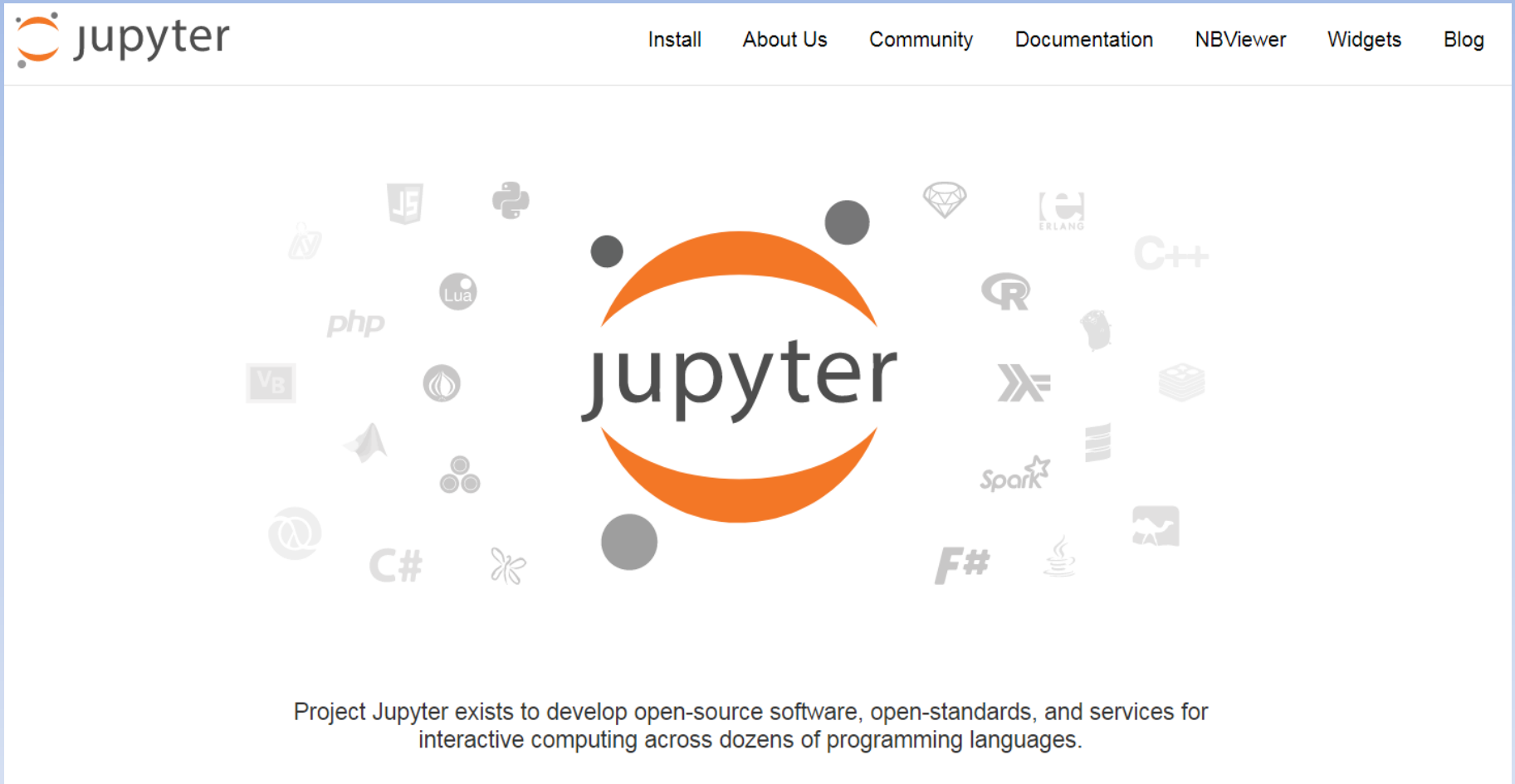
Machine Learning Tools

Python Stack --- FREE

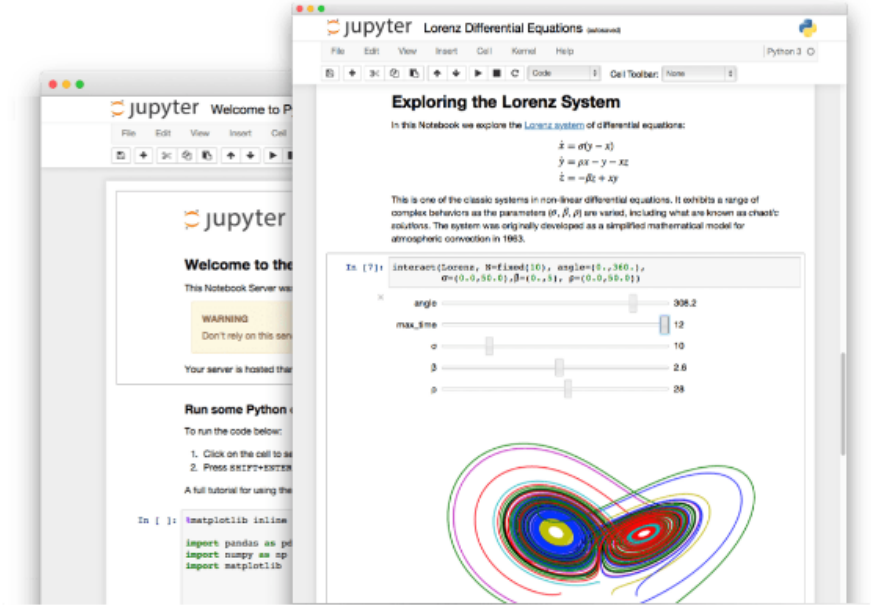
- Common set of tools for doing Machine Learning
- Several Common Modules:
  - Jupyter, NumPy, Pandas, Matplotlib, and Scikit-Learn

