

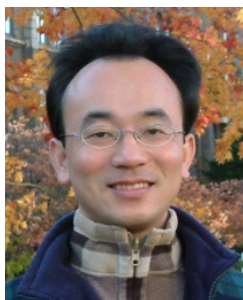
大數據分析 (Big Data Analysis)

Python 大數據分析基礎 (Foundations of Big Data Analysis in Python)

1091BDA03

MBA, IM, NTPU (M5127) (Fall 2020)

Wed 7, ,8, 9 (15:10-18:00) (B8F40)



Min-Yuh Day

戴敏育

Associate Professor

副教授

Institute of Information Management, National Taipei University

國立臺北大學 資訊管理研究所

<https://web.ntpu.edu.tw/~myday>

2020-09-30



課程大綱 (Syllabus)

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
1	2020/09/16	大數據分析介紹 (Introduction to Big Data Analysis)
2	2020/09/23	AI人工智慧與大數據分析 (AI and Big Data Analysis)
3	2020/09/30	Python 大數據分析基礎 (Foundations of Big Data Analysis in Python)
4	2020/10/07	數位沙盒第一堂課：數位沙盒服務平台簡介 (Digital Sandbox Lesson 1: Introduction to FintechSpace Digital Sandbox)
5	2020/10/14	數位沙盒第二堂課：工程師操作說明與實作教學 (Digital Sandbox Lesson 2: Hands-on Practices)
6	2020/10/21	Python Pandas 大數據量化分析 (Quantitative Big Data Analysis with Pandas in Python)

課程大綱 (Syllabus)

- | 週次 (Week) | 日期 (Date) | 內容 (Subject/Topics) |
|-----------|------------|--|
| 7 | 2020/10/28 | 數位沙盒第三堂課：學生小組討論實作與成果發表
(Digital Sandbox Lesson 3: Learning Teams
Hands-on Project Discussion and Project Presentation) |
| 8 | 2020/11/04 | Python Scikit-Learn 機器學習 I
(Machine Learning with Scikit-Learn In Python I) |
| 9 | 2020/11/11 | 期中報告 (Midterm Project Report) |
| 10 | 2020/11/18 | Python Scikit-Learn 機器學習 II
(Machine Learning with Scikit-Learn In Python II) |
| 11 | 2020/11/25 | TensorFlow 深度學習金融大數據分析 I
(Deep Learning for Finance Big Data Analysis with TensorFlow I) |
| 12 | 2020/12/02 | 大數據分析個案研究
(Case Study on Big Data Analysis) |

課程大綱 (Syllabus)

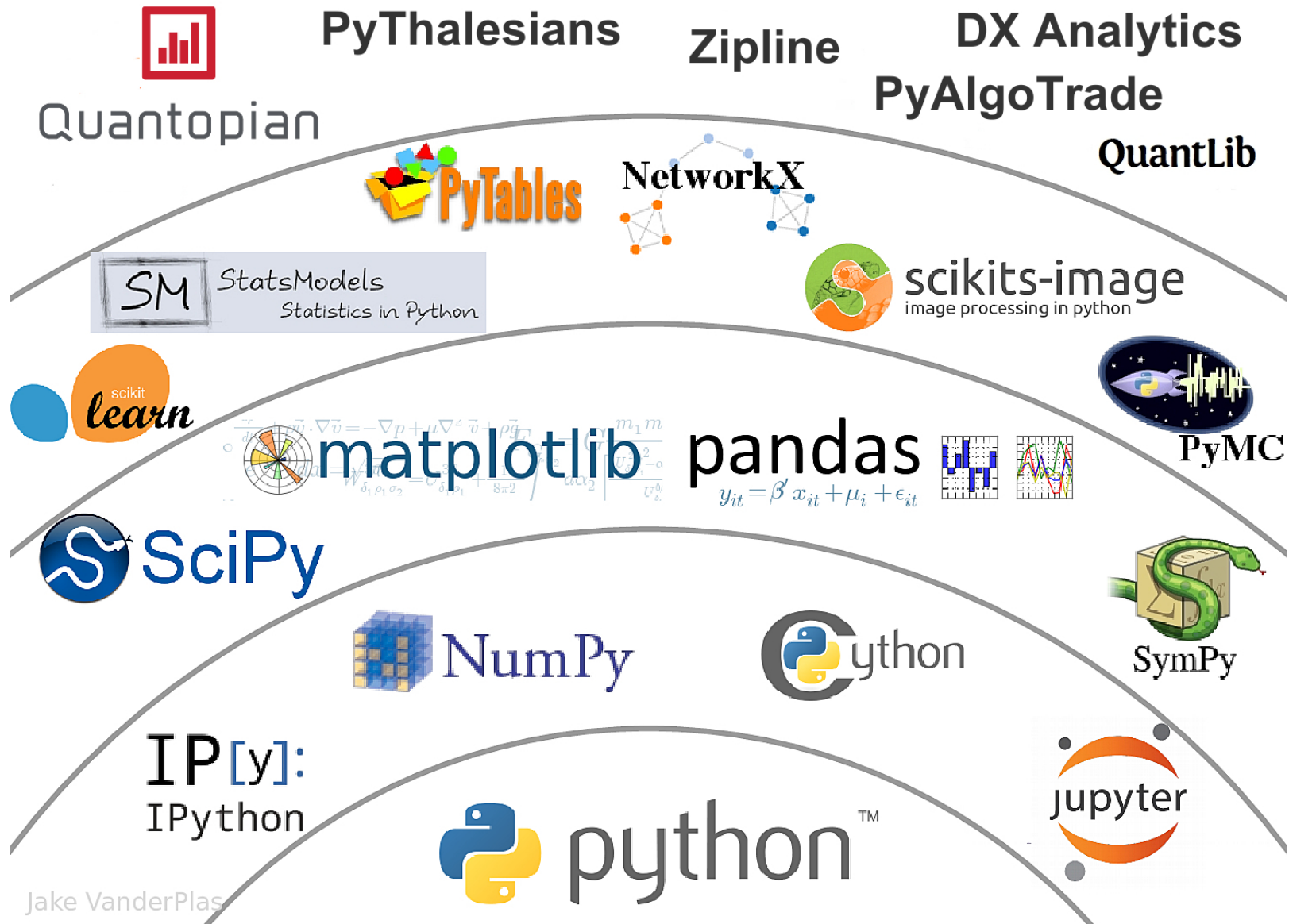
- | 週次 (Week) | 日期 (Date) | 內容 (Subject/Topics) |
|-----------|------------|---|
| 13 | 2020/12/09 | TensorFlow 深度學習金融大數據分析 II
(Deep Learning for Finance Big Data Analysis with TensorFlow II) |
| 14 | 2020/12/16 | TensorFlow 深度學習金融大數據分析 III
(Deep Learning for Finance Big Data Analysis with TensorFlow III) |
| 15 | 2020/12/23 | AI 機器人理財顧問
(Artificial Intelligence for Robo-Advisors) |
| 16 | 2020/12/30 | 金融科技智慧型交談機器人
(Conversational Commerce and Intelligent Chatbots for Fintech) |
| 17 | 2021/01/06 | 期末報告 I (Final Project Report I) |
| 18 | 2021/01/13 | 期末報告 II (Final Project Report I) |

Foundations of Big Data Analysis in Python

Outline

- **Foundations of Big Data Analysis in Python**
 - **Python**
 - Programming language
 - **Numpy**
 - Scientific computing

The Quant Finance PyData Stack



Jake VanderPlas

Source: http://nbviewer.jupyter.org/format/slides/github/quantopian/pyfolio/blob/master/pyfolio/examples/overview_slides.ipynb/#5



Python

Python is an
interpreted,
object-oriented,
high-level
programming language
with
dynamic semantics.

Google Colab

The screenshot shows the Google Colaboratory web interface. At the top, the browser address bar displays the URL <https://colab.research.google.com/notebooks/welcome.ipynb>. The main header includes the 'Hello, Colaboratory' logo and a menu with options like File, Edit, View, Insert, Runtime, Tools, and Help. On the right, there are 'SHARE' and 'CONNECT' buttons, along with an 'EDITING' mode indicator.

The left sidebar contains a 'Table of contents' with sections for 'Getting Started', 'Highlighted Features', 'TensorFlow execution', 'GitHub', 'Visualization', 'Forms', 'Examples', and 'Local runtime support'. A '+ SECTION' button is visible at the bottom of the sidebar.

The main content area features a large 'Welcome to Colaboratory!' message with the Colab logo and a brief description: 'Colaboratory is a free Jupyter notebook environment that requires no setup and runs entirely in the cloud. See our [FAQ](#) for more info.' Below this is a 'Getting Started' section with a list of links: 'Overview of Colaboratory', 'Loading and saving data: Local files, Drive, Sheets, Google Cloud Storage', 'Importing libraries and installing dependencies', 'Using Google Cloud BigQuery', 'Forms, Charts, Markdown, & Widgets', 'TensorFlow with GPU', and 'Machine Learning Crash Course: Intro to Pandas & First Steps with TensorFlow'.

