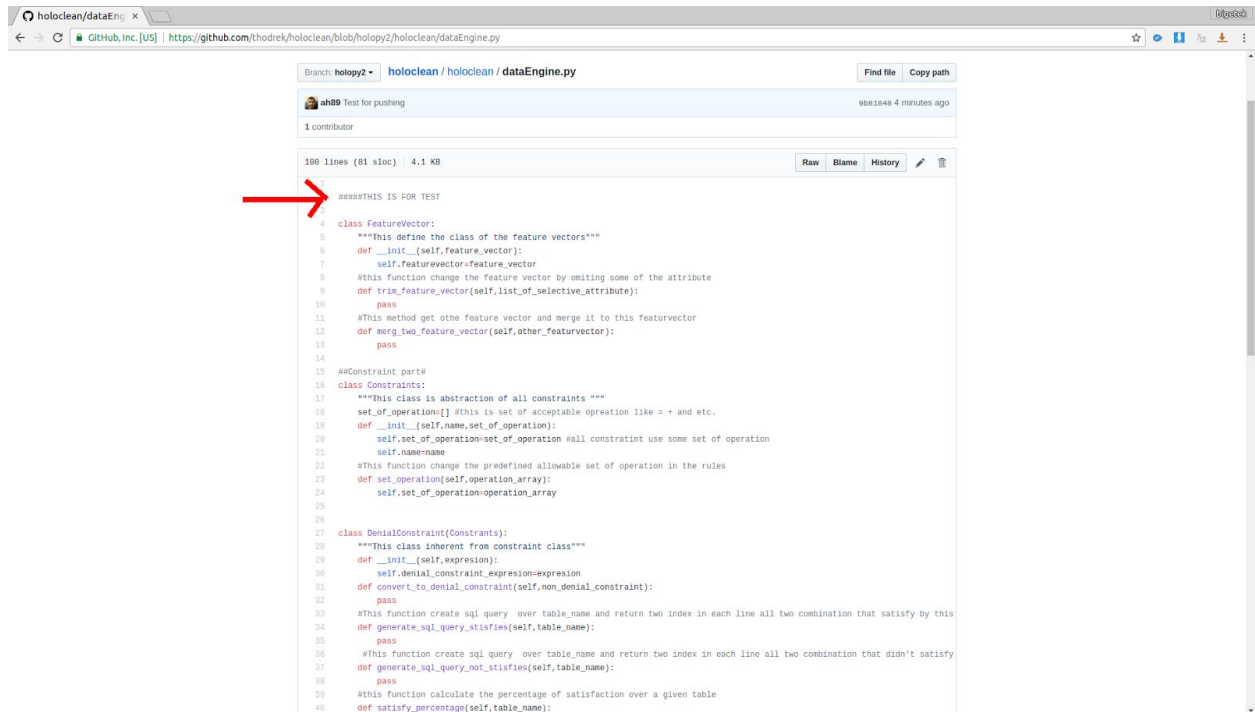
When push.default is set to 'matching', git will push local branches
to the remote branches that already exist with the same name.

Since Git 2.0, Git defaults to the more conservative 'simple'
behavior, which only pushes the current branch to the corresponding
remote branch that 'git pull' uses to update the current branch.

See 'git help config' and search for 'push.default' for further information.
(the 'simple' mode was introduced in Git 1.7.11. Use the similar mode
'current' instead of 'simple' if you sometimes use older versions of Git)

Username for 'https://github.com': ah89
Password for 'https://ah89@github.com':
Counting objects: 4, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 376 bytes | 0 bytes/s, done.
Total 4 (delta 3), reused 0 (delta 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To https://github.com/thodrek/holoclean.git
 1dedf17..9b81848  holopy2 -> holopy2
a5heidar@scslt189:~/git/holoclean$
```

We can go to github.com and check if the codes been pushed correctly.