Jupyter Notebook



jupyter

Install About Us Community Documentation NBViewer Widgets Blog



The Jupyter Notebook

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

Language of choice

The Notebook has support for over 40 programming languages, including Python, R, Julia, and Scala.

Share notebooks

Notebooks can be shared with others using email, Dropbox, GitHub and the Jupyter Notebook Viewer.

Interactive output

Your code can produce rich, interactive output: HTML, images, videos, LaTeX, and custom MIME types.

Big data integration

Leverage big data tools, such as Apache Spark, from Python, R and Scala. Explore that same data with pandas, scikit-learn, ggplot2, TensorFlow.

Steering Council

The role of the Jupyter Steering Council is to ensure, through working with and serving the broader Jupyter community, the long-term well-being of the project, both technically and as a community. The Jupyter Steering Council currently consists of the following members (in alphabetical order).

Damian Avila
Continuum Analytics
@damianavila on GitHub

Matthias Bussonnier
UC Berkeley
@carreau on GitHub

Sylvain Corlay
QuantStack
@sylvaincorlay on GitHub

Jonathan Frederic
Cal Poly, San Luis Obispo
@jdfreder on GitHub

Brian Granger
Cal Poly, San Luis Obispo
@ellisonbg on GitHub

Jason Grout
Bloomberg
@jasongrout on GitHub

Jessica Hamrick
UC Berkeley
@jhamrick on GitHub

Paul Ivanov
Bloomberg
@ivanov on GitHub

Thomas Kluyver
University of Southampton
@takluyver on GitHub

Kyle Kelley
Netflix
@rgbkrk on GitHub

Peter Parente
MaxPoint
@parente on GitHub

Fernando Perez
UC Berkeley
@fperez on GitHub

Min Ragan-Kelley
Simula Research Lab
@minrk on GitHub

Steven Silvester
Continuum Analytics
@blink1073 on GitHub

Carol Willing
Cal Poly
@willingc on GitHub

everestso / handson-ml
forked from ageron/handson-ml

Unwatch 1 | Star 0 | Fork 2,439

Code | Pull requests 0 | Projects 0 | Wiki | Insights | Settings

Branch: master | handson-ml / 02_end_to_end_machine_learning_project.ipynb | Find file | Copy path

Kxrr Fix incorrect description in chapter 2 — 91b142c on Nov 24, 2017

4 contributors

1.29 MB | Download | History

Chapter 2 – End-to-end Machine Learning project

Welcome to Machine Learning Housing Corp.! Your task is to predict median house values in Californian districts, given a number of features from these districts.

This notebook contains all the sample code and solutions to the exercices in chapter 2.

Note: You may find little differences between the code outputs in the book and in these Jupyter notebooks: these slight differences are mostly due to the random nature of many training algorithms: although I have tried to make these notebooks' outputs as constant as possible, it is impossible to guarantee that they will produce the exact same output on every platform. Also, some data structures (such as dictionaries) do not preserve the item order. Finally, I fixed a few minor bugs (I added notes next to the concerned cells) which lead to slightly different results, without changing the ideas presented in the book.

Setup

First, let's make sure this notebook works well in both python 2 and 3, import a few common modules, ensure MatplotLib plots figures inline and prepare a function to save the figures:

In [1]:
```
# To support both python 2 and python 3
from __future__ import division, print_function, unicode_literals

# Common imports
import numpy as np
import os

# to make this notebook's output stable across runs
np.random.seed(42)

# To plot pretty figures
%matplotlib inline
import matplotlib
import matplotlib.pyplot as plt
plt.rcParams['axes.labelsize'] = 14
plt.rcParams['xtick.labelsize'] = 12
plt.rcParams['ytick.labelsize'] = 12

# Where to save the figures
PROJECT_ROOT_DIR = "."
CHAPTER_ID = "end_to_end_project"
IMAGES_PATH = os.path.join(PROJECT_ROOT_DIR, "images", CHAPTER_ID)

def save_fig(fig_id, tight_layout=True, fig_extension="png", resolution=300):
    path = os.path.join(IMAGES_PATH, fig_id + "." + fig_extension)
    print("Saving figure", fig_id)
    if tight_layout:
        plt.tight_layout()
    plt.savefig(path, format=fig_extension, dpi=resolution)
```