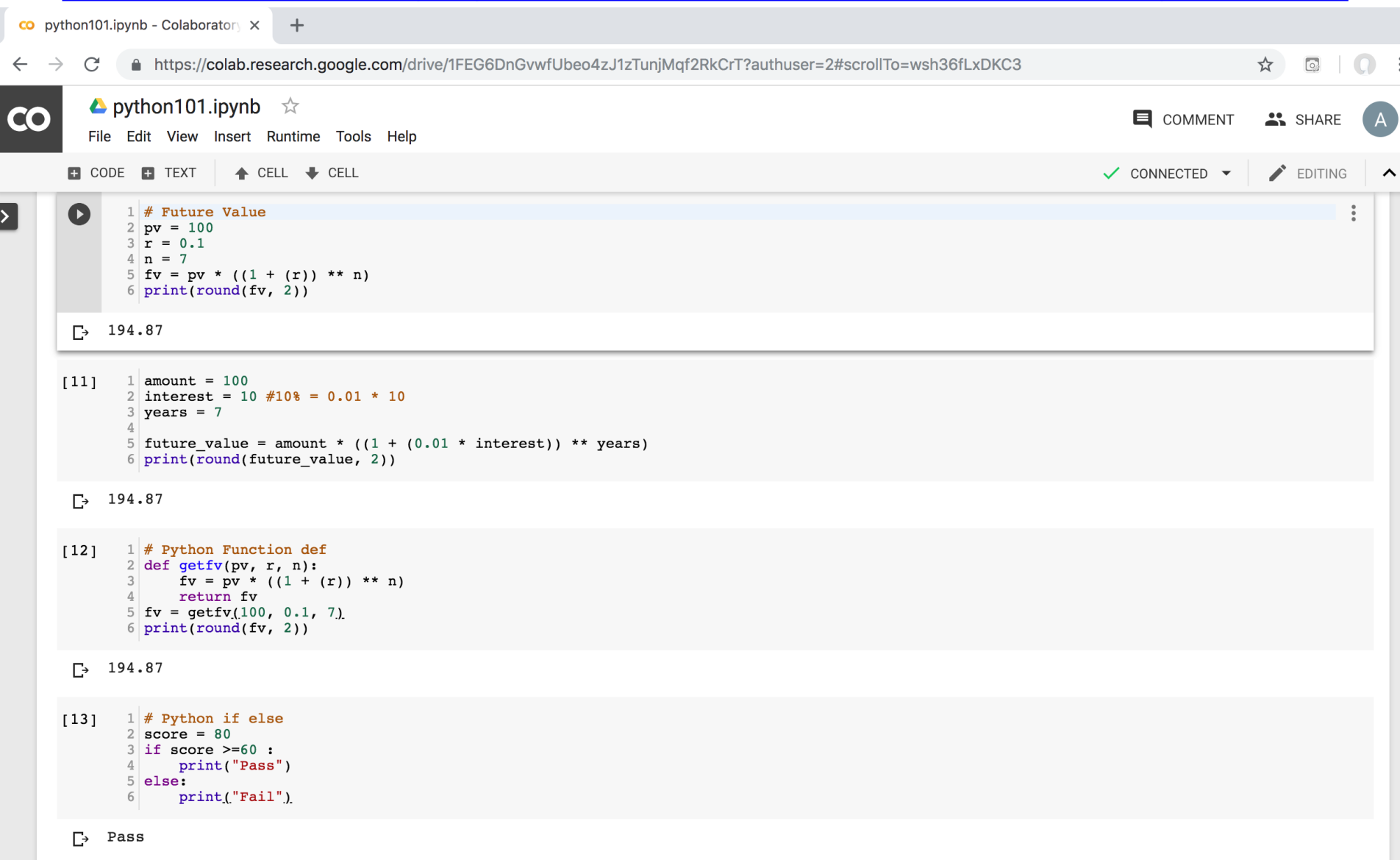
Further down, there are two expandable sections: 'Highlighted Features' and 'TensorFlow execution'. The 'Highlighted Features' section includes a 'Seedbank' subsection with the text: 'Looking for Colab notebooks to learn from? Check out [Seedbank](#), a place to discover interactive machine learning examples.' The 'TensorFlow execution' section contains the text: 'Colaboratory allows you to execute TensorFlow code in your browser with a single click. The example below adds two matrices.'

At the bottom of the 'TensorFlow execution' section, a matrix addition is shown:

$$\begin{bmatrix} 1. & 1. & 1. \end{bmatrix} + \begin{bmatrix} 1. & 2. & 3. \end{bmatrix} = \begin{bmatrix} 2. & 3. & 4. \end{bmatrix}$$

Python in Google Colab (Python101)

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>



python101.ipynb - Colaboratory

File Edit View Insert Runtime Tools Help

COMMENT SHARE

CONNECTED EDITING

```
1 # Future Value
2 pv = 100
3 r = 0.1
4 n = 7
5 fv = pv * ((1 + (r)) ** n)
6 print(round(fv, 2))
```

194.87

```
[11] 1 amount = 100
2 interest = 10 #10% = 0.01 * 10
3 years = 7
4
5 future_value = amount * ((1 + (0.01 * interest)) ** years)
6 print(round(future_value, 2))
```

194.87

```
[12] 1 # Python Function def
2 def getfv(pv, r, n):
3     fv = pv * ((1 + (r)) ** n)
4     return fv
5 fv = getfv(100, 0.1, 7)
6 print(round(fv, 2))
```

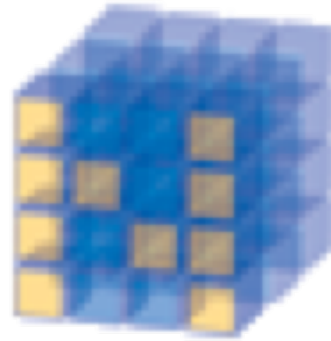
194.87

```
[13] 1 # Python if else
2 score = 80
3 if score >=60 :
4     print("Pass")
5 else:
6     print("Fail").
```

Pass

<https://tinyurl.com/aintpupython101>

NumPy



NumPy
Base

**N-dimensional array
package**

Python

matplotlib

matplotlib

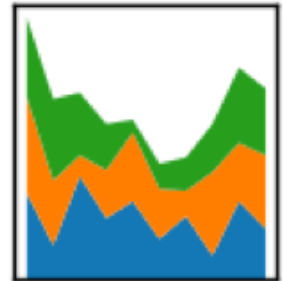
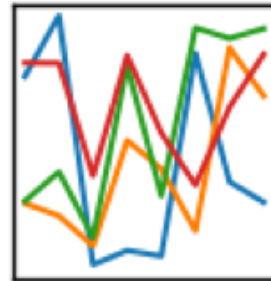
The logo for Matplotlib, which is a circular plot with a white background and a light blue border. Inside the circle, there are several colored segments (orange, yellow, green, blue) radiating from the center, resembling a pie chart or a polar plot.

