

Tutorial tools

National Cheng Kung
University

IKM Lab

Outline

- Conda
- Github tutorial repositories
- Jupyter

Conda



- Conda
 - Open source package and environment management system for any language
- Install Miniconda
 - Links : <https://conda.io/miniconda.html>
 - Please download the Python3.6 version

Conda

- Install Miniconda

- Windows

- Download the exe installer and install it

- Linux

- Download the bash installer

- run “bash Miniconda3-latest-Linux-x86_64.sh” in terminal

- Mac OS X

- Download the bash installer

- run “bash Miniconda3-latest-MacOSX-x86_64.sh” in terminal

	 Windows	 Mac OS X	 Linux
Python 3.6	64-bit (exe installer) 32-bit (exe installer)	64-bit (bash installer)	64-bit (bash installer) 32-bit (bash installer)
Python 2.7	64-bit (exe installer) 32-bit (exe installer)	64-bit (bash installer)	64-bit (bash installer) 32-bit (bash installer)

Conda

- Install Miniconda
 - Check installation
 - open a terminal and run “conda list”
 - if you see the following message, the installation is completed.

```
kjes89011@kjes89011-System-Product-Name:~/miniconda3/bin$ conda list
# packages in environment at /home/kjes89011/miniconda3:
#
# Name                                Version                                Build      Channel
asn1crypto                            0.24.0                                py36_0
ca-certificates                       2018.03.07                            0
certifi                               2018.4.16                             py36_0
cffi                                   1.11.5                                py36h9745a5d_0
```

Conda

- Install necessary python library in tutorial
 - Run the following commands in terminal
 - matplotlib :
 - “conda install matplotlib”
 - scikit-learn :
 - “conda install scikit-learn”
 - pandas :
 - “conda install pandas”
 - keras :
 - “conda install -c conda-forge keras”

Github tutorial

- Download and unzip tutorial repositories

The screenshot shows a GitHub repository page for 'Logistic Regression' by user 'timniven'. The repository contains several files and folders, including '.ipynb_checkpoints', 'img', 'qa', 'README.md', 'SAheart.data.csv', 'logistic_regression_tutorial.ipynb', and 'logistic_regression_tutorial_answers.ipynb'. A modal window is open, showing options to clone the repository using HTTPS or SSH. The 'Clone or download' button is highlighted with a purple box. The 'Download ZIP' button is also highlighted with a purple box. The 'Open in Desktop' button is visible. The repository description at the bottom states: 'Accompanying slides here: https://drive.google.com/open?id=1ftjCtYTr6iwpicKbAasRsZ2qV4n_lqGZ4-o9ujL_8AY'. The URL is highlighted with a purple box.

Branch: publish ▾ New pull request

Create new file Upload files Find file Clone or download ▾

timniven added answers and blank

.ipynb_checkpoints	added answers and blank	
img	add images	
qa	moved August materials into main folder for August	
README.md	moved August materials into main folder for August	
SAheart.data.csv	add dataset, update images load method	29 days ago
logistic_regression_tutorial.ipynb	added answers and blank	3 minutes ago
logistic_regression_tutorial_answers.ipynb	added answers and blank	3 minutes ago

Clone with HTTPS ⓘ Use SSH

Use Git or checkout with SVN using the web URL.

<https://github.com/IKMLab/Logistic-Regre>

Open in Desktop Download ZIP

README.md

Logistic Regression

Accompanying slides here: https://drive.google.com/open?id=1ftjCtYTr6iwpicKbAasRsZ2qV4n_lqGZ4-o9ujL_8AY

Github tutorial

- Download and unzip tutorial repositories
 - Linear and Non Linear regression :
<https://github.com/IKMLab/Linear-Regression-Tutorial>
 - Logistic regression :
<https://github.com/IKMLab/Logistic-Regression-Tutorial>
 - Decision tree and Random Forest :
<https://github.com/IKMLab/decision-tree-and-random-forest-tutorial>