Get the data

- Examples in Notebooks on Github

Course Example w/ Colab bit.ly/c264s20dtree1

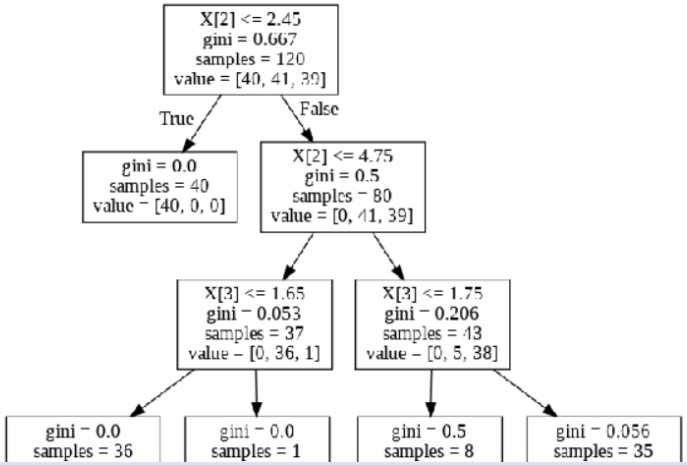
- http://bit.ly/s20dtree1

DecisionTree.ipynb

File Edit View Insert Runtime Tools Help Last edited on January 30

+ Code + Text

```
[ ] 12 tree.export_graphviz(classifier, out_file=dot_data)
    13 graph = pydotplus.graph_from_dot_file("dtree2.dot")
    14 graph.write_png('test.png')
    15 img=mpimg.imread('test.png')
    16 fig = plt.figure(figsize=(10, 10))
    17 plt.axis("off")
    18 plt.imshow(img, cmap = plt.cm.binary,
    19            interpolation="nearest")
    20 plt.show()
```



Colab Notebooks

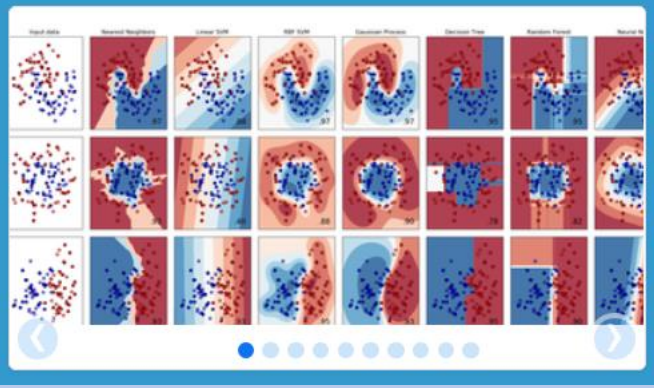
- http://bit.ly/c164s20colabs

scikit learn

Home Installation Documentation Examples

Google Custom Search Search

Fork me on GitHub



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 extraction. — Examples

SciPy.org Sponsored By ENTHOUGHT

Install | Getting Started | Documentation | Report Bugs | Blogs

SciPy (pronounced "Sigh Pie") is a Python-based ecosystem of open-source software for mathematics, science, and engineering. In particular, these are some of the core packages:

NumPy
Base N-dimensional array package

SciPy library
Fundamental library for scientific computing

Matplotlib
Comprehensive 2D Plotting

IPython
Enhanced Interactive Console

Sympy
Symbolic mathematics

pandas
Data structures & analysis

NumPy

Scipy.org

NumPy

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

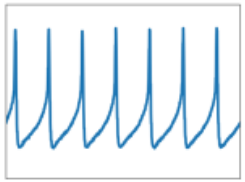
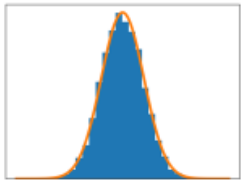
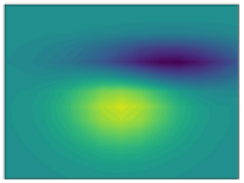
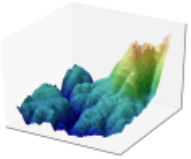
NumPy is licensed under the BSD license, enabling reuse with few restrictions.

matplotlib

home | examples | tutorials | pyplot | docs »

Introduction

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shell, the jupyter notebook, web application servers, and four graphical user interface toolkits.

Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, errorcharts, scatterplots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the `pyplot` module provides a MATLAB-like interface, particularly when combined with `IPython`. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

pandas

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

Python Data Analysis Library

pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.

pandas is a NumFOCUS sponsored project. This will help ensure the success of development of *pandas* as a world-class open-source project, and makes it possible to donate to the project.

A Fiscally Sponsored Project of

NUMFOCUS

OPEN CODE = BETTER SCIENCE

VERSIONS

Release
0.22.0 - December 2017
download // docs // pdf

Development
0.23.0 - 2018
github // docs


Anaconda Cloud
Documentation
Blog
Contact
ANACONDA.
What is Anaconda?
Products
Support
Community
About
Resources
Download
ANACONDA.
The Most Popular Python Data Science Platform
4.5M+
Users
1,000+
Data Science Packages
150+
Enterprise Customers


ANACONDA.
What is Anaconda?
Products
Support
Community
About
Resources
Download
Download Anaconda Distribution
Version 5.0.1 | Release Date: October 25, 2017
Download For:
High-Performance Distribution
Easily install 1,000+ data science packages
Package Management
Manage packages, dependencies and environments with conda
Portal to Data Science
Uncover insights in your data and create interactive visualizations
Windows
macOS
Linux