Python

Pandas

pandas

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



Iris flower data set

setosa



versicolor



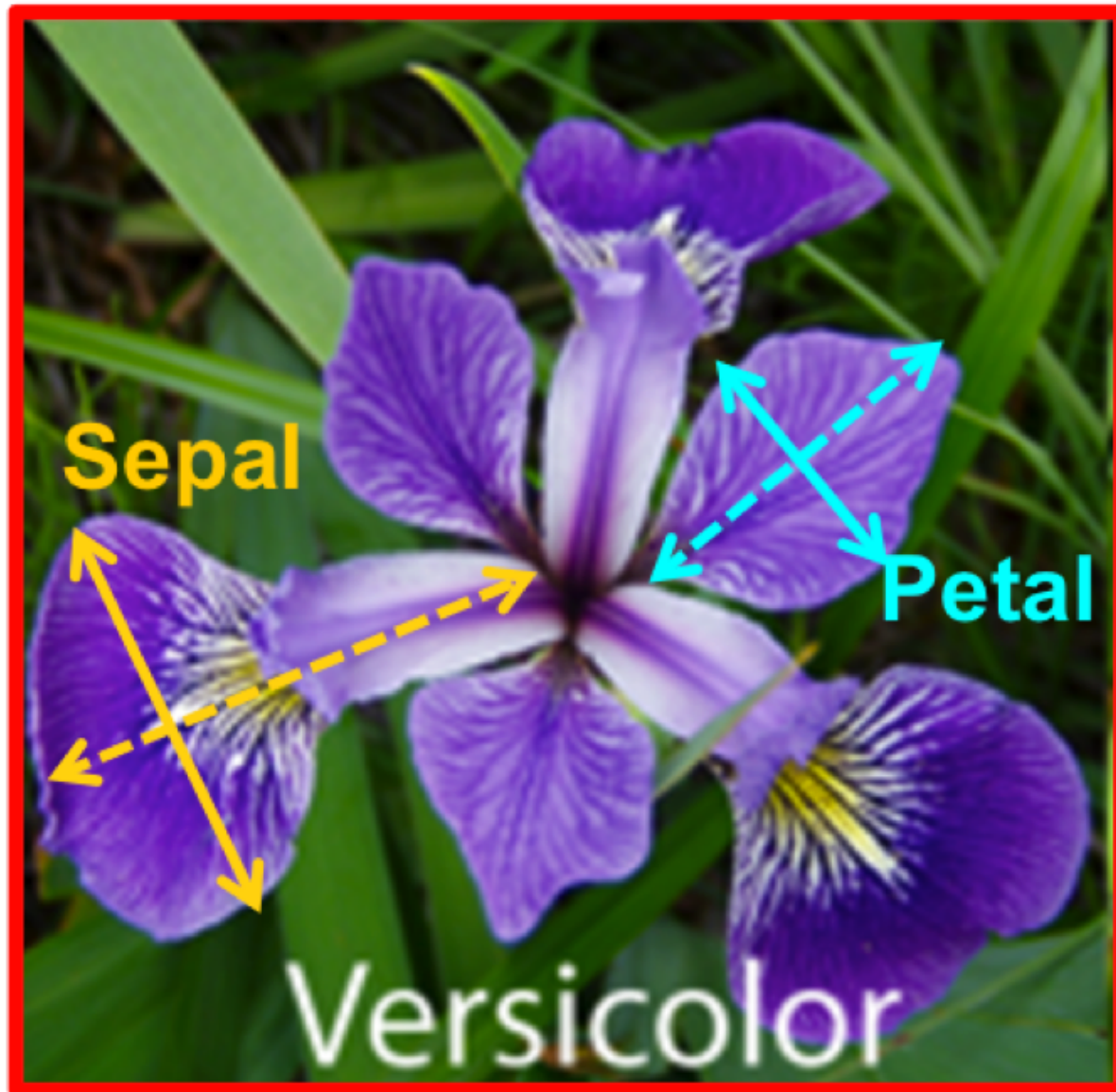
virginica



Source: https://en.wikipedia.org/wiki/Iris_flower_data_set

Source: <http://suruchifialoke.com/2016-10-13-machine-learning-tutorial-iris-classification/>

Iris Classification



iris.data

<https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data>

```
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
5.4,3.7,1.5,0.2,Iris-setosa
4.8,3.4,1.6,0.2,Iris-setosa
4.8,3.0,1.4,0.1,Iris-setosa
4.3,3.0,1.1,0.1,Iris-setosa
5.8,4.0,1.2,0.2,Iris-setosa
5.7,4.4,1.5,0.4,Iris-setosa
5.4,3.9,1.3,0.4,Iris-setosa
5.1,3.5,1.4,0.3,Iris-setosa
5.7,3.8,1.7,0.3,Iris-setosa
5.1,3.8,1.5,0.3,Iris-setosa
5.4,3.4,1.7,0.2,Iris-setosa
5.1,3.7,1.5,0.4,Iris-setosa
4.6,3.6,1.0,0.2,Iris-setosa
5.1,3.3,1.7,0.5,Iris-setosa
4.8,3.4,1.9,0.2,Iris-setosa
5.0,3.0,1.6,0.2,Iris-setosa
5.0,3.4,1.6,0.4,Iris-setosa
```

setosa



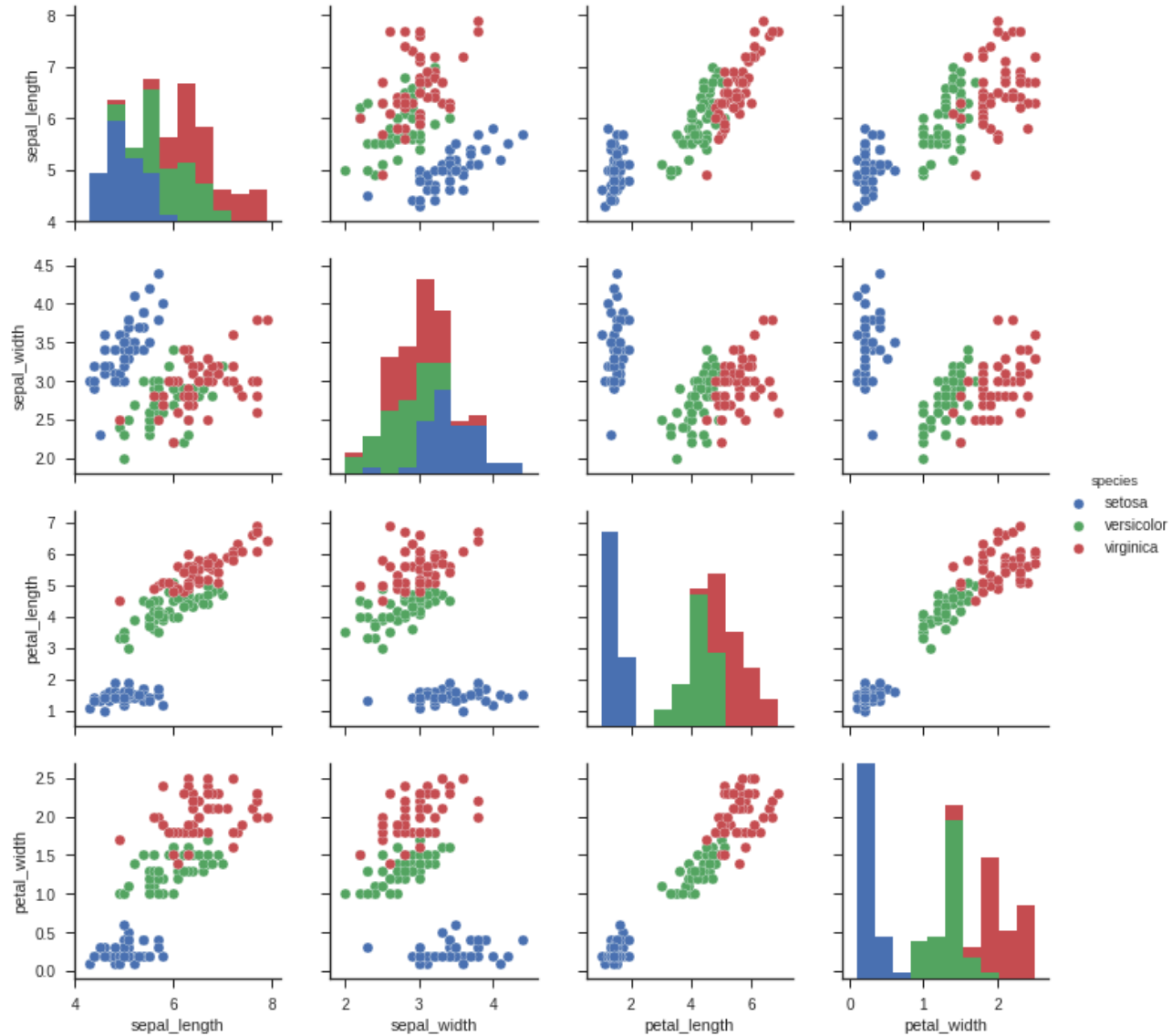
virginica



versicolor



Iris Data Visualization



Connect Google Colab in Google Drive

The image shows a browser window with the Google Drive interface. The address bar displays "https://drive.google.com/drive/u/2/my-drive". The main header includes the Drive logo, a search bar, and navigation icons. On the left sidebar, the "New" button is highlighted with a red dashed border. A dropdown menu is open, listing options: "New folder...", "Upload files...", "Upload folder...", "Google Docs", "Google Sheets", "Google Slides", "More", "Google Forms", "Google Drawings", "Google My Maps", "Google Sites", and "Connect more apps". The "More" and "Connect more apps" options are also highlighted with red dashed borders. The "Storage" section shows "0 bytes of 15 GB used" and a link to "UPGRADE STORAGE". A notification for "Get Backup and Sync for Mac" is visible at the bottom left.

Google Colab








My Drive - Google Drive x +

https://drive.google.com/drive/u/2/my-drive

Drive Search Drive

Connect apps to Drive

All colab x

 ZIP Extractor Extract ZIP files to Google Drive Extraction complete. View extracted files Share Extract another  Test.zip ZIP Extractor 307,585 users	 LUMIN PDF The fast and simple PDF Viewer box	 cloudconvert CloudConvert 373,161 users
 Sejda Merge PDF - Split PDF - Sejda.com ★★★★★ (1106)	 DocHub Edit, Send & Sign PDFs DocHub - Edit and Sign PDF Docu... 2,131,600 users	 Google Forms Google Forms 4,803,614 users

Get Backup and Sync for Mac

Access anywhere
Share easily

Name ↑

Google Colab

My Drive - Google Drive x +

https://drive.google.com/drive/u/2/my-drive

Drive Search Drive

New

My Drive

Computers

Shared with me

Recent

Starred

Trash

Backups

Storage

0 bytes of 15 GB used
[UPGRADE STORAGE](#)


Get Backup and Sync for Mac

Access anywhere

Share easily

Connect apps to Drive

All colab



Colaboratory
offered by <https://colab.research.google.com>
A data analysis tool that combines code, output, and descriptive text into one collaborative document.

+ CONNECT

Productivity
★★★★★ (195)

Name ↑

Connect Colaboratory to Google Drive

The image shows a browser window with the Google Drive interface. The address bar displays `https://drive.google.com/drive/u/2/my-drive`. The main content area is a 'Connect apps to Drive' dialog box. At the top of the dialog, there is a search bar containing the text 'colab'. Below the search bar, a notification for Colaboratory is displayed. The notification features the Colaboratory logo (two yellow circles) and the text: 'Colaboratory was connected to Google Drive.' Below this, there is a checked checkbox and the text: 'Make Colaboratory the default app for files it can open'. At the bottom right of the notification, there is a blue button with the text 'OK' and a red dashed border. To the right of the notification, there is a 'RATE IT' section with a green star icon, the text 'RATE IT', and a rating of five stars with '(195)' reviews. The background of the dialog box shows a list of apps, with the Colaboratory app highlighted. The Google Drive interface is visible in the background, including the search bar, navigation menu, and storage information.

