C.A.S.A. in C.V.

Context Aware Security Analytics in Computer Vision

Paola Barra

UNIVERSITÀ
DEGLI STUDI
DI SALERNO

Tools

Git and GitHub

Anaconda … & friends

Python 3.6

What is Machine Learning?

UNIVERSITÀ DEGLI STUDI DI SALERNO

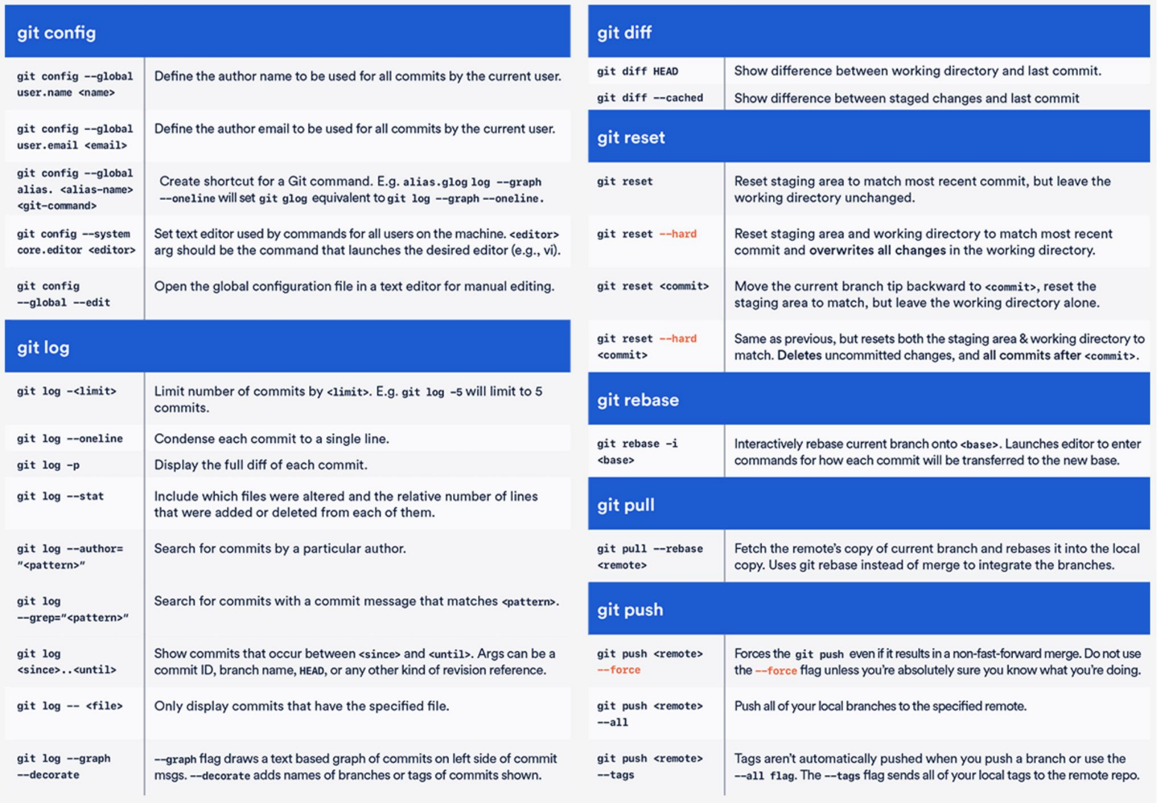
Git

- is a command-line tool used for version control;
- runs locally on a computer;
- is completely self-sufficient;
- it does not require the use of any external, cloud hosting service.

Most used Git command-line tools:

- commit
- push
- pull
- revert
- ...

Git command-line tools



git config	
`git config --global user.name <name>`	Define the author name to be used for all commits by the current user.
`git config --global user.email <email>`	Define the author email to be used for all commits by the current user.
`git config --global alias. <alias-name> <git-command>`	Create shortcut for a Git command. E.g. `alias.glog log --graph --oneline` will set `git glog` equivalent to `git log --graph --oneline.`
`git config --system core.editor <editor>`	Set text editor used by commands for all users on the machine. `<editor>` arg should be the command that launches the desired editor (e.g., vi).
`git config --global --edit`	Open the global configuration file in a text editor for manual editing.

git log	
`git log -<limit>`	Limit number of commits by `<limit>`. E.g. `git log -5` will limit to 5 commits.
`git log --oneline`	Condense each commit to a single line.
`git log -p`	Display the full diff of each commit.
`git log --stat`	Include which files were altered and the relative number of lines that were added or deleted from each of them.
`git log --author= "<pattern>"`	Search for commits by a particular author.
`git log --grep="<pattern>"`	Search for commits with a commit message that matches `<pattern>`.
`git log <since>..<until>`	Show commits that occur between `<since>` and `<until>`. Args can be a commit ID, branch name, `HEAD`, or any other kind of revision reference.
`git log -- <file>`	Only display commits that have the specified file.
`git log --graph --decorate`	`--graph` flag draws a text based graph of commits on left side of commit msgs. `--decorate` adds names of branches or tags of commits shown.

git diff	
`git diff HEAD`	Show difference between working directory and last commit.
`git diff --cached`	Show difference between staged changes and last commit

git reset	
`git reset`	Reset staging area to match most recent commit, but leave the working directory unchanged.
`git reset --hard`	Reset staging area and working directory to match most recent commit and **overwrites all changes** in the working directory.
`git reset <commit>`	Move the current branch tip backward to `<commit>`, reset the staging area to match, but leave the working directory alone.
`git reset --hard <commit>`	Same as previous, but resets both the staging area & working directory to match. **Deletes** uncommitted changes, and **all commits after** `<commit>`.