```
holoclean/dataEngine.py
Branch: holopy2
holoclean / holoclean / dataEngine.py
Find file Copy path
sh89 Test for pushing 98e1e4e 4 minutes ago
1 contributor
100 lines (81 sloc) 4.1 KB
Raw Blame History
#####THIS IS FOR TEST
5 class FeatureVector:
6     """This define the class of the feature vectors"""
7     def __init__(self,feature_vector):
8         self.featurevector=feature_vector
9         #this function change the feature vector by omiting some of the attribute
10        def trim_feature_vector(self,list_of_selective_attribute):
11            pass
12        #This method get othe feature vector and merge it to this featurvector
13        def merg_two_feature_vector(self,other_featurvector):
14            pass
15
16    @Constraint parts
17    class Constraints:
18        """This class is abstraction of all constraints """
19        set_of_operation=[] #this is set of acceptable operation like + and etc.
20        def __init__(self,name,set_of_operation):
21            self.set_of_operation=set_of_operation #all constratint use some set of operation
22            self.name=name
23            #This function change the predefined allowable set of operation in the rules
24            def set_operation(self,operation_array):
25                self.set_of_operation=operation_array
26
27    class DenialConstraint(Constraints):
28        """This class inherit from constraint class"""
29        def __init__(self,expression):
30            self.denial_constraint_expression=expression
31            def convert_to_denial_constraint(self,non_denial_constraint):
32                pass
33            #This function create sql query over table_name and return two index in each line all two combination that satisfy by this
34            def generate_sql_query_stisfies(self,table_name):
35                pass
36            #This function create sql query over table_name and return two index in each line all two combination that didn't satisfy
37            def generate_sql_query_not_stisfies(self,table_name):
38                pass
39            #this function calculate the percentage of satisfaction over a given table
40            def satisfy_percentage(self,table_name):
```

You can also change your branch to other branches by checkout command
git checkout [Some other branches]

2.7 Set SSH on Github

Sometimes you entering username and password are frustrating so you have enough confidence to your machine and so you can set SSH on your github account to avoid enter password every time you push. To this end you should make key on your machine first so at first you can check if you have any key before using following command :

ls -al ~/.ssh

```
alireza@scsIt189:~/.ssh$ ls -al ~/.ssh
total 12
drwx----- 2 alireza alireza 4096 Sep 12 14:56 .
drwxr-xr-x 43 alireza alireza 4096 Sep 12 12:49 ..
-rw-r--r-- 1 alireza alireza 1326 Sep 12 12:40 known_hosts
alireza@scsIt189:~/.ssh$
```

2.7.1 Generating a new SSH key and adding it to the ssh-agent

For creating a new ssh that can be used in your github you should create a new one with email that you create your github account .

```
ssh-keygen -t rsa
```

After this step you can choose passphrase to a security layer if you want to know more see [this](#) but we doesn't enter any passphrase.

```

alireza@scsIt189:~/.ssh$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/alireza/.ssh/id_rsa): ghk
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in ghk.
Your public key has been saved in ghk.pub.
The key fingerprint is:
SHA256:tNUI90D2agU1KtGU4m4M3lHb6pbCHsEWQxFwtyGUs8Y alireza@scsIt189
The key's randomart image is:
+---[RSA 2048]---+
|      .+*+X+o      |
|      .o+0o* o      |
|      ..o*+*oo      |
|      . o+.*Eo      |
|      . = .So*      |
|      . = .+ .      |
|      . ooo.        |
|      . = ..        |
|      . . .         |
+-----[SHA256]-----+
alireza@scsIt189:~/.ssh$ █

```

2.7.2 Adding your SSH key to the ssh-agent

After you made a key you need to introduce it to you ssh agent which manage all you ssh connection first run an agent :

```
eval "$(ssh-agent -s)"
```



```

alireza@scsIt189:~/.ssh$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/alireza/.ssh/id_rsa): ghk
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in ghk.
Your public key has been saved in ghk.pub.
The key fingerprint is:
SHA256:tNUI90D2agU1KtGU4m4M3lHb6pbCHsEWQxFwtyGUs8Y alireza@scsIt189
The key's randomart image is:
+----[RSA 2048]-----+
|      .+*+X+o      |
|      .o+0o* o      |
|      ..o*+*oo      |
|      . o+.*Eo      |
|      . = .So*      |
|      . = .+ .      |
|      . ooo.        |
|      . = ..        |
|      . ..         |
+-----[SHA256]-----+
alireza@scsIt189:~/.ssh$ eval "$(ssh-agent -s)"
Agent pid 16293
alireza@scsIt189:~/.ssh$

```

Then add your key our key name is 'gh.pub'

ssh-add ~/.ssh/<Private key>

```

Enter file in which to save the key (/home/alireza/.ssh/id_rsa): ghk
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in ghk.
Your public key has been saved in ghk.pub.
The key fingerprint is:
SHA256:xaFkuWNmWUHbB06FPxkc4by2xc7hGH2z8mD8fK3G710 alireza@scsIt189
The key's randomart image is:
+----[RSA 2048]-----+
|      o*++++o      |
|      o+B.=o      |
|      oo.=.+o      |
|      =. .++      |
|      .S. +=..      |
|      ..0 oo      |
|      o*+.E      |
|      . 0 =      |
|      ..B=      |
+-----[SHA256]-----+
alireza@scsIt189:~/.ssh$ eval "$(ssh-agent -s)"
Agent pid 17294
alireza@scsIt189:~/.ssh$ ssh-add ~/.ssh/ghk
Identity added: /home/alireza/.ssh/ghk (/home/alireza/.ssh/ghk)
alireza@scsIt189:~/.ssh$

```

Note : You might can see error because you haven't set your private key permission you change the permission of private key to 600 .