Jupyter

- A IDE on web which allow us to develop the program.
- Install jupyter
 - run “conda install jupyter” in terminal
 - if the program ask you to update package
 - The answer is “yes!”

```
The following packages will be UPDATED:
```

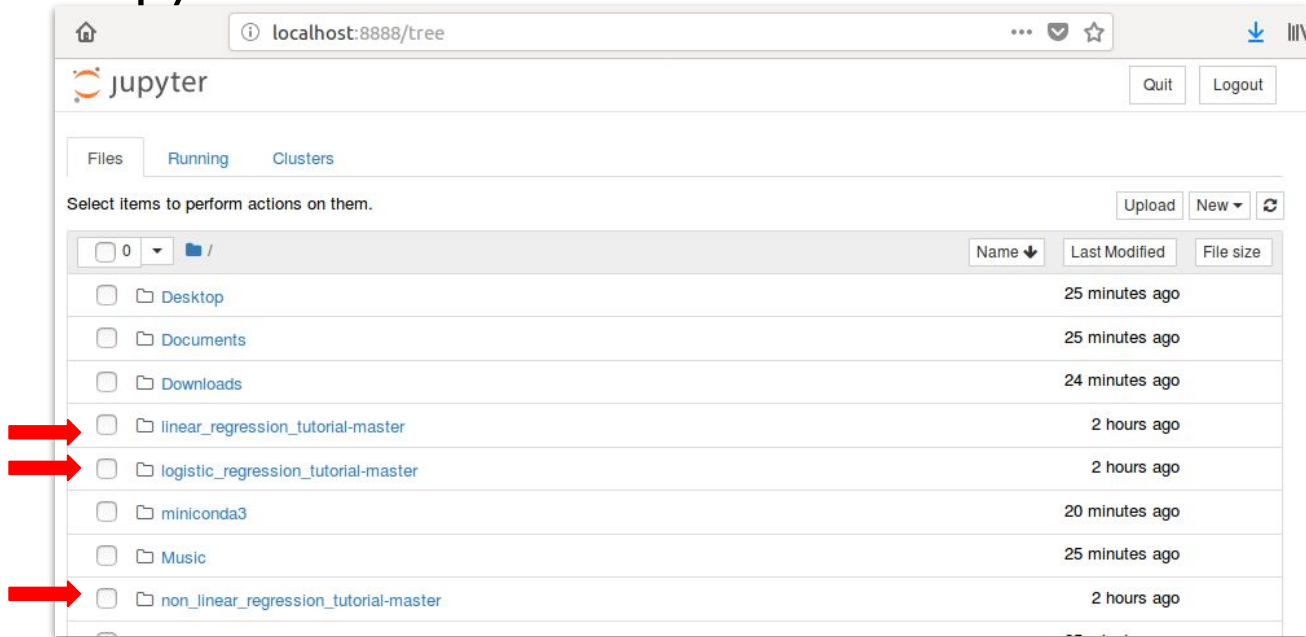
```
asn1crypto:      0.24.0-py36_0      --> 0.24.0-py37_0
certifi:          2018.4.16-py36_0    --> 2018.4.16-py37_0
cffi:             1.11.5-py36h9745a5d_0 --> 1.11.5-py37h9745a5d_0
chardet:          3.0.4-py36h0f667ec_1 --> 3.0.4-py37_1
```

```
Proceed ([y]/n)? y
```