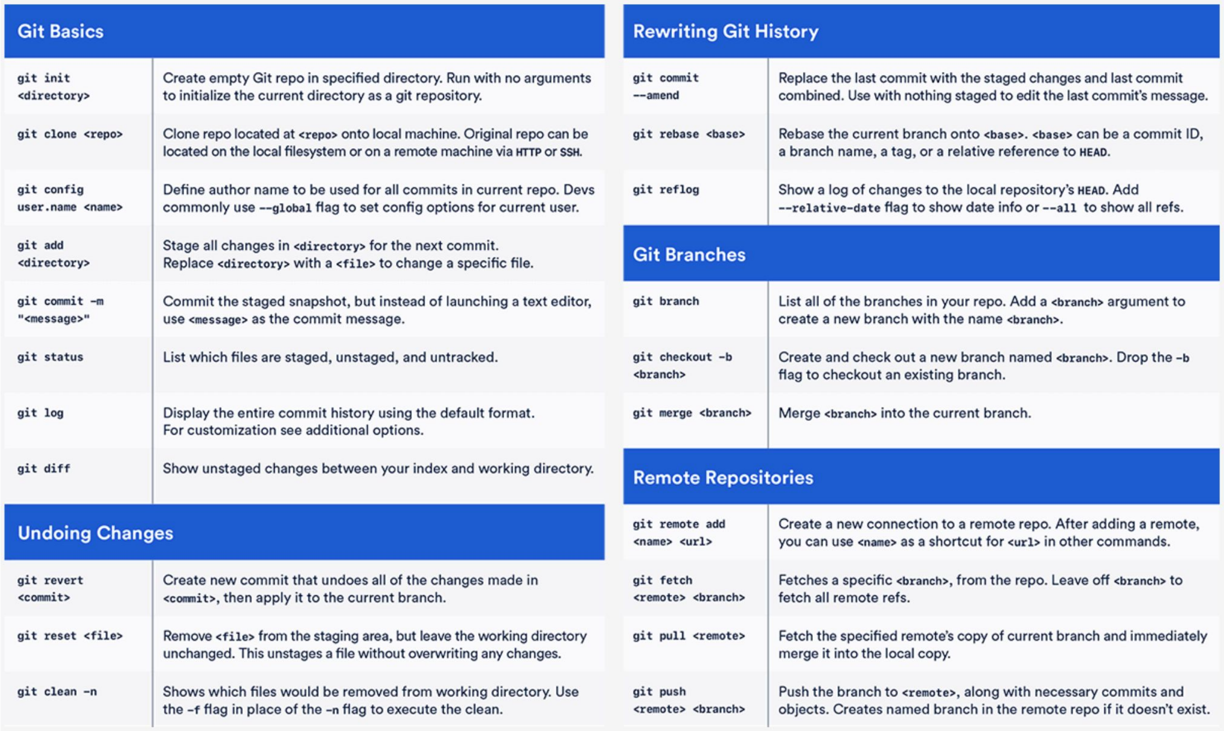
git rebase	
`git rebase -i <base>`	Interactively rebase current branch onto `<base>`. Launches editor to enter commands for how each commit will be transferred to the new base.

git pull	
`git pull --rebase <remote>`	Fetch the remote's copy of current branch and rebases it into the local copy. Uses git rebase instead of merge to integrate the branches.

git push	
`git push <remote> --force`	Forces the `git push` even if it results in a non-fast-forward merge. Do not use the `--force` flag unless you're absolutely sure you know what you're doing.
`git push <remote> --all`	Push all of your local branches to the specified remote.
`git push <remote> --tags`	Tags aren't automatically pushed when you push a branch or use the `--all flag`. The `--tags` flag sends all of your local tags to the remote repo.

UNIVERSITÀ DEGLI STUDI DI SALERNO

Git command-line tools



Git Basics

`git init <directory>`	Create empty Git repo in specified directory. Run with no arguments to initialize the current directory as a git repository.
`git clone <repo>`	Clone repo located at `<repo>` onto local machine. Original repo can be located on the local filesystem or on a remote machine via `HTTP` or `SSH`.
`git config user.name <name>`	Define author name to be used for all commits in current repo. Devs commonly use `--global` flag to set config options for current user.
`git add <directory>`	Stage all changes in `<directory>` for the next commit. Replace `<directory>` with a `<file>` to change a specific file.
`git commit -m "<message>"`	Commit the staged snapshot, but instead of launching a text editor, use `<message>` as the commit message.
`git status`	List which files are staged, unstaged, and untracked.
`git log`	Display the entire commit history using the default format. For customization see additional options.
`git diff`	Show unstaged changes between your index and working directory.