```
sudo chmod 600 ~/.ssh/gh
```

```
sudo chmod 600 ~/.ssh/gh.pub
```

```
+---[RSA 4096]---+
|o.o  .          |
|.. o  . .       |
|ooo. Bo.        |
|o.=.++=+..      |
| o.Oo+.=S o     |
|.+.*=.+. .     |
|.  o o o        |
|....O  . .      |
|oE=*..          |
+---[SHA256]-----+
alireza@scsIt189:~/.ssh$ eval "$(ssh-agent -s)"
Agent pid 15161
alireza@scsIt189:~/.ssh$ ssh-add ~/.ssh/gh.pub
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@                WARNING: UNPROTECTED PRIVATE KEY FILE!          @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
Permissions 0644 for '/home/alireza/.ssh/gh.pub' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
alireza@scsIt189:~/.ssh$ sudo chmod 600 ~/.ssh/gh
[sudo] password for alireza:
alireza@scsIt189:~/.ssh$ sudo chmod 600 ~/.ssh/gh.pub
alireza@scsIt189:~/.ssh$
```

And then add your private key to you agent

1.3 Adding a new SSH key to your GitHub account

Copy your RSA key (this is very sensitive so we need to install an application for this)

```
sudo apt-get install xclip
```

```

alireza@scsIt189: ~
This private key will be ignored.
alireza@scsIt189:~$ sudo apt-get install xclip
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libreadline5 linux-headers-4.10.0-28 linux-headers-4.10.0-28-generic
  linux-image-4.10.0-28-generic linux-image-extra-4.10.0-28-generic
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  xclip
0 upgraded, 1 newly installed, 0 to remove and 1 not upgraded.
Need to get 17.0 kB of archives.
After this operation, 72.7 kB of additional disk space will be used.
Get:1 http://ca.archive.ubuntu.com/ubuntu xenial/universe amd64 xclip amd64 0.12
+svn84-4 [17.0 kB]
Fetched 17.0 kB in 0s (57.0 kB/s)
Selecting previously unselected package xclip.
(Reading database ... 403441 files and directories currently installed.)
Preparing to unpack .../xclip_0.12+svn84-4_amd64.deb ...
Unpacking xclip (0.12+svn84-4) ...
Processing triggers for man-db (2.7.5-1) ...
Setting up xclip (0.12+svn84-4) ...
alireza@scsIt189:~$

```

then copy your public key using xclip as follow :


`xclip -sel clip < [Directory of your .pub file]`