Python Stack Example

http://bit.ly/c164s20ml1

- Get Data

s20ml1.ipynb

File Edit View Insert Runtime Tools Help Last edited on January 28

Comment Share

+ Code + Text

Connect Editing

```
[ ]  plt.savefig(path, format=fig_extension, dpi=resolution)
```

```
[ ]  csv_path = os.path.join("/content/gdrive/My Drive/csuf/S20/Resources", "Datasets","Housing", "housing.csv")
     print(csv_path)
```

```
/content/gdrive/My Drive/csuf/S20/Resources/Datasets/Housing/housing.csv
```

```
[ ]  housing = pd.read_csv(csv_path)
     housing.head()
```

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value	ocean_proximity
0	-122.23	37.88	41.0	880.0	129.0	322.0	126.0	8.3252	452600.0	NEAR BAY
1	-122.22	37.86	21.0	7099.0	1106.0	2401.0	1138.0	8.3014	358500.0	NEAR BAY
2	-122.24	37.85	52.0	1467.0	190.0	496.0	177.0	7.2574	352100.0	NEAR BAY
3	-122.25	37.85	52.0	1274.0	235.0	558.0	219.0	5.6431	341300.0	NEAR BAY
4	-122.25	37.85	52.0	1627.0	280.0	565.0	259.0	3.8462	342200.0	NEAR BAY

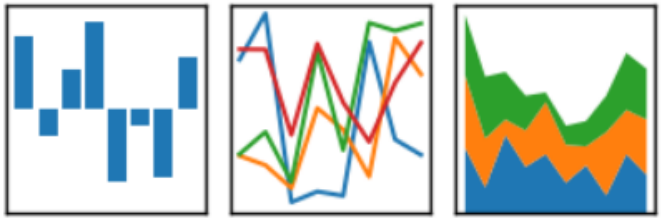
```
[ ]  housing.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 10 columns):
longitude             20640 non-null float64
latitude              20640 non-null float64
housing_median_age    20640 non-null float64
total_rooms           20640 non-null float64
total_bedrooms        20433 non-null float64
```

Data w/ Pandas

pandas

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



home // about // get pandas // documentation // community // talks // donate

Python Data Analysis Library

pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.

pandas is a NumFOCUS sponsored project. This will help ensure the success of development of *pandas* as a world-class open-source project, and makes it possible to donate to the project.

A Fiscally Sponsored Project of

NUMFOCUS

OPEN CODE = BETTER SCIENCE

VERSIONS

Release
0.22.0 - December 2017
download // docs // pdf

Development
0.23.0 - 2018
github // docs

Table Of Contents

IO Tools (Text, CSV, HDF5, ...)

The pandas I/O API is a set of top level `reader` functions accessed like `pd.read_csv()` that generally return a `pandas` object. The corresponding `writer` functions are object methods that are accessed like `df.to_csv()`

Format Type	Data Description	Reader	Writer
text	CSV	read_csv	to_csv
text	JSON	read_json	to_json
text	HTML	read_html	to_html
text	Local clipboard	read_clipboard	to_clipboard
binary	MS Excel	read_excel	to_excel
binary	HDF5 Format	read_hdf	to_hdf
binary	Feather Format	read_feather	to_feather
binary	Parquet Format	read_parquet	to_parquet
binary	Msgpack	read_msgpack	to_msgpack
binary	Stata	read_stata	to_stata
binary	SAS	read_sas	
binary	Python Pickle Format	read_pickle	to_pickle
SQL	SQL	read_sql	to_sql
SQL	Google Big Query	read_gbq	to_gbq

Here is an informal performance comparison for some of these IO methods.

> **Note:** For examples that use the `StringIO` class, make sure you import it according to your Python version, i.e. `from StringIO import StringIO` for Python 2 and `from io import StringIO` for Python 3.

CSV & Text files

The two workhorse functions for reading text files (a.k.a. flat files) are `read_csv()` and `read_table()`. They both use the same parsing code to intelligently convert tabular data into a DataFrame object. See the cookbook for some advanced strategies.

Some Pandas Basics

pandas 0.22.0 documentation »

Table Of Contents

Essential Basic Functionality

Here we discuss a lot of the essential functionality common to the pandas data structures. Here's how to create some of the objects used in the examples from the previous section:

```
In [1]: index = pd.date_range('1/1/2000', periods=8)

In [2]: s = pd.Series(np.random.randn(5), index=['a', 'b', 'c', 'd', 'e'])

In [3]: df = pd.DataFrame(np.random.randn(8, 3), index=index,
   ...:                   columns=['A', 'B', 'C'])
   ...:

In [4]: wp = pd.Panel(np.random.randn(2, 5, 4), items=['Item1', 'Item2'],
   ...:               major_axis=pd.date_range('1/1/2000', periods=5),
   ...:               minor_axis=['A', 'B', 'C', 'D'])
   ...:
```