Undoing Changes

`git revert <commit>`	Create new commit that undoes all of the changes made in `<commit>`, then apply it to the current branch.
`git reset <file>`	Remove `<file>` from the staging area, but leave the working directory unchanged. This unstages a file without overwriting any changes.
`git clean -n`	Shows which files would be removed from working directory. Use the `-f` flag in place of the `-n` flag to execute the clean.

Rewriting Git History

`git commit --amend`	Replace the last commit with the staged changes and last commit combined. Use with nothing staged to edit the last commit's message.
`git rebase <base>`	Rebase the current branch onto `<base>`. `<base>` can be a commit ID, a branch name, a tag, or a relative reference to `HEAD`.
`git reflog`	Show a log of changes to the local repository's `HEAD`. Add `--relative-date` flag to show date info or `--all` to show all refs.

Git Branches

`git branch`	List all of the branches in your repo. Add a `<branch>` argument to create a new branch with the name `<branch>`.
`git checkout -b <branch>`	Create and check out a new branch named `<branch>`. Drop the `-b` flag to checkout an existing branch.
`git merge <branch>`	Merge `<branch>` into the current branch.

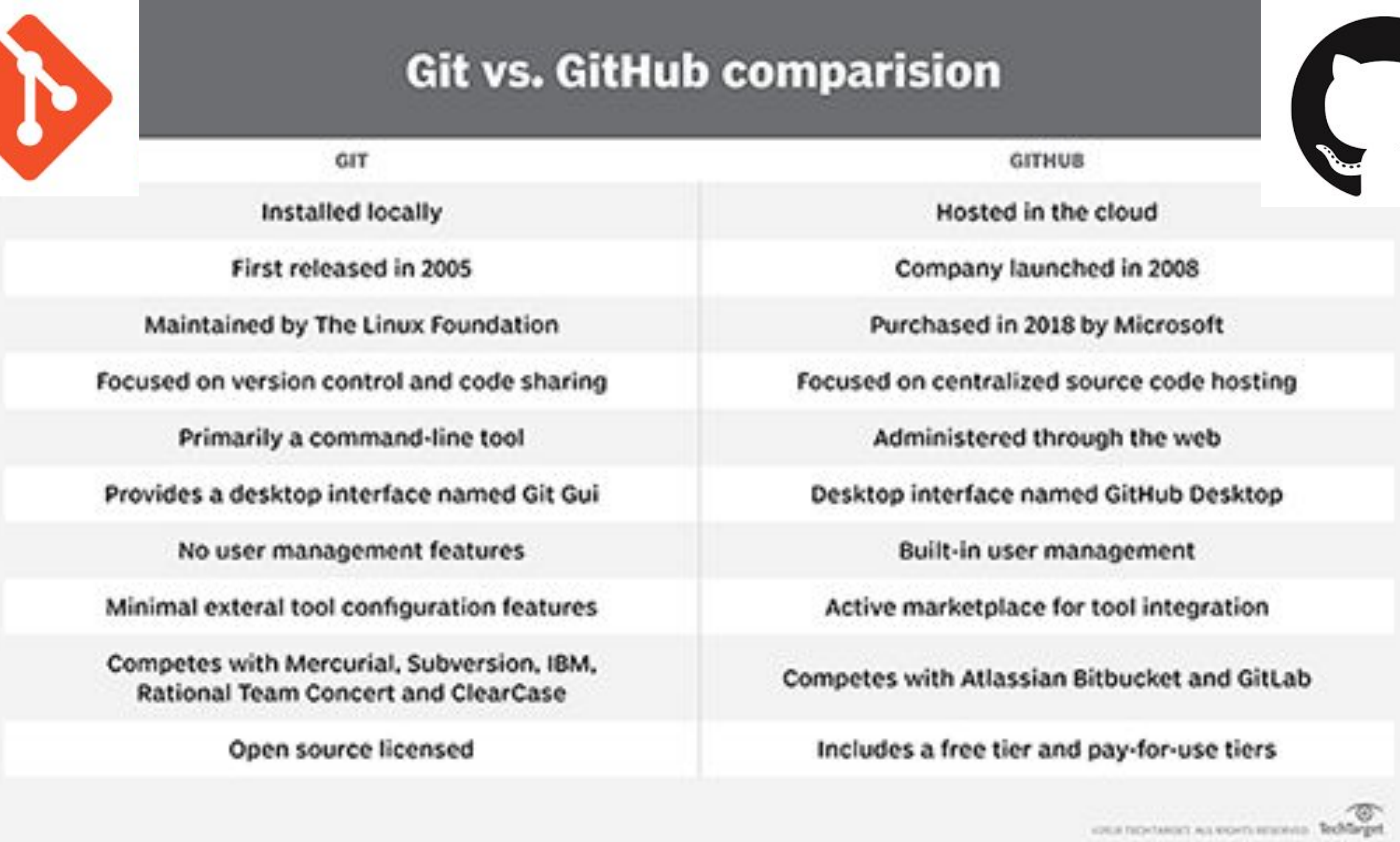
Remote Repositories

`git remote add <name> <url>`	Create a new connection to a remote repo. After adding a remote, you can use `<name>` as a shortcut for `<url>` in other commands.
`git fetch <remote> <branch>`	Fetches a specific `<branch>`, from the repo. Leave off `<branch>` to fetch all remote refs.