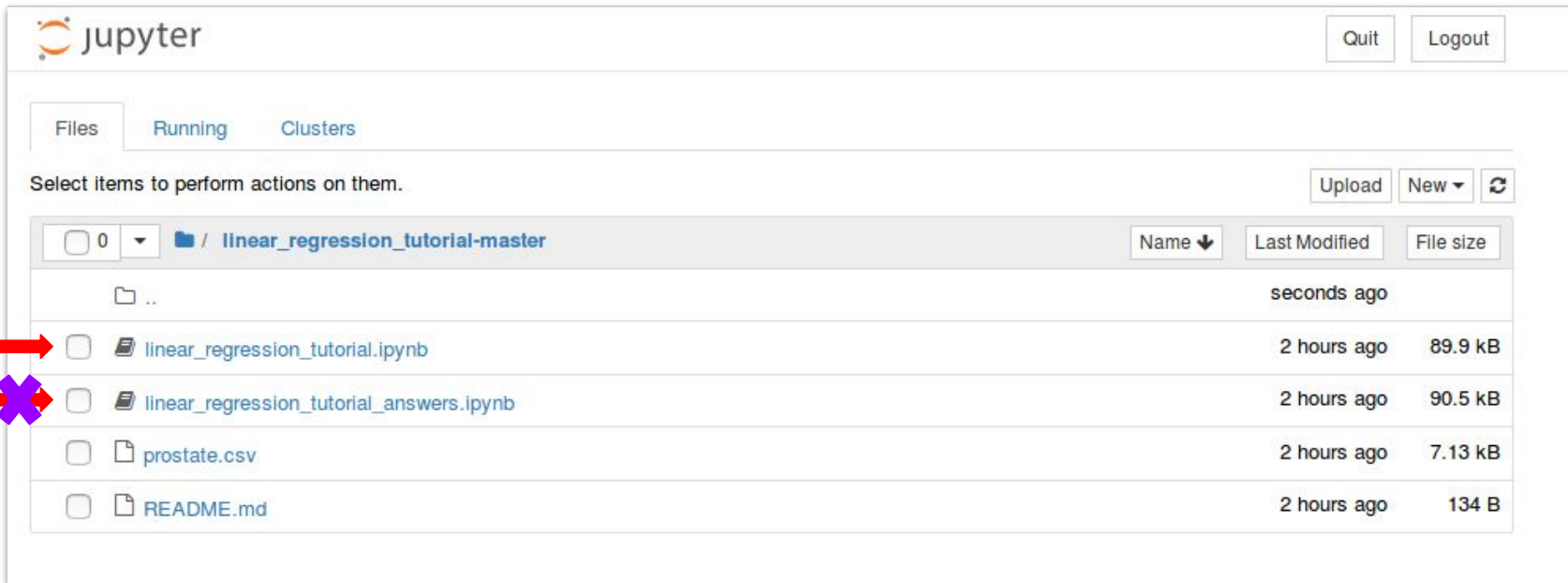
Jupyter

- Start jupyter
 - run “ipython notebook” in terminal



Jupyter

- Choose the “XXX_tutorial.ipynb” file to start !
 - Don't chose the “XXX_tutorial_answers.ipynb”



The screenshot shows the Jupyter web interface with the 'Files' tab selected. The current directory is 'linear_regression_tutorial-master'. The file list includes:


	Name	Last Modified	File size
<input type="checkbox"/>	..	seconds ago	
<input type="checkbox"/>	linear_regression_tutorial.ipynb	2 hours ago	89.9 kB
<input type="checkbox"/>	linear_regression_tutorial_answers.ipynb	2 hours ago	90.5 kB
<input type="checkbox"/>	prostate.csv	2 hours ago	7.13 kB
<input type="checkbox"/>	README.md	2 hours ago	134 B

A red arrow points to the checkbox for 'linear_regression_tutorial.ipynb', and a purple X is placed over the checkbox for 'linear_regression_tutorial_answers.ipynb'.

Jupyter

- Type your code in “cell.”

Linear Regression Tutorial 📖



```
In [1]: 1 # packages we will be using
2 import matplotlib.pyplot as plt
3 from sklearn import linear_model, metrics, model_selection
4 import numpy as np
5 import pandas as pd
```


What is Linear Regression?

Finding a straight line of best fit through the data. This works well when the true underlying function is linear.

Example

We use features \mathbf{x} to predict a "response" y . For example we might want to regress `num_hours_studied` onto `exam_score` - in other words we predict exam score from number of hours studied.