```




Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in ghk.
Your public key has been saved in ghk.pub.
The key fingerprint is:
SHA256:xaFkuWNMwUHbB06FPxkc4by2xc7hGH2z8mD8fK3G710 alireza@scsIt189
The key's randomart image is:
+---[RSA 2048]---+
|      o*+++o      |
|      o+B.=o      |
|      oo.=.+o      |
|      =. .++      |
|      .S.  +.=..   |
|      ..0 oo|      |
|      o*+.E|      |
|      . 0 =|      |
|      ..B=|      |
+-----[SHA256]-----+
alireza@scsIt189:~/.ssh$ eval "$(ssh-agent -s)"
Agent pid 17294
alireza@scsIt189:~/.ssh$ ssh-add ~/.ssh/ghk
Identity added: /home/alireza/.ssh/ghk (/home/alireza/.ssh/ghk)
alireza@scsIt189:~/.ssh$ xclip -sel clip < ~/.ssh/ghk.pub
alireza@scsIt189:~/.ssh$

```

The go to your **github account**> **Settings > SSH and GPG keys > New SSH key or Add SSH key** and then paste your ssh key on 'key' field and click **Add SSH key**

 Search GitHub

Pull requestsIssuesMarketplaceExplore



Personal settings

Profile

Account

Emails

Notifications

Billing

SSH and GPG keys

Security

Blocked users

Repositories

Organizations

Saved replies

Authorized OAuth Apps

Authorized GitHub Apps

Installed GitHub Apps


Developer settings

OAuth Apps

SSH keys

New SSH key

This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

 SSH

githubk

Fingerprint: 29:d8:af:4c:8c:4b:ba:7b:3c:5a:f9:71:bc:e6:e4:f4

Added on 12 Sep 2017

Never used — Read/write

Delete

Check out our guide to [generating SSH keys](#) or troubleshoot [common SSH Problems](#).

GPG keys

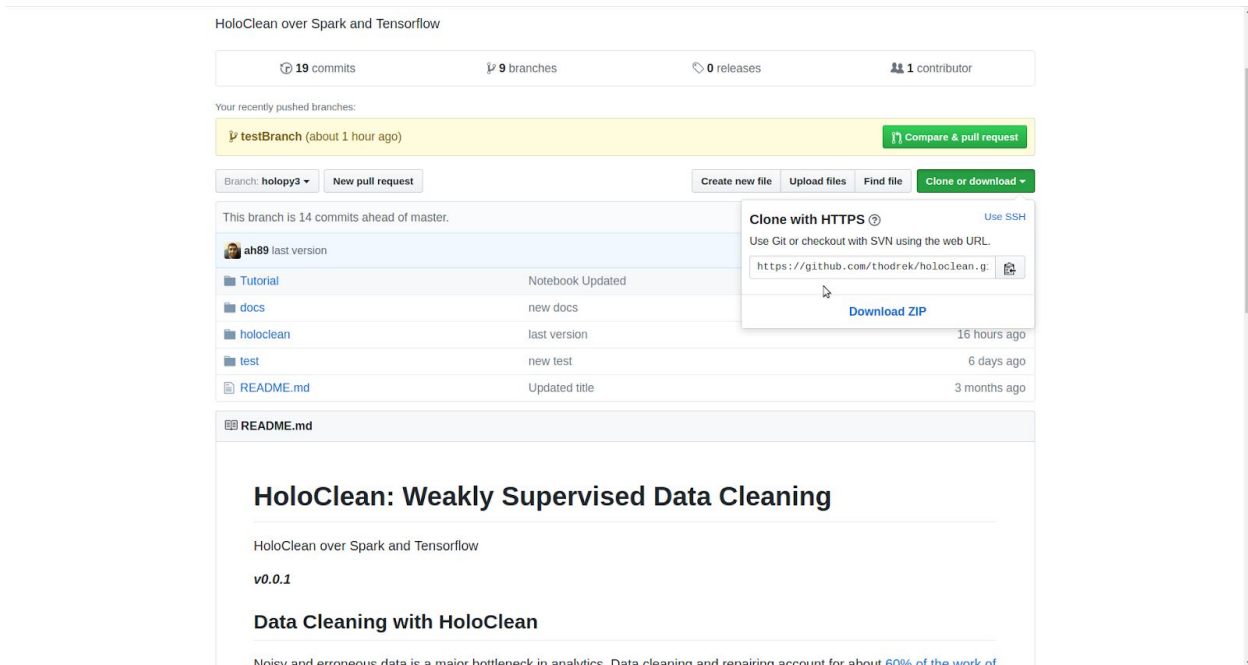
New GPG key

There are no GPG keys with access to your account.

Learn how to [generate a GPG key](#) and add it to your account.

1.4 How use SSH with github

For using it you should clone with SSH download method when you go to github account you can see option for SSH clone address too.



After that you can clone with .git address that you copied to your system

```
alireza@scsIt189: ~/test
alireza@scsIt189:~/test$ git clone git@github.com:thodrek/holoclean.git
Cloning into 'holoclean'...
remote: Counting objects: 868, done.
remote: Compressing objects: 100% (79/79), done.
remote: Total 868 (delta 161), reused 217 (delta 146), pack-reused 636
Receiving objects: 100% (868/868), 282.89 KiB | 0 bytes/s, done.
Resolving deltas: 100% (544/544), done.
Checking connectivity... done.
alireza@scsIt189:~/test$ ls
holoclean
alireza@scsIt189:~/test$
```

3- Project

Each session might run multiple project in it so it has method in it that by calling it a project will be started

4- Reading data

This part a reading object by getting a directory for dataset and constraints and noisy cells (that created in the error detection process) try to read them and pass them to the parsing package

5- Parsing

6- Learning

This package has three module one make the data manipulation for training and testing data, second part by a model that we choose start to learn model with preset iteration and step , we can change this two parameters

7- Interaction and feedback

This package has to communicate with the users show result to them and grab their