`git pull <remote>`	Fetch the specified remote's copy of current branch and immediately merge it into the local copy.
`git push <remote> <branch>`	Push the branch to `<remote>`, along with necessary commits and objects. Creates named branch in the remote repo if it doesn't exist.

UNIVERSITÀ DEGLI STUDI DI SALERNO

GitHub

- is a cloud-based platform built around the Git tool;
- online editing of file;
- is an online service to which developers who use Git can connect and upload or download resources.



Git vs. GitHub comparision

GIT	GITHUB
Installed locally	Hosted in the cloud
First released in 2005	Company launched in 2008
Maintained by The Linux Foundation	Purchased in 2018 by Microsoft
Focused on version control and code sharing	Focused on centralized source code hosting
Primarily a command-line tool	Administered through the web
Provides a desktop interface named Git Gui	Desktop interface named GitHub Desktop
No user management features	Built-in user management
Minimal exteral tool configuration features	Active marketplace for tool integration
Competes with Mercurial, Subversion, IBM, Rational Team Concert and ClearCase	Competes with Atlassian Bitbucket and GitLab
Open source licensed	Includes a free tier and pay-for-use tiers

Git & GitHub

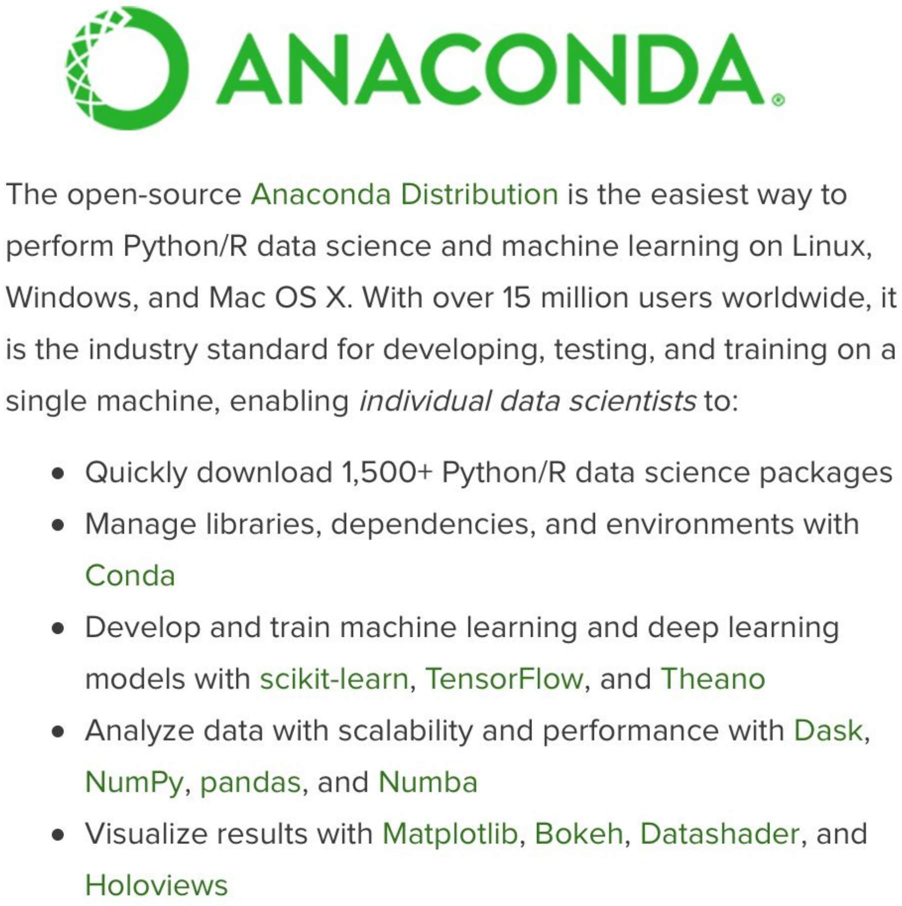
As you can see, Git and GitHub are clearly different in features and purpose

- they compliment each other
- they are not competitors

Integrating Git and GitHub together is a powerful combination.