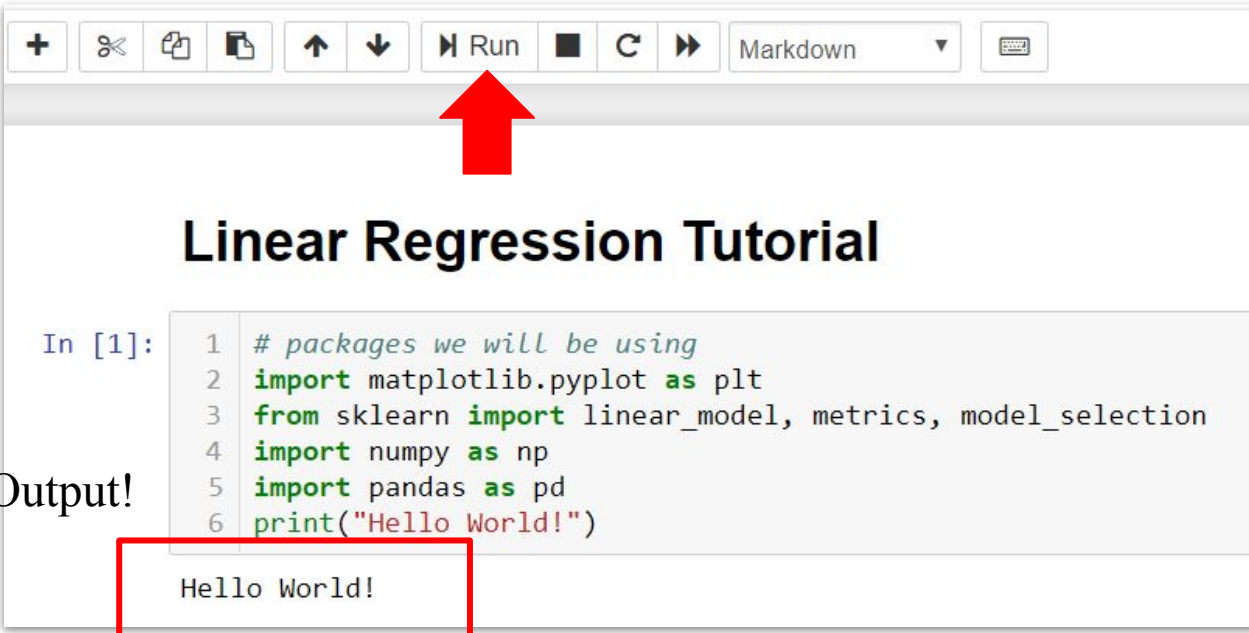
Let's generate some example data for this case and examine the relationship between \mathbf{x} and y .



```
In [2]: 1 num_hours_studied = np.array([1, 3, 3, 4, 5, 6, 7, 7, 8, 8, 10])
2 exam_score = np.array([18, 26, 31, 40, 55, 62, 71, 70, 75, 85, 97])
3 plt.scatter(num_hours_studied, exam_score)
4 plt.xlabel('num_hours_studied')
5 plt.ylabel('exam_score')
6 plt.show()
```

Jupyter

- Run your code in current cell



The image shows a Jupyter Notebook interface. At the top is a toolbar with various icons. A red arrow points to the 'Run' button, which is represented by a play icon. Below the toolbar, the notebook content is displayed. It starts with a large heading 'Linear Regression Tutorial'. Below this is a code cell labeled 'In [1]:'. The code cell contains six lines of Python code: a comment, and imports for matplotlib.pyplot, sklearn, numpy, and pandas, followed by a print statement. Below the code cell, the output 'Hello World!' is shown, which is enclosed in a red rectangular box. To the left of this box, the word 'Output!' is written.