Head and Tail

To view a small sample of a Series or DataFrame object, use the `head()` and `tail()` methods. The default number of elements to display is five, but you may pass a custom number.

```
In [5]: long_series = pd.Series(np.random.randn(1000))

In [6]: long_series.head()
Out[6]:
0    0.229453
1    0.304418
2    0.736135
3   -0.859631
4   -0.424100
dtype: float64
```

read_csv()
head()

s20ml1.ipynb

File Edit View Insert Runtime Tools Help Last edited on January 28

Comment Share

+ Code + Text

Connect Editing

```
[ ] 26     plt.savefig(path, format=fig_extension, dpi=resolution)
```

```
[ ] 1 csv_path = os.path.join("/content/gdrive/My Drive/csuf/S20/Resources", "Datasets","Housing", "housing.csv")
    2 print(csv_path)
```

```
/content/gdrive/My Drive/csuf/S20/Resources/Datasets/Housing/housing.csv
```

```
[ ] 1 housing = pd.read_csv(csv_path)
    2 housing.head()
```

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value	ocean_proximity
0	-122.23	37.88	41.0	880.0	129.0	322.0	126.0	8.3252	452600.0	NEAR BAY
1	-122.22	37.86	21.0	7099.0	1106.0	2401.0	1138.0	8.3014	358500.0	NEAR BAY
2	-122.24	37.85	52.0	1467.0	190.0	496.0	177.0	7.2574	352100.0	NEAR BAY
3	-122.25	37.85	52.0	1274.0	235.0	558.0	219.0	5.6431	341300.0	NEAR BAY
4	-122.25	37.85	52.0	1627.0	280.0	565.0	259.0	3.8462	342200.0	NEAR BAY

```
[ ] 1 housing.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 10 columns):
longitude             20640 non-null float64
latitude              20640 non-null float64
housing_median_age    20640 non-null float64
total_rooms           20640 non-null float64
total_bedrooms        20433 non-null float64
```

Do Demo

https://git-scm.com/

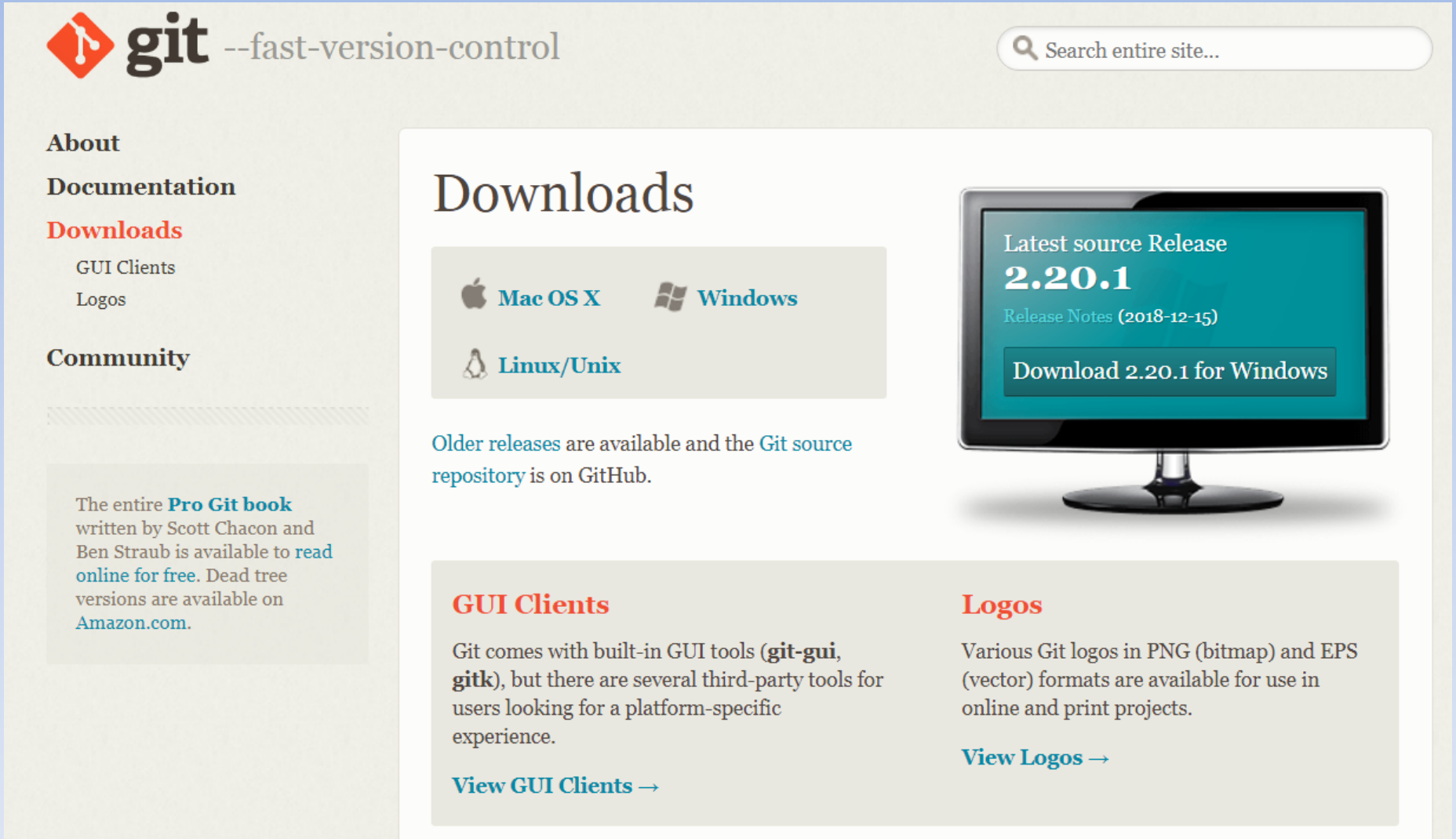
git --local-branching-on-the-cheap

Search entire site...

Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is easy to learn and has a tiny footprint with lightning fast performance. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like cheap local branching, convenient staging areas, and multiple workflows.

https://git-scm.com/downloads

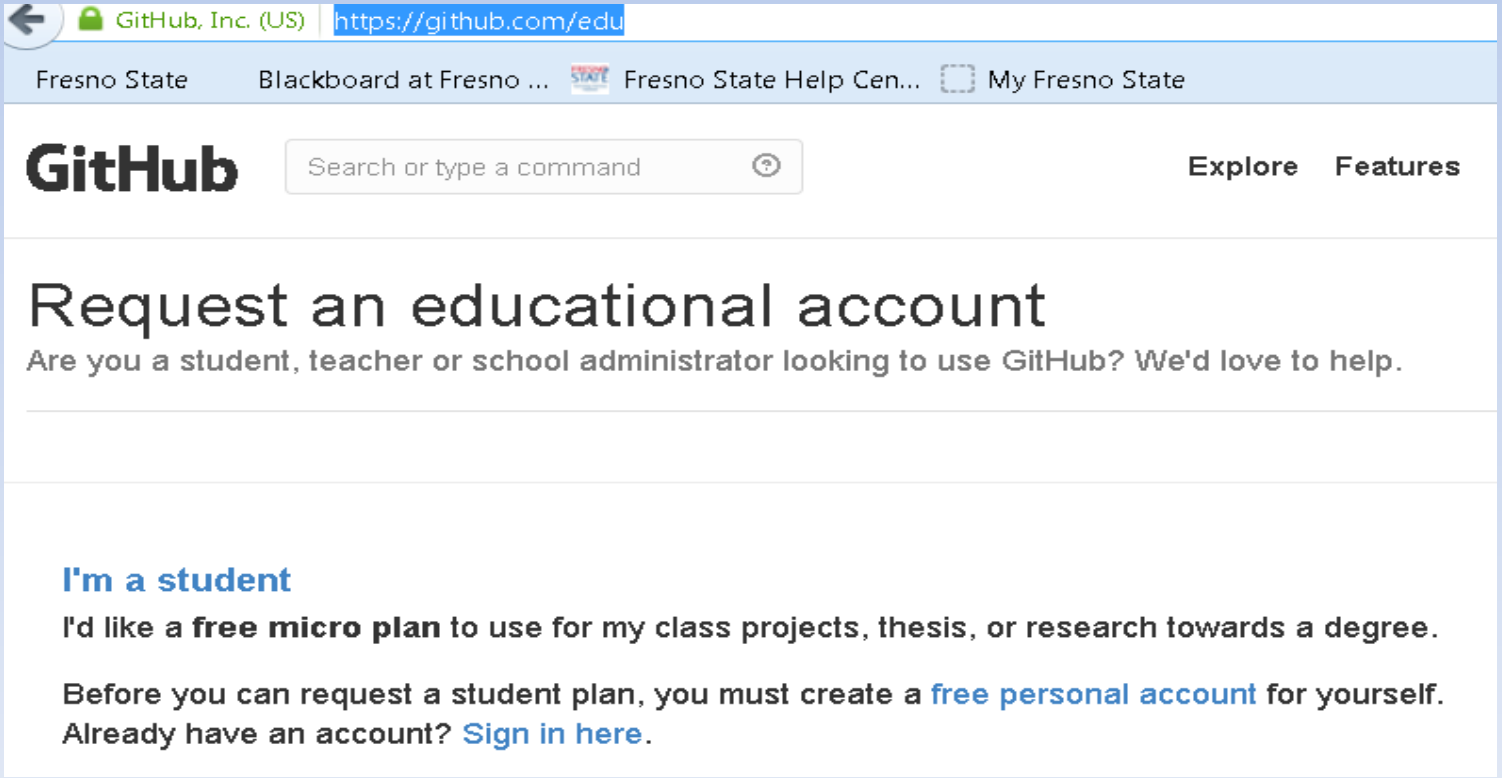


GitHub

- Popular Code Repository for Open-Source Projects
- Free Student Accounts
- In-Class: Code uploaded for class use.
- Student Projects: Interviews/Personal Development
- Great repository when looking for code ideas.