Let’s start

1) Set up a GitHub account https://github.com
2) Download GitHub desktop on you computer https://desktop.github.com
3) Create a repository in local and an URL will uniquely identifies it.
4) With the repository created, developers can install Git locally and **clone** the repository.
5) Developers can make local changes to project files and then **push** the **commit** back to GitHub.
6) To get updates developers use **pull** and **fetch** commands.

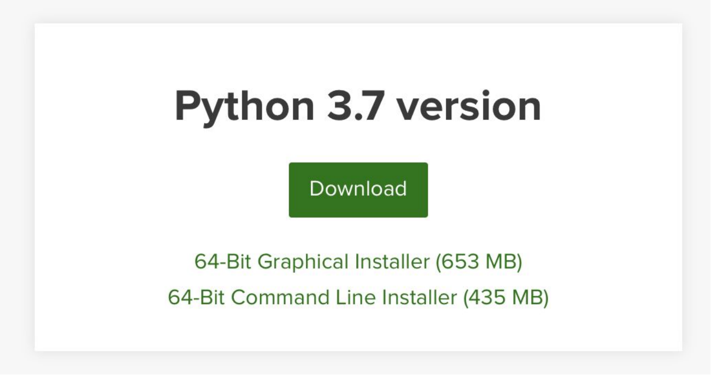


ANACONDA

The open-source Anaconda Distribution is the easiest way to perform Python/R data science and machine learning on Linux, Windows, and Mac OS X. With over 15 million users worldwide, it is the industry standard for developing, testing, and training on a single machine, enabling *individual data scientists* to:

- Quickly download 1,500+ Python/R data science packages
- Manage libraries, dependencies, and environments with Conda
- Develop and train machine learning and deep learning models with scikit-learn, TensorFlow, and Theano
- Analyze data with scalability and performance with Dask, NumPy, pandas, and Numba
- Visualize results with Matplotlib, Bokeh, Datashader, and Holoviews

Download Anaconda on www.anaconda.com

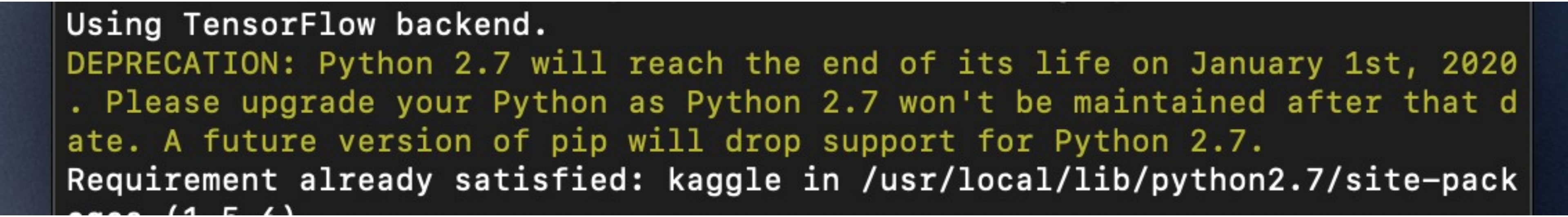


Python 3.7 version

64-Bit Graphical Installer (653 MB)

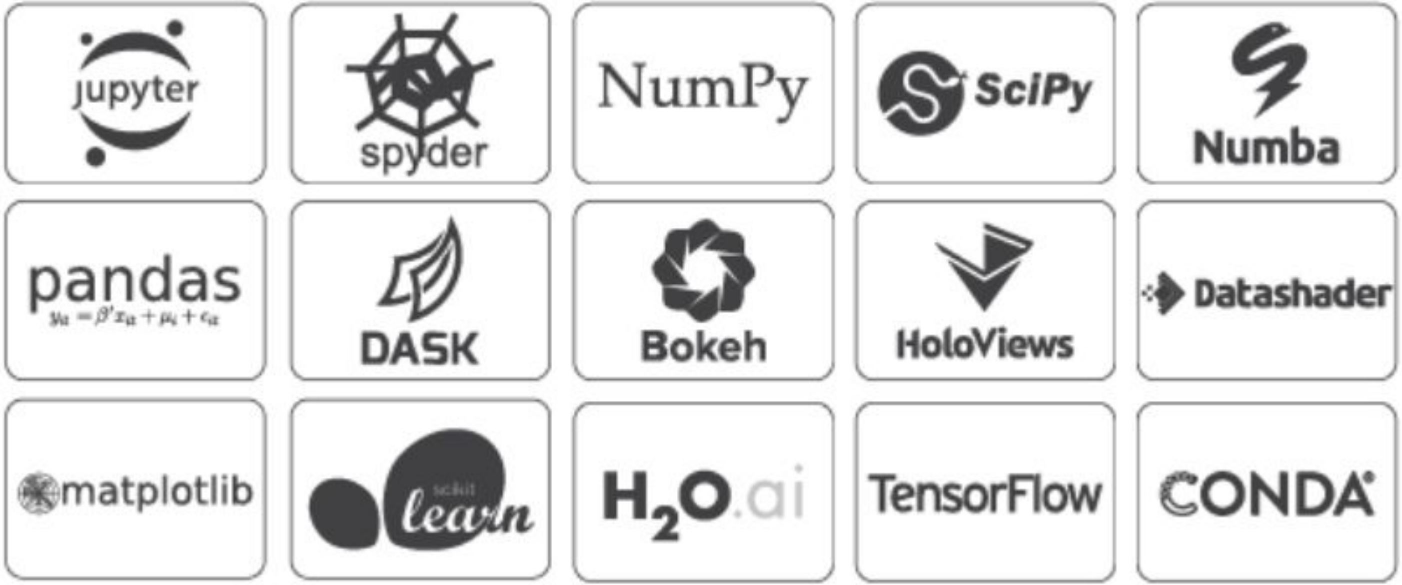
64-Bit Command Line Installer (435 MB)