Linear Regression Tutorial

In [1]:

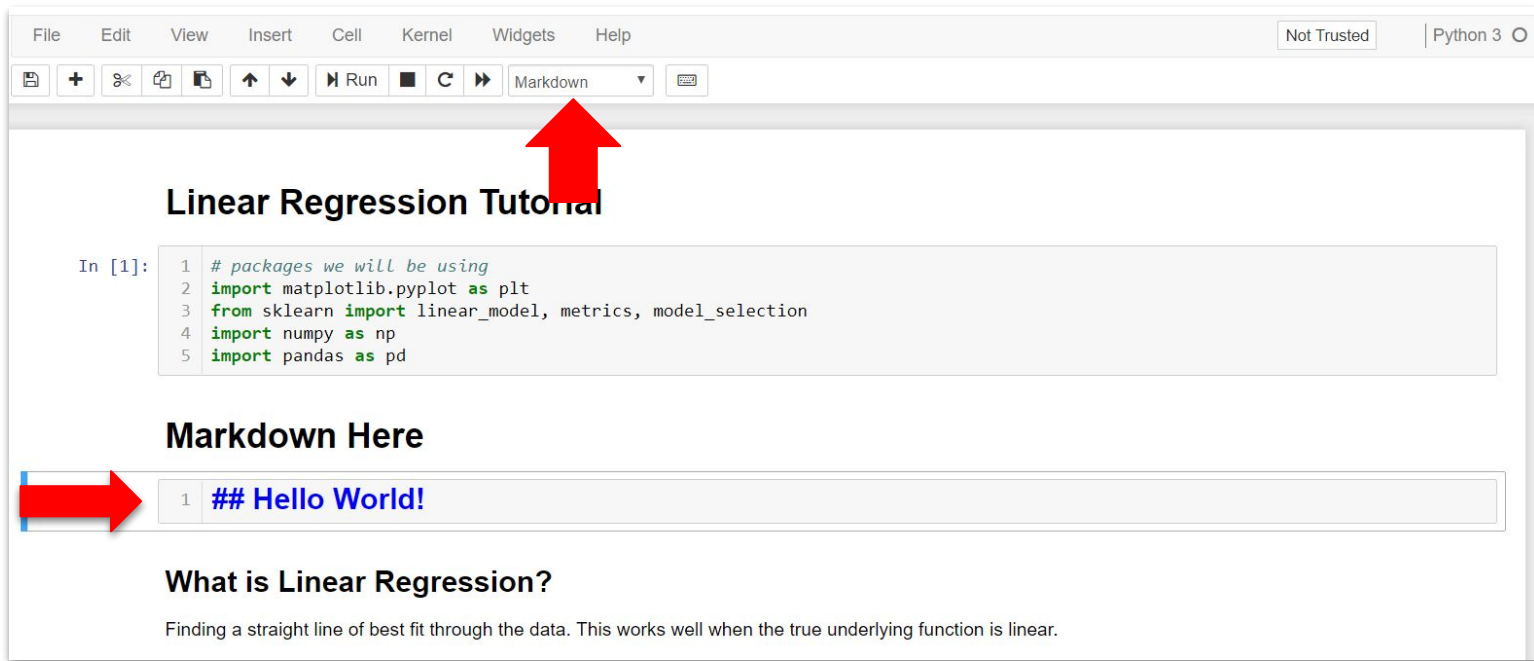
```
1 # packages we will be using
2 import matplotlib.pyplot as plt
3 from sklearn import linear_model, metrics, model_selection
4 import numpy as np
5 import pandas as pd
6 print("Hello World!")
```

Output!

Hello World!

Jupyter

- You can also write Markdown syntax in the cell



Jupyter

- Run the Markdown cell to get the output

Linear Regression Tutorial

```
In [1]: 1 # packages we will be using
        2 import matplotlib.pyplot as plt
        3 from sklearn import linear_model, metrics, model_selection
        4 import numpy as np
        5 import pandas as pd
```

Markdown Here



Hello World!

What is Linear Regression?

Finding a straight line of best fit through the data. This works well when the true underlying function is linear.