Google Colab

The image shows a browser window with the Google Drive interface. The address bar shows the URL `https://drive.google.com/drive/u/2/my-drive`. The 'New' button is open, displaying a list of options. The 'More' option is highlighted with a red dashed box, and its sub-menu is open, showing 'Colaboratory' also highlighted with a red dashed box. Other options in the 'New' menu include 'New folder...', 'Upload files...', 'Upload folder...', 'Google Docs', 'Google Sheets', 'Google Slides', 'Google Forms', 'Google Drawings', 'Google My Maps', 'Google Sites', and 'Connect more apps'. The 'Storage' section shows '0 bytes of 15 GB used' and a 'UPGRADE STORAGE' button. A 'Get Backup and Sync for Mac' notification is visible at the bottom left.

My Drive - Google Drive

Search Drive

My Drive

Quick Access

New

- My Drive
- Computer
- Shared with me
- Recent
- Starred
- Trash
- Backups

New folder...

Upload files...

Upload folder...

Google Docs

Google Sheets

Google Slides

More

- Google Forms
- Google Drawings
- Google My Maps
- Google Sites
- Colaboratory
- Connect more apps

Name ↑

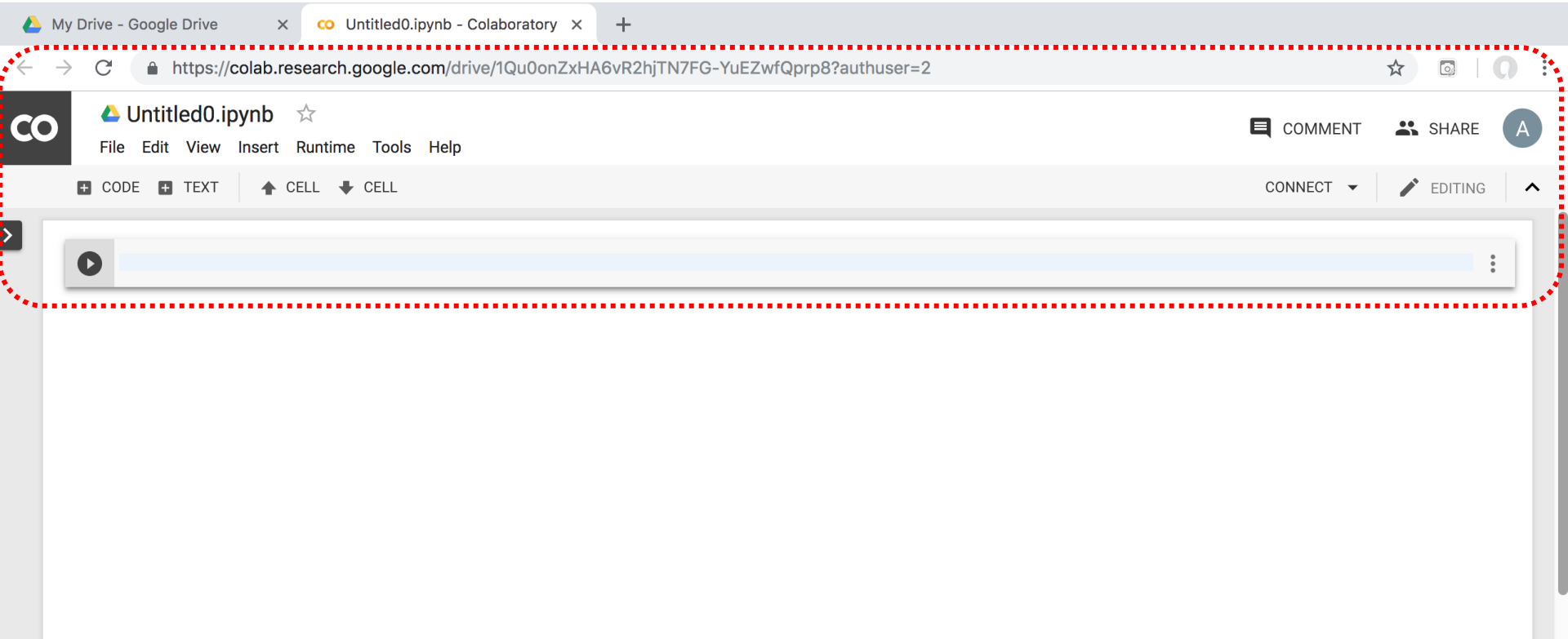
Storage

0 bytes of 15 GB used

UPGRADE STORAGE

Get Backup and Sync for Mac

Google Colab



Google Colab

The image shows a browser window with two tabs: "My Drive - Google Drive" and "Untitled0.ipynb - Colaboratory". The address bar shows the URL: <https://colab.research.google.com/drive/1Qu0onZxHA6vR2hjTN7FG-YuEZwfQprp8?authuser=2>. The main content area displays the Google Colab interface for a notebook titled "Untitled0.ipynb". The menu bar includes "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help". The "Runtime" menu is open, showing the following options:

- Run all (⌘/Ctrl+F9)
- Run before (⌘/Ctrl+F8)
- Run the focused cell (⌘/Ctrl+Enter)
- Run selection (⌘/Ctrl+Shift+Enter)
- Run after (⌘/Ctrl+F10)
- Interrupt execution (⌘/Ctrl+M I)
- Restart runtime... (⌘/Ctrl+M .)
- Restart and run all...
- Reset all runtimes...
- Change runtime type
- Manage sessions

The "Runtime" menu title and the "Change runtime type" option are highlighted with red dashed boxes. The interface also shows a "CONNECT" button and an "EDITING" mode indicator.

Run Jupyter Notebook Python3 GPU Google Colab

The screenshot shows the Google Colab web interface. The browser tab is titled "Untitled0.ipynb - Colaboratory". The address bar shows the URL: <https://colab.research.google.com/drive/1Qu0onZxHA6vR2hjTN7FG-YuEZwfQprp8?authuser=2>. The interface includes a menu bar with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help". Below the menu bar are buttons for "+ CODE", "+ TEXT", "↑ CELL", and "↓ CELL". On the right side, there are buttons for "COMMENT", "SHARE", "CONNECT", and "EDITING".

A "Notebook settings" dialog box is open in the center of the screen. It contains the following options:

- Runtime type:** A dropdown menu currently set to "Python 3".
- Hardware accelerator:** A dropdown menu currently set to "GPU".
- Omit code cell output when saving this notebook

At the bottom of the dialog box, there are two buttons: "CANCEL" and "SAVE".

Google Colab Python Hello World

```
print('Hello World')
```



The screenshot displays the Google Colaboratory web interface. At the top, the browser tab is labeled "Untitled0.ipynb - Colaboratory". The address bar shows the URL: <https://colab.research.google.com/drive/1Qu0onZxHA6vR2hjTN7FG-YuEZwfQprp8?authuser=2#scrollTo=6s-m3sER8G1u>. The page title is "Untitled0.ipynb". The navigation menu includes "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help". On the right side, there are buttons for "COMMENT" and "SHARE", along with a user profile icon labeled "A". Below the menu, the interface shows a toolbar with options: "+ CODE", "+ TEXT", "↑ CELL", and "↓ CELL". The status bar indicates "CONNECTED" with a green checkmark and "EDITING" with a pencil icon. The main workspace contains a code cell with a play button icon and the code `print('Hello World')`. Below the code cell, the output is displayed as "Hello World" with a copy icon to its left.

Data Visualization in Google Colab

<https://colab.research.google.com/drive/1KRqtEUd2Hg4dM2au9bfVQKrxWnWN3O9->

datav.ipynb - Colaboratory

https://colab.research.google.com/drive/1KRqtEUd2Hg4dM2au9bfVQKrxWnWN3O9-?authuser=2