Do not install Python 2.7



```
Using TensorFlow backend.
DEPRECATION: Python 2.7 will reach the end of its life on January 1st, 2020
. Please upgrade your Python as Python 2.7 won't be maintained after that d
ate. A future version of pip will drop support for Python 2.7.
Requirement already satisfied: kaggle in /usr/local/lib/python2.7/site-pack
ages (1.5.6)
```

ANACONDA


jupyter
spyder
NumPy
SciPy
Numba
pandas
$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$
DASK
Bokeh
HoloViews
Datashader
matplotlib
scikit learn
H2O.ai
TensorFlow
CONDA

Conda

Package, dependency and environment management for any language—Python, R, Ruby, Lua, Scala, Java, JavaScript, C/ C++, FORTRAN, and more.

Conda as a package manager helps you find and install packages. If you need a package that requires a different version of Python, you do not need to switch to a different environment manager, because conda is also an environment manager. With just a few commands, you can set up a totally separate environment to run that different version of Python, while continuing to run your usual version of Python in your normal environment.

And a lot of other things….

scikit 1

Create a conda environment

```
#Create a new environment
conda create -n name_env python=3.7
conda activate name_env        #Enter your new environment
conda deactivate               #Exit all the environments
```

matplotlib

Is the best plotting library out there. Install it:

```
conda install -c conda-forge matplotlib
```

scikit-learn

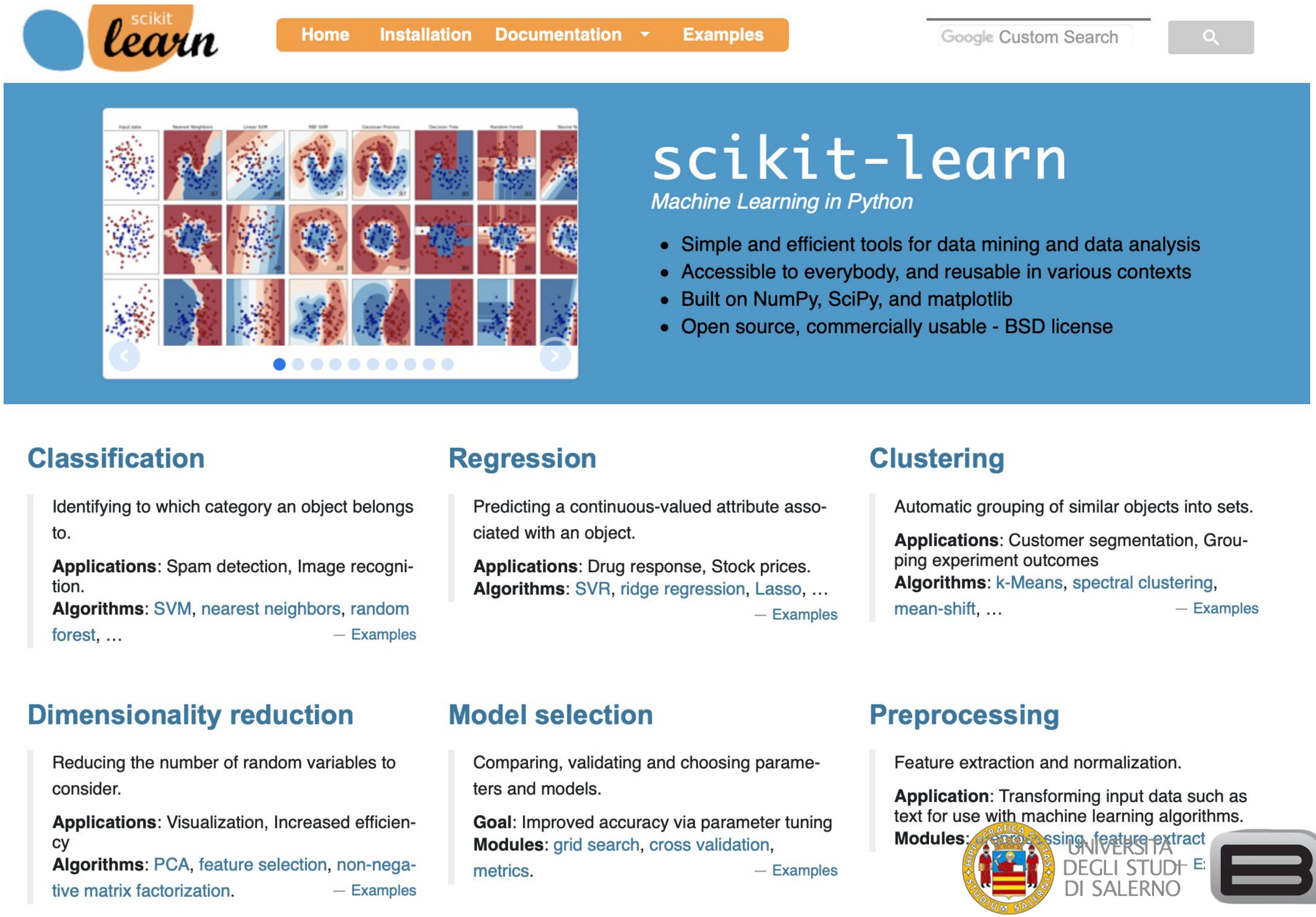
Is the best Machine Learning library in the world.