GitHub
github.com

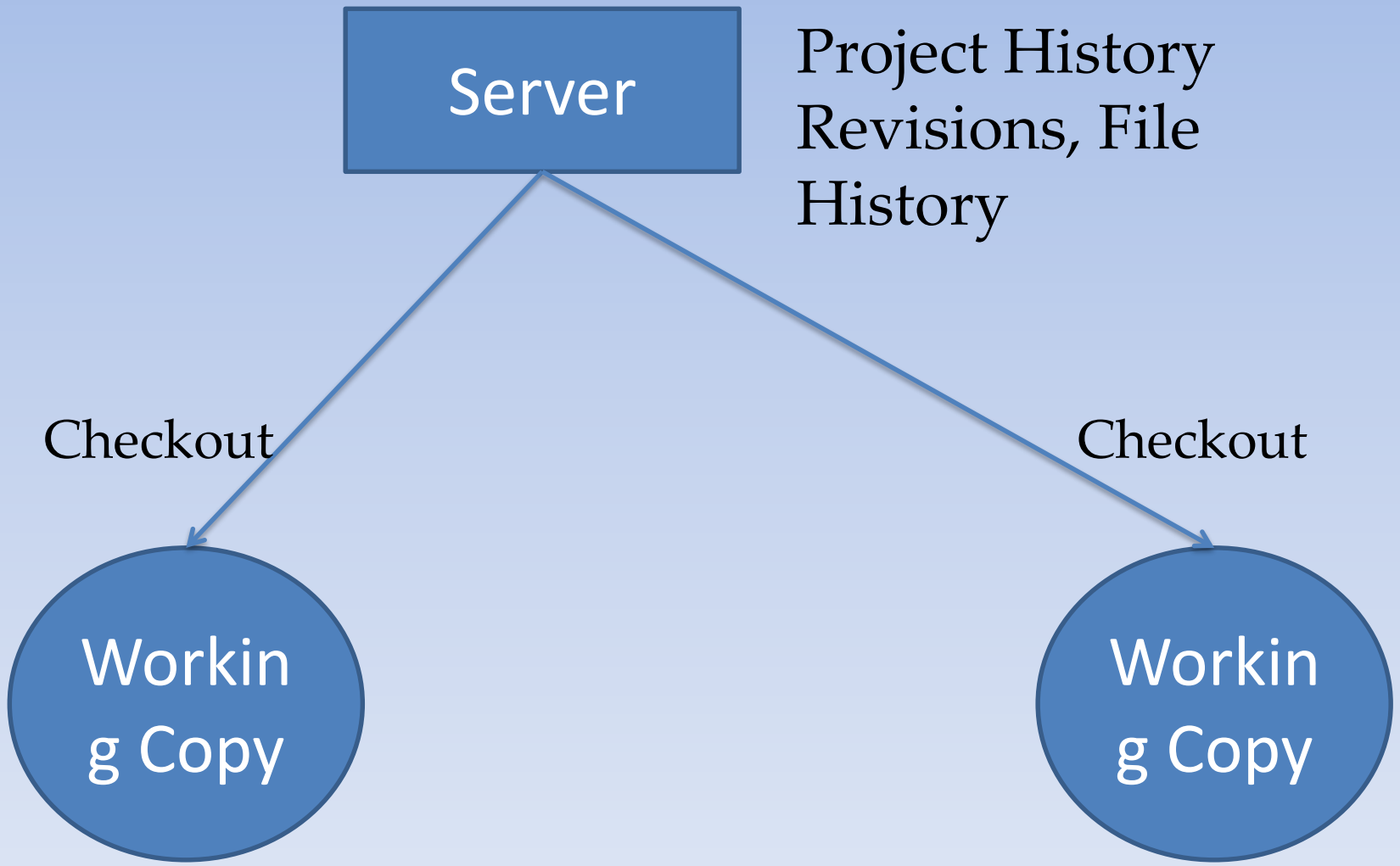
- Register for a basic account
- Consider requesting student account
  - https://Github.com/edu