datav.ipynb

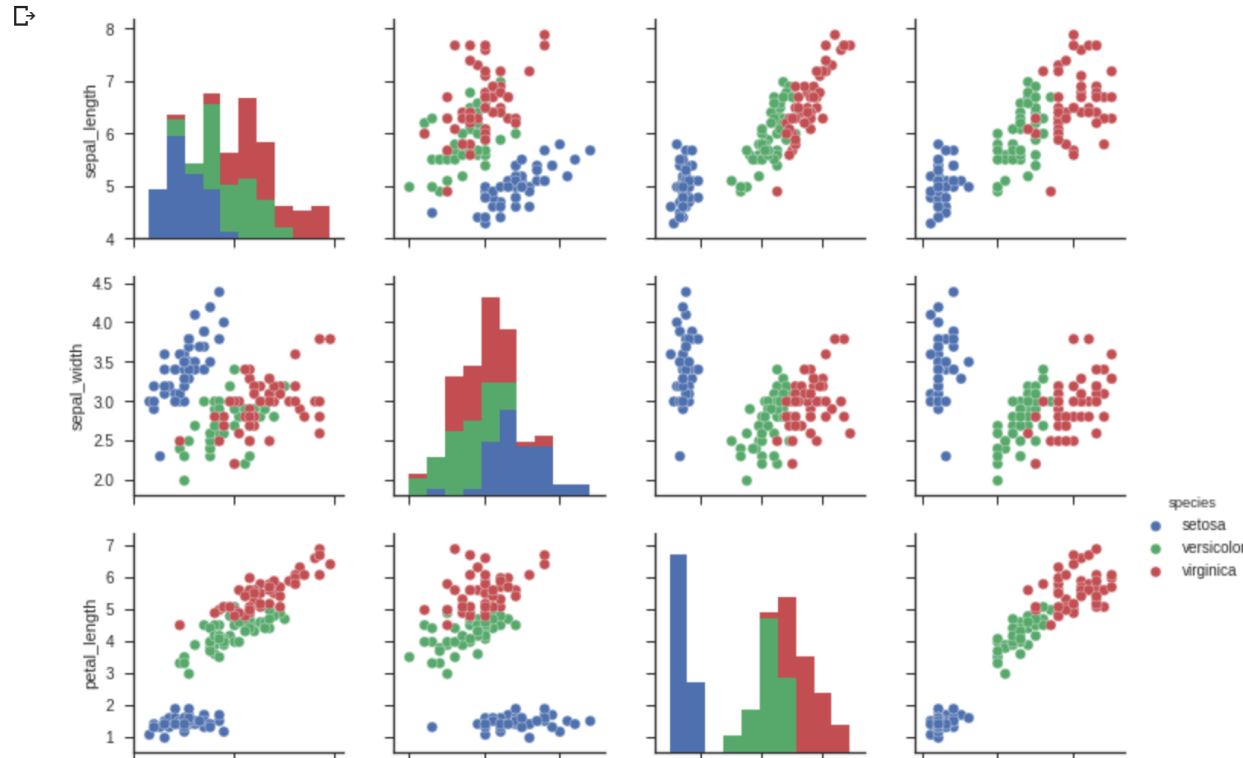
File Edit View Insert Runtime Tools Help

COMMENT SHARE

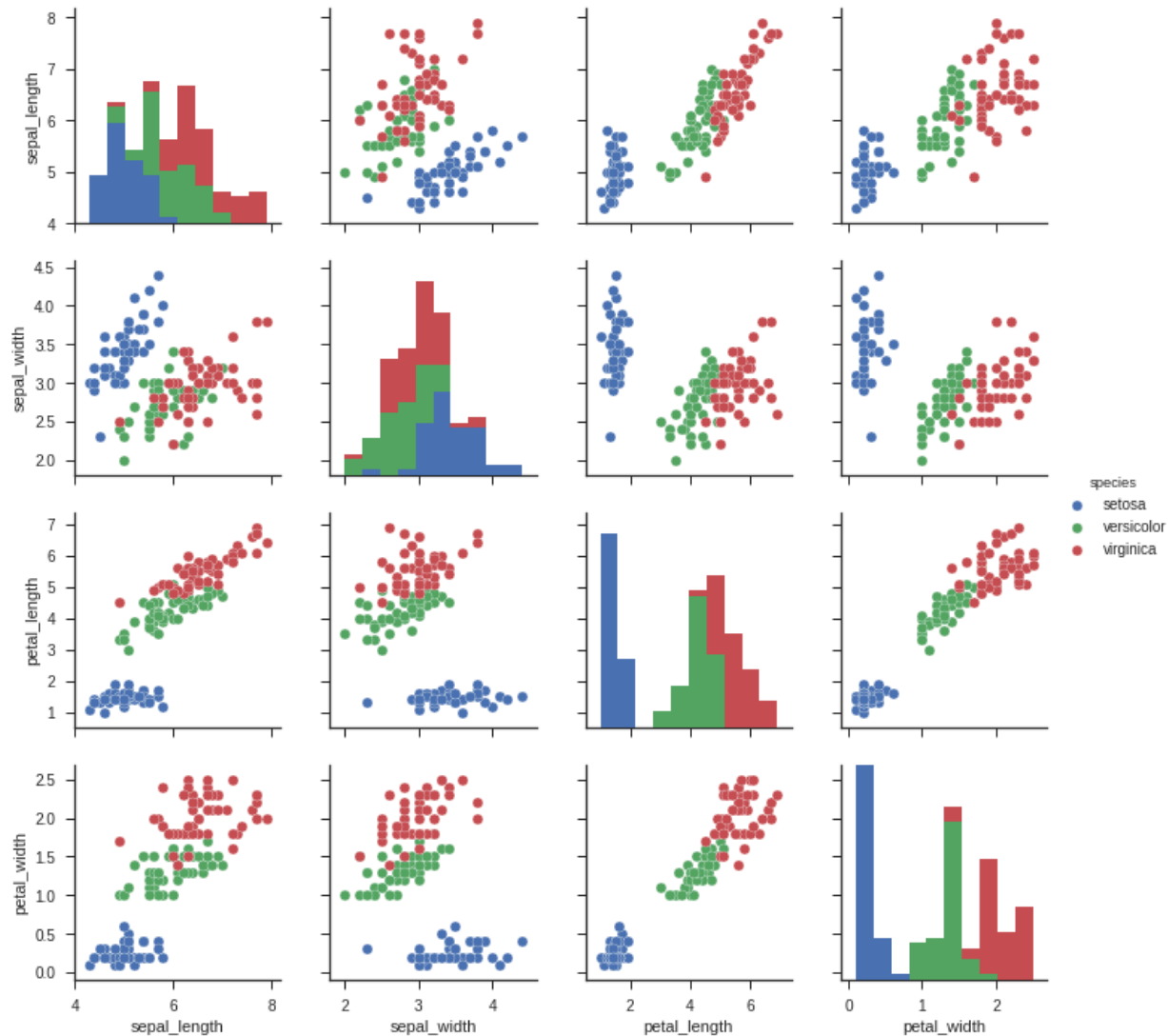
CODE TEXT CELL CELL

CONNECTED EDITING

```
import seaborn as sns
sns.set(style="ticks", color_codes=True)
iris = sns.load_dataset("iris")
g = sns.pairplot(iris, hue="species").
```




```
import seaborn as sns
sns.set(style="ticks", color_codes=True)
iris = sns.load_dataset("iris")
g = sns.pairplot(iris, hue="species")
```



```
import numpy as np
import pandas as pd
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
from pandas.plotting import scatter_matrix

# Load dataset
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
df = pd.read_csv(url, names=names)

print(df.head(10))
print(df.tail(10))
print(df.describe())
print(df.info())
print(df.shape)
print(df.groupby('class').size())

plt.rcParams["figure.figsize"] = (10,8)
df.plot(kind='box', subplots=True, layout=(2,2), sharex=False, sharey=False)
plt.show()

df.hist()
plt.show()

scatter_matrix(df)
plt.show()

sns.pairplot(df, hue="class", size=2)
```

```
import numpy as np
import pandas as pd
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
from pandas.plotting import scatter_matrix
```

```
# Import Libraries
import numpy as np
import pandas as pd
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
from pandas.plotting import scatter_matrix
print('imported')
```

imported

```
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"  
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']  
df = pd.read_csv(url, names=names)  
print(df.head(10))
```

```
# Load dataset  
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"  
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']  
df = pd.read_csv(url, names=names)  
print(df.head(10)).
```

	sepal-length	sepal-width	petal-length	petal-width	class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
5	5.4	3.9	1.7	0.4	Iris-setosa
6	4.6	3.4	1.4	0.3	Iris-setosa
7	5.0	3.4	1.5	0.2	Iris-setosa
8	4.4	2.9	1.4	0.2	Iris-setosa
9	4.9	3.1	1.5	0.1	Iris-setosa

df.tail(10)

```
print(df.tail(10)).
```

	sepal-length	sepal-width	petal-length	petal-width	class
140	6.7	3.1	5.6	2.4	Iris-virginica
141	6.9	3.1	5.1	2.3	Iris-virginica
142	5.8	2.7	5.1	1.9	Iris-virginica
143	6.8	3.2	5.9	2.3	Iris-virginica
144	6.7	3.3	5.7	2.5	Iris-virginica
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

df.describe()

```
print(df.describe())
```

	sepal-length	sepal-width	petal-length	petal-width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

```
print(df.info())  
print(df.shape)
```

```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 150 entries, 0 to 149  
Data columns (total 5 columns):  
sepal-length      150 non-null float64  
sepal-width       150 non-null float64  
petal-length      150 non-null float64  
petal-width       150 non-null float64  
class             150 non-null object  
dtypes: float64(4), object(1)  
memory usage: 5.9+ KB  
None
```

```
print(df.shape)
```

```
(150, 5)
```

```
df.groupby('class').size()
```

```
print(df.groupby('class').size())
```

```
class
Iris-setosa      50
Iris-versicolor 50
Iris-virginica   50
dtype: int64
```