Is important that you are familiar with python and statistics

You can load dataset using **pandas**. Is Built on **Numpy**, **Scipy** & **Matplotlib**.There is a huge documentation on **scikit-learn.org**

Install it:

```
conda install -c anaconda scikit-learn
```



scikit-learn

Machine Learning in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying to which category an object belongs to.

Applications: Spam detection, Image recognition.
Algorithms: SVM, nearest neighbors, random forest, … — Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.
Algorithms: SVR, ridge regression, Lasso, … — Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, … — Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency
Algorithms: PCA, feature selection, non-negative matrix factorization. — Examples

Model selection

Comparing, validating and choosing parameters and models.

Goal: Improved accuracy via parameter tuning
Modules: grid search, cross validation, metrics. — Examples

Preprocessing

Feature extraction and normalization.

Application: Transforming input data such as text for use with machine learning algorithms.
Modules: preprocessing, feature extract

UNIVERSITÀ
DEGLI STUDI
DI SALERNO

BIP LAB

data manipulation

pandas

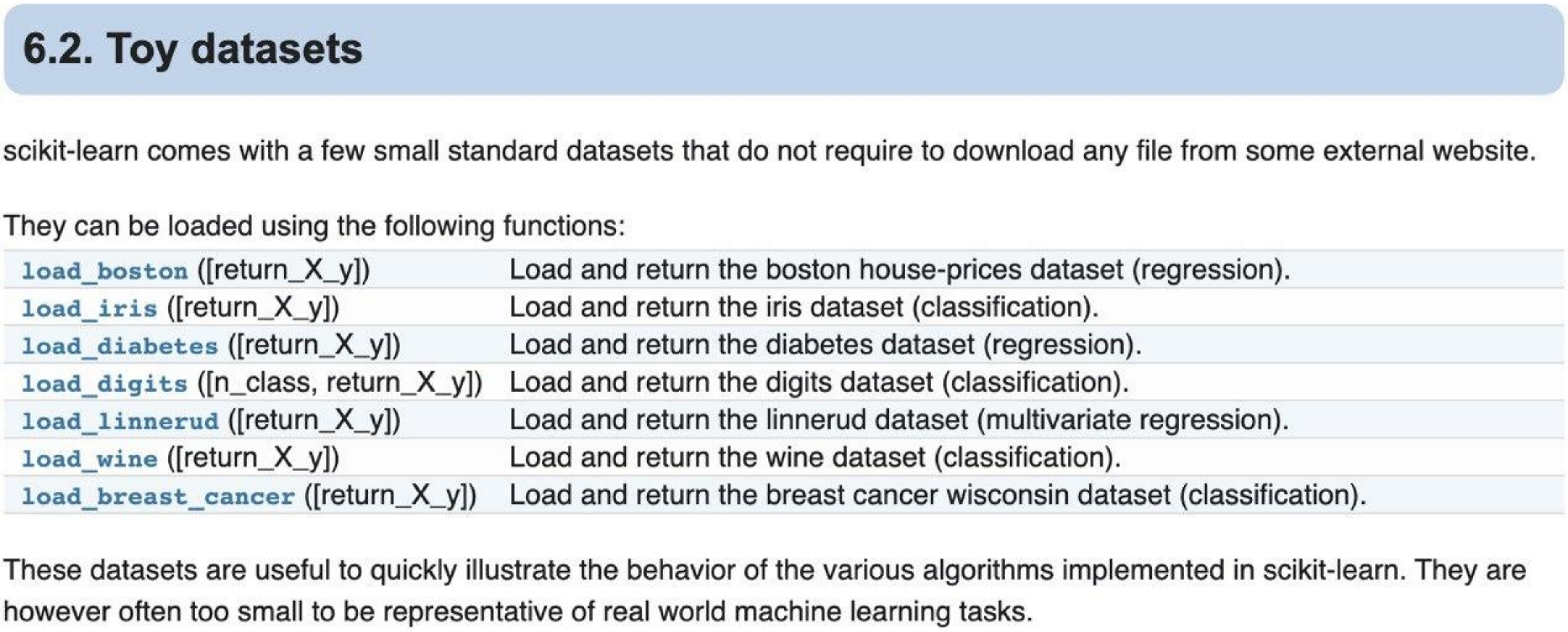
$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$