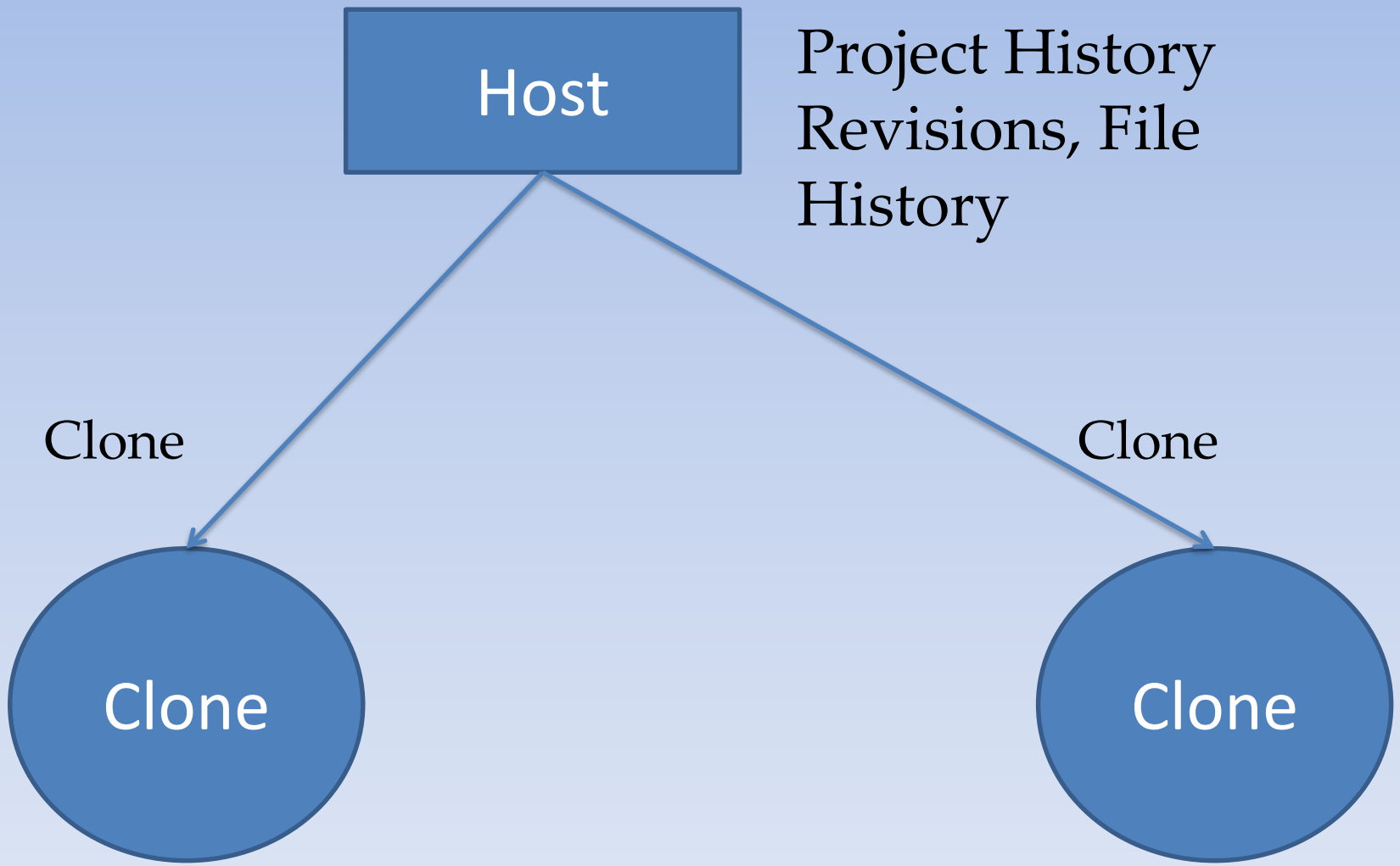
Managing Source Code

- Source code involves a lot of files
- Especially when working with a team, managing files is challenge.
- A tool that manages and tracks different versions of software or other content is called:
  - Version Control System (VCS)
  - Source Code Manager (SCM)
  - Revision Control System (RCS)

Traditional Source Control



Git


Host
Project History
Revisions, File
History
Clone
Clone
Clone
Clone

Git History

- Git was invented by Linus Torvalds to support development of Linux Kernel in April 2005.

Key Dates in VCS History

- Source Code Control System (SCCS) 1970s
- Revision Control System (RCS) 1980s
- CVS (Concurrent Version System): 1986
  - Very popular
- Subversion (SVN): 2001
  - Added better support for branches
- BitKeeper (May 2000) and Mercurial (April 2005) went in a different direction
  - Eliminated central repository
  - Each developer has a shareable copy of the project
  - Peer-To-Peer Model
- Git is derived from this BitKeeper/Mercurial approach.

Git Basic Concepts

- Repository
  - A database of all the information needed to retain and manage the revisions and history of a project.
  - A set of configuration value are kept within each repository.
  - Object Store and Index are two primary data structures for Repository
- Object Store
  - Contains the original data files and all the log messages, author information, dates.
  - Contains all info needed to rebuild any version or branch of the project.
- Index
  - Temporary dynamic file that allows stage changes before a commit.

Git Configuration

- git config --list
- git help git
- git help git-config
- git config --help

- git config --global user.name “John Doe100”
- git config --global user.email ‘JD@test.com’

Making Changes

- git add .
  - Adds all new files
- git add –u
  - Updates tracking for files that changed names or were deleted
- **git add –A**
  - Does both of the previous
- Git Add update the Index Repository Structure

Git Status

- git status
  - Show the working tree status
- After making changes to files GitHub for Windows will show uncommitted changes.
- Git Status will indicate changes are not staged.

Git Status

posh~git ~ csci164 [master]

```
Mode                LastWriteTime     Length Name
----                -------------     ------ ----
d----         7/16/2014   4:31 PM            Lectures
d----         7/18/2014  11:18 AM            Resources
-a---         6/20/2014   9:12 AM         34 README.md


D:\Documents\GitHub\csci164 [master +1 ~0 -3 !]> git status --help
Launching default browser to display HTML ...
D:\Documents\GitHub\csci164 [master +1 ~0 -3 !]> git status --long
# On branch master
# Changes not staged for commit:
#   (use "git add/rm <file>..." to update what will be committed)
#   (use "git checkout -- <file>..." to discard changes in working directory)
#
#       deleted:    Resources/226/Coursera.FInal.docx
#       deleted:    Resources/226/xml.q1.docx
#       deleted:    Resources/226/xml.q2.docx
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#       Resources/226/Resources/
no changes added to commit (use "git add" and/or "git commit -a")
D:\Documents\GitHub\csci164 [master +1 ~0 -3 !]>
```

Git Commit

- git commit –m “commit message”
  - Will record changes to repository.
- git log
  - Shows all prior commits
- git push
  - Pushes changes to remote repository.

Clone In Repository

- Simplest is to open git shell and :

git clone https://github.com/ageron/handson-ml

- New git repository will be in subdirectory: handson-ml