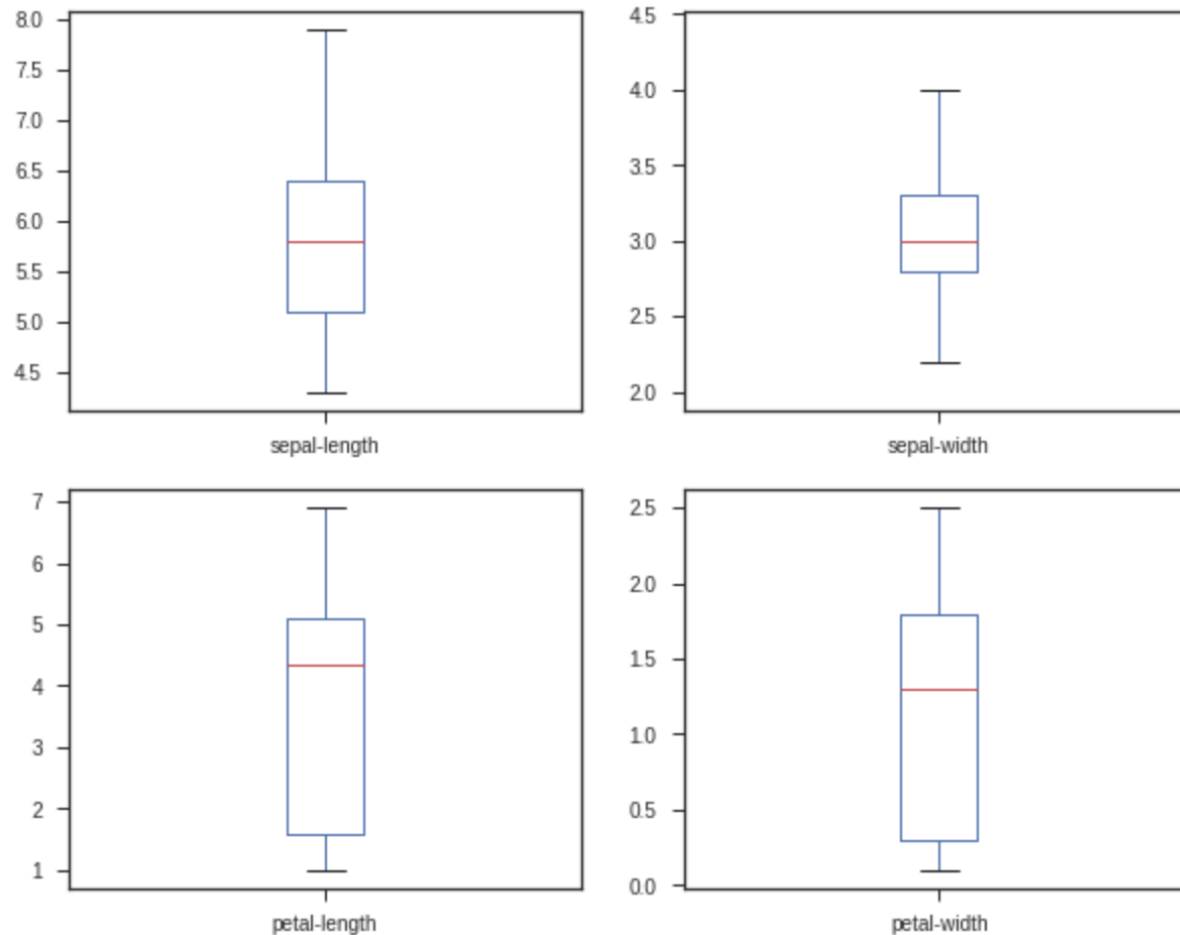


```
plt.rcParams["figure.figsize"] = (10,8)
```

```
df.plot(kind='box', subplots=True, layout=(2,2), sharex=False, sharey=False)
```

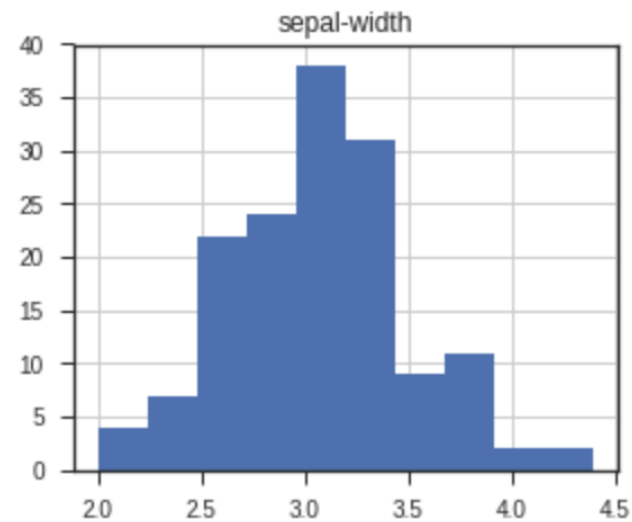
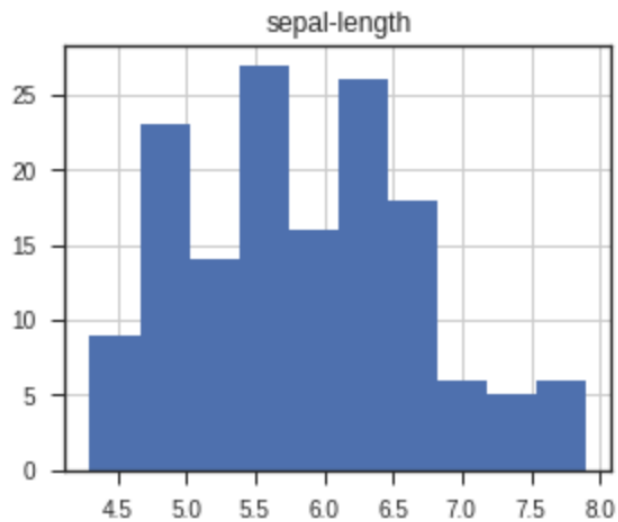
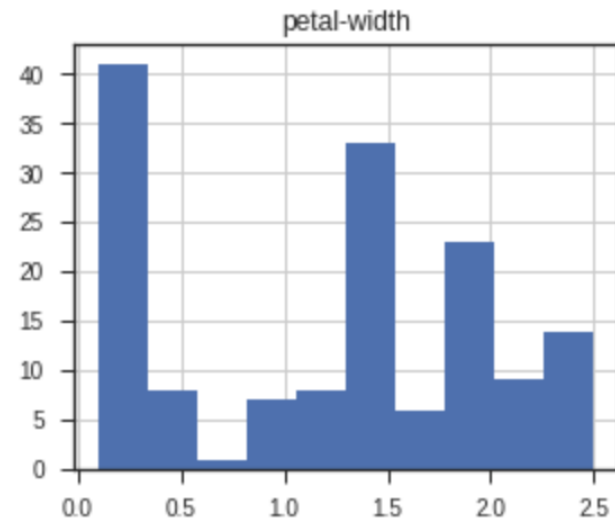
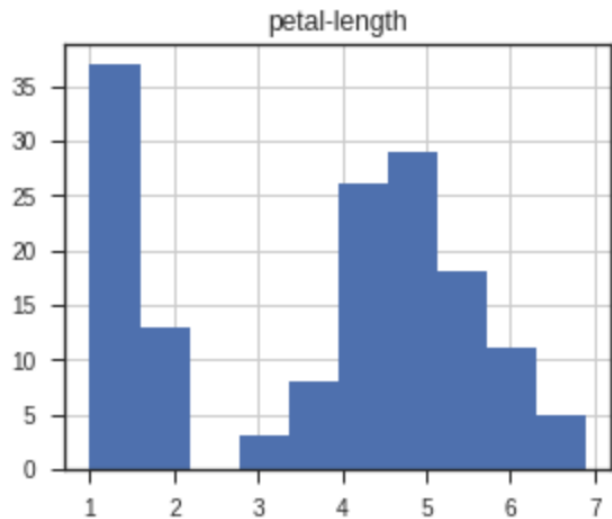
```
plt.show()
```

```
plt.rcParams["figure.figsize"] = (10,8)  
df.plot(kind='box', subplots=True, layout=(2,2), sharex=False, sharey=False)  
plt.show()
```



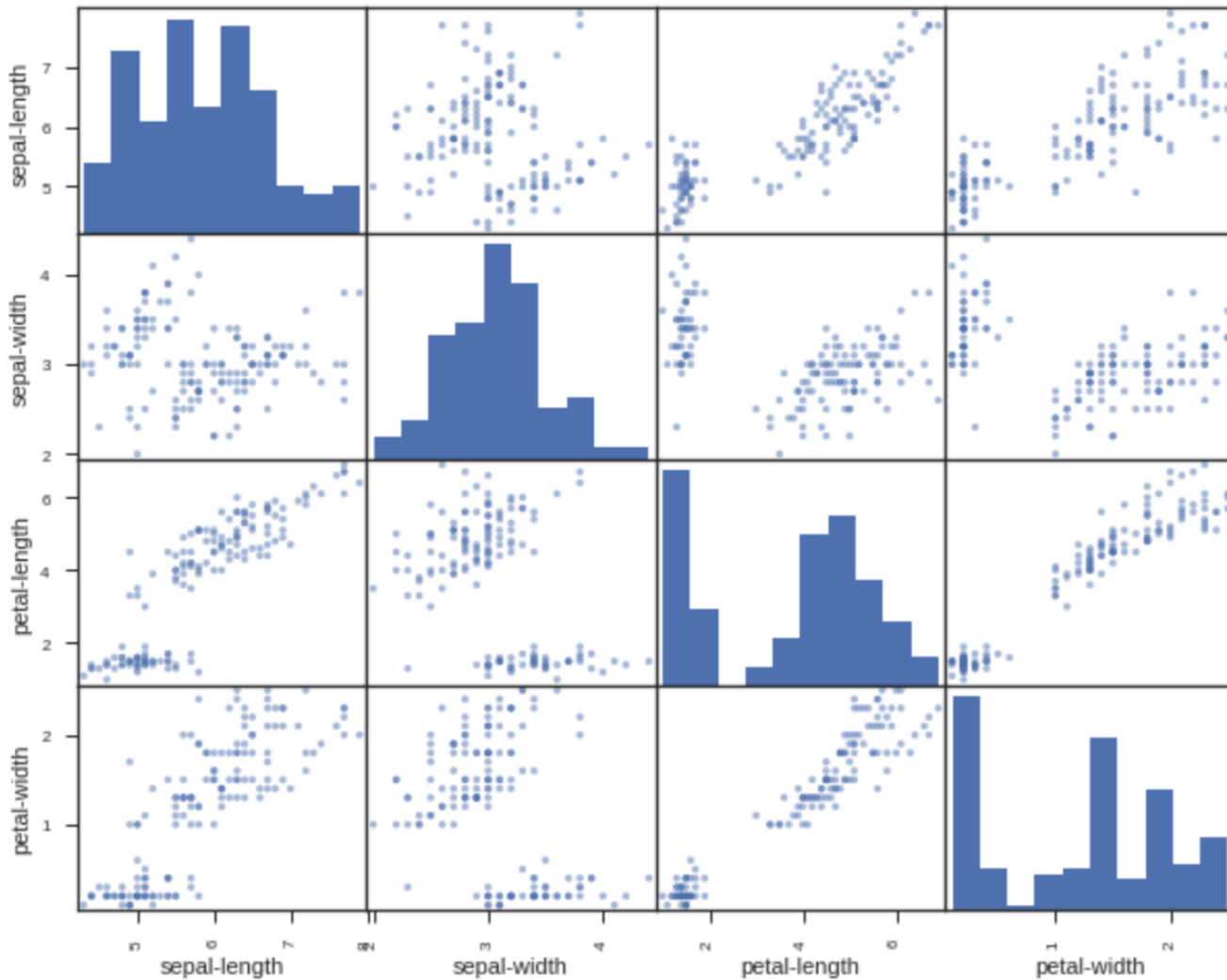
```
df.hist()  
plt.show()
```

```
df.hist()  
plt.show()
```



```
scatter_matrix(df)
plt.show()
```

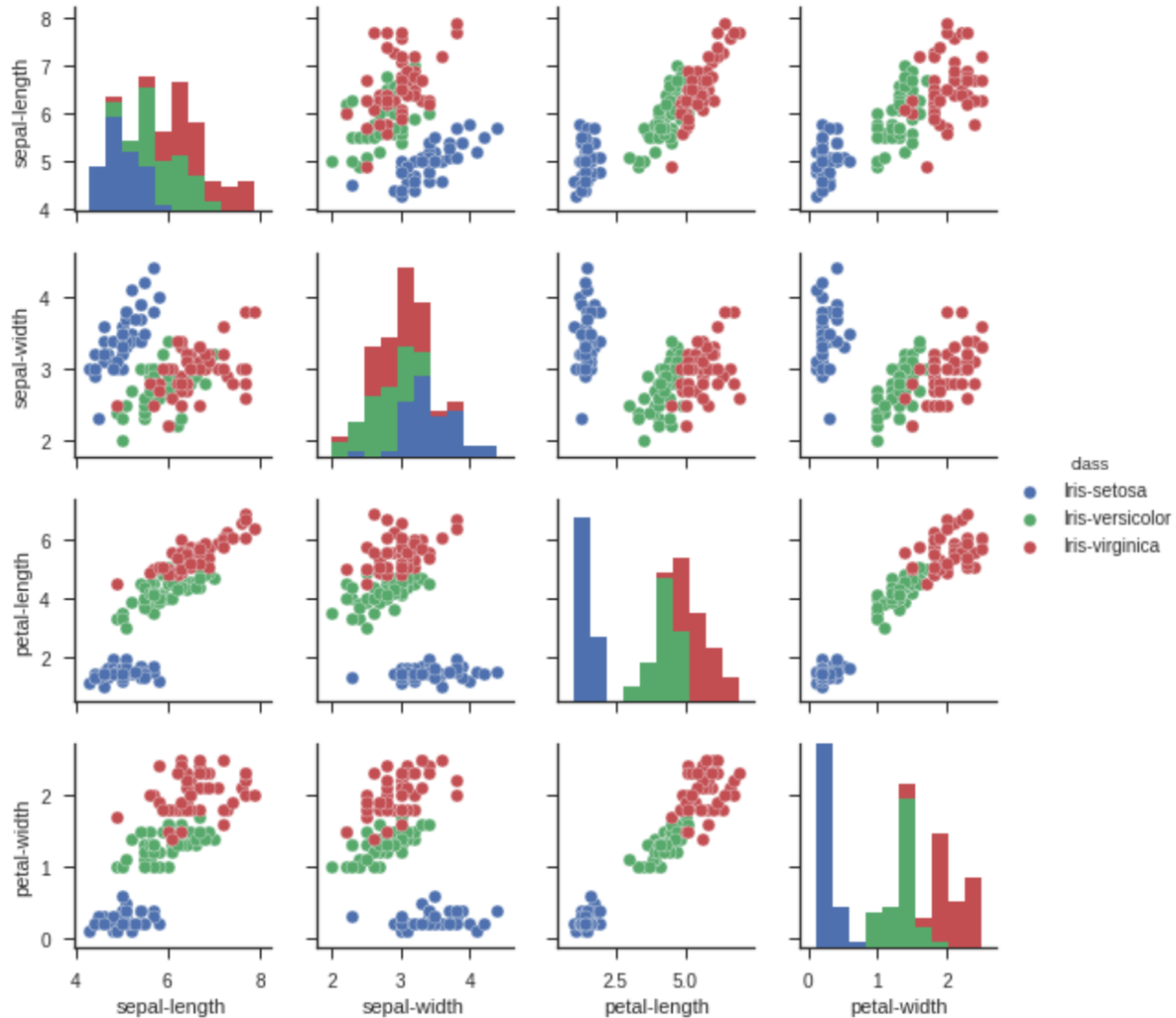
```
scatter_matrix(df)
plt.show(.
```



sns.pairplot(df, hue="class", size=2)

```
sns.pairplot(df, hue="class", size=2)
```

```
<seaborn.axisgrid.PairGrid at 0x7f1d21267390>
```





Anaconda
The Most Popular
Python
Data Science Platform

Download Anaconda



[Documentation](#) [Blog](#) [Contact](#) [Anaconda Cloud](#) [Q](#)

[What is Anaconda?](#) [Products](#) [Support](#) [Community](#) [Resources](#) [About](#)

[Download](#)

[Don't Miss AnacondaCon Apr 8-11 Austin TX!](#)

Download Anaconda Distribution

Version 5.1 | Release Date: February 15, 2018

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



[Anaconda 5.1 For macOS Installer](#)

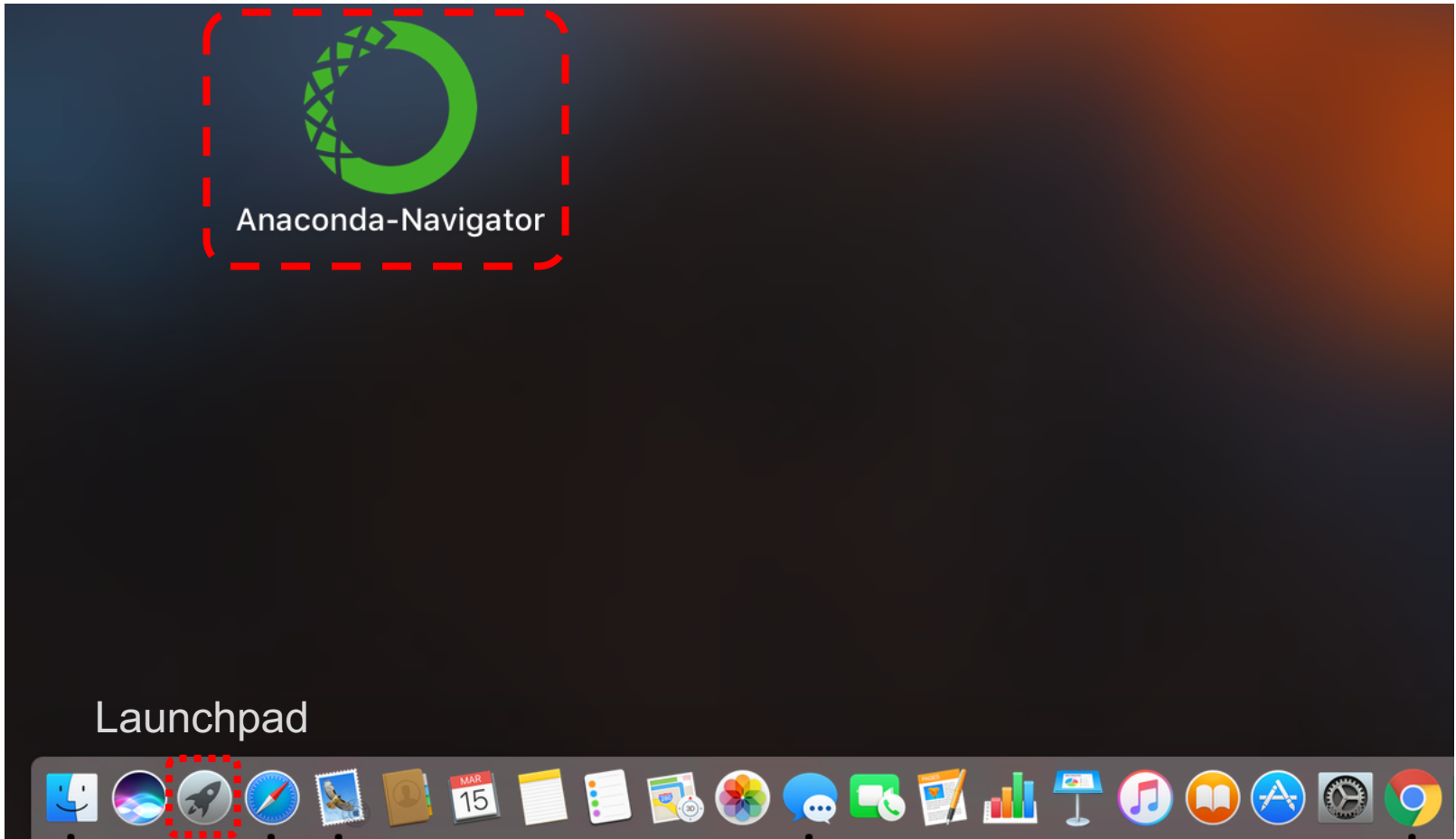
<https://www.anaconda.com/download>



Python

HelloWorld

Anaconda-Navigator



Anaconda Navigator

Anaconda Navigator

ANACONDA NAVIGATOR

Sign in to Anaconda Cloud

Home

Environments

Learning

Community

Documentation

Developer Blog

Feedback



Applications on

base (root)

Channels

Refresh



jupyterlab

0.31.5

An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Launch



notebook

5.4.0

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

Launch



qtconsole

4.3.1

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Launch



spyder

3.2.6

Scientific PYTHON Development Environment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

Launch

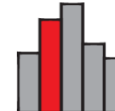


vscode

1.22.2

Streamlined code editor with support for development operations like debugging, task running and version control.

Launch



glueviz

0.12.4

Multidimensional data visualization across files. Explore relationships within and among related datasets.

Install

Jupyter Notebook

The screenshot shows a web browser window with the Jupyter Notebook interface. The browser's address bar displays the URL `localhost:8888/tree/Documents/Data/BDA`. The Jupyter logo and a "Logout" button are visible in the top left and right corners, respectively. Below the navigation tabs, there are buttons for "Files", "Running", and "Clusters". A message "Select items to perform actions on them." is followed by "Upload", "New", and a refresh icon. The file browser shows the path `/ Documents / Data / BDA` with a selection count of 0. A table with columns "Name" and "Last Modified" is shown, containing a single entry `..` with a timestamp of "seconds ago". A message "The notebook list is empty." is displayed at the bottom of the file browser area.

Home x

localhost:8888/tree/Documents/Data/BDA

jupyter Logout

Files Running Clusters

Select items to perform actions on them. Upload New ↕ ↻

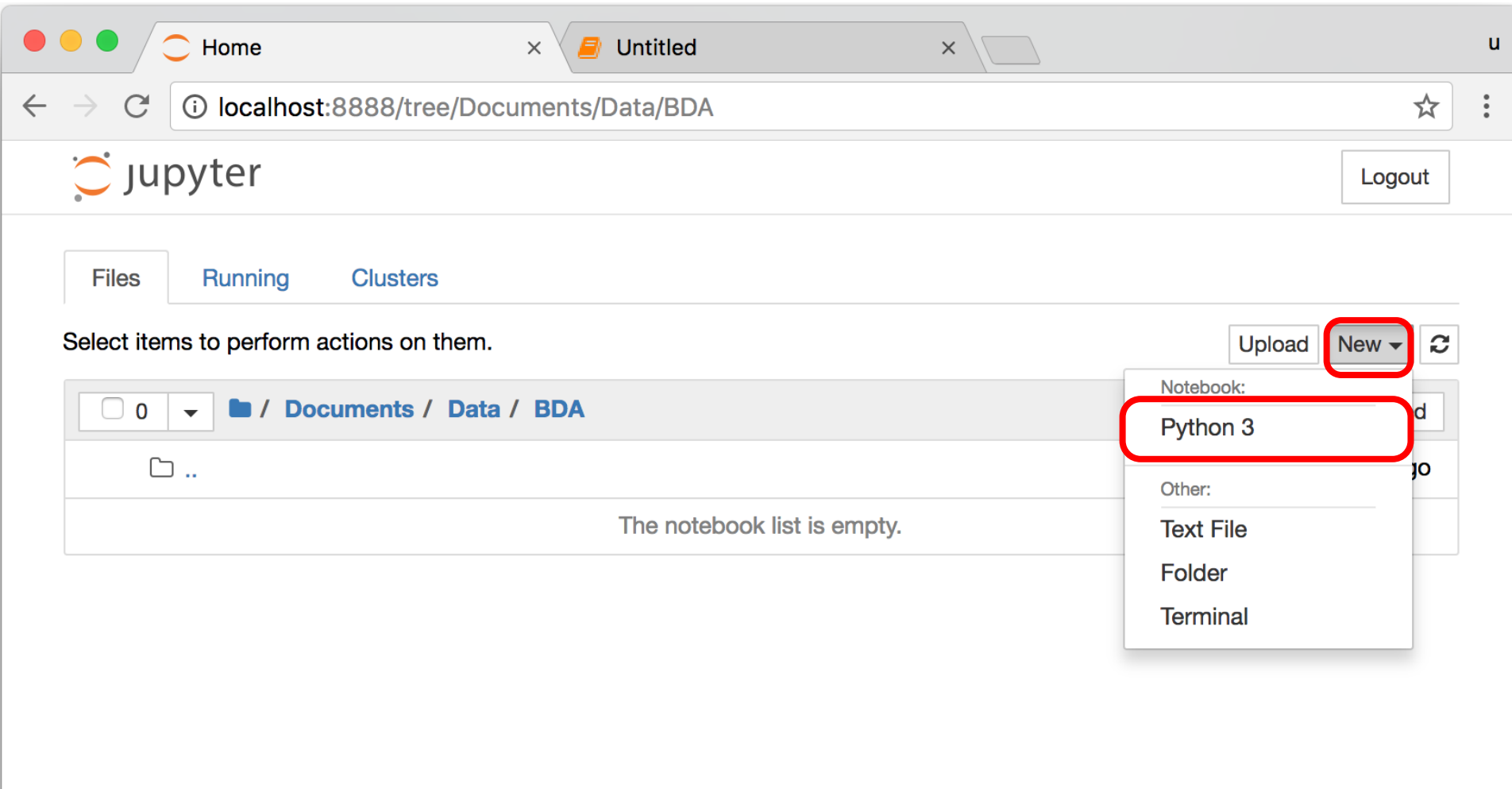
0 ▾ / Documents / Data / BDA

Name ↓	Last Modified
..	seconds ago

The notebook list is empty.

Jupyter Notebook

New Python 3



The screenshot shows a web browser window with the Jupyter Notebook interface. The browser's address bar displays `localhost:8888/tree/Documents/Data/BDA`. The Jupyter logo and a "Logout" button are visible at the top. Below the navigation tabs ("Files", "Running", "Clusters"), there is a prompt: "Select items to perform actions on them." To the right of this prompt are "Upload" and "New" buttons. The "New" button is highlighted with a red circle, and its dropdown menu is open, showing options: "Notebook:", "Python 3" (highlighted with a red box), "Other:", "Text File", "Folder", and "Terminal". The file browser below shows the path `/ Documents / Data / BDA` and a message: "The notebook list is empty."

print("hello, world")

The screenshot shows a web browser window with two tabs: 'Home' and 'HelloWorld'. The address bar shows the URL `localhost:8888/notebooks/Documents/Data/BDA/HelloWorld.ipynb`. The Jupyter interface includes a top navigation bar with 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', 'Widgets', and 'Help' menus. A 'Trusted' status indicator and 'Python 3' version are also visible. Below the menu is a toolbar with icons for saving, adding, deleting, and running cells. The 'Run' button, represented by a play icon, is highlighted with a red box. The main workspace contains a code cell with the text `In [1]: print("hello, world")` and its output `hello, world`. The code line is also highlighted with a red box. Below the code cell is an empty input field labeled `In []:`.

```
from platform import python_version
print("Python Version:", python_version())
```

The screenshot shows a web browser window with a Jupyter Notebook. The browser tabs include 'Home' and 'HelloWorld'. The address bar shows 'localhost:8888/notebooks/Documents/Data/BDA/HelloWorld.ipynb'. The Jupyter interface includes a header with the 'jupyter HelloWorld' logo, a 'Logout' button, and a menu bar with options: File, Edit, View, Insert, Cell, Kernel, Widgets, Help. A 'Trusted' status indicator and 'Python 3' are also visible. Below the menu bar is a toolbar with icons for file operations and execution. The notebook content consists of three input cells:

- In [1]:** `print("hello, world")`
Output: `hello, world`
- In [2]:** `from platform import python_version`
`print("Python Version:", python_version())`
Output: `Python Version: 3.6.5`
- In []:** (Empty cell)



Python

Programming



Python Fiddle