Pandas is used to manage data. It works with dataframes (strings of numerical data). Here we will see:

- how to load csv file in Pandas
- how to use prebuilt datasets, to help you testing scikit-learn algorithms (you'll never use this kind of dataset but they are useful indeed)

Install it:

conda install -c anaconda pandas



6.2. Toy datasets

scikit-learn comes with a few small standard datasets that do not require to download any file from some external website.

They can be loaded using the following functions:

`load_boston` ([return_X_y])	Load and return the boston house-prices dataset (regression).
`load_iris` ([return_X_y])	Load and return the iris dataset (classification).
`load_diabetes` ([return_X_y])	Load and return the diabetes dataset (regression).
`load_digits` ([n_class, return_X_y])	Load and return the digits dataset (classification).
`load_linnerud` ([return_X_y])	Load and return the linnerud dataset (multivariate regression).
`load_wine` ([return_X_y])	Load and return the wine dataset (classification).
`load_breast_cancer` ([return_X_y])	Load and return the breast cancer wisconsin dataset (classification).

These datasets are useful to quickly illustrate the behavior of the various algorithms implemented in scikit-learn. They are however often too small to be representative of real world machine learning tasks.

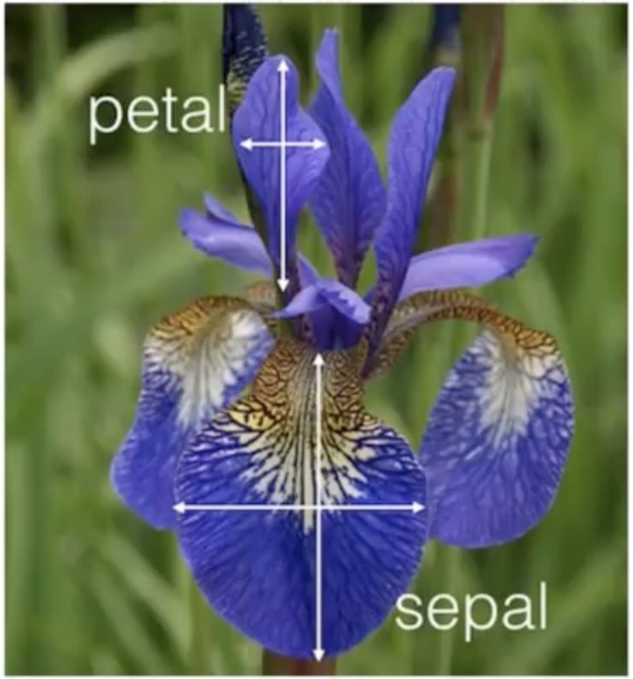
UNIVERSITÀ
DEGLI STUDI
DI SALERNO

IRIS Dataset

- is a famous example dataset

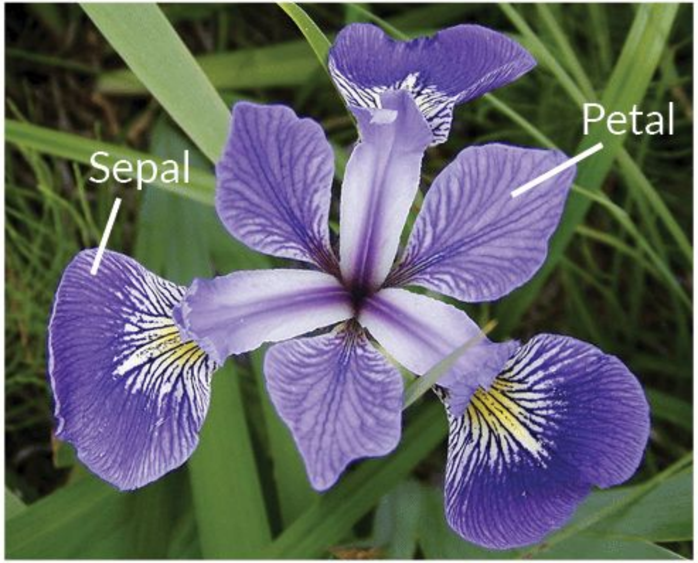
- the dataset consists of 50 samples from three species of Iris (**Setosa**, **Virginica** and **Versicolor**)

- Four features were measured from each samples (**sepals lenght**, **sepals width**, **petal lenght** and **petal width**) in centimeters.



UNIVERSITÀ DEGLI STUDI DI SALERNO

IRIS Dataset



Iris Versicolor

Iris Setosa

Iris Virginica

jupyter
php
C#
Spark
F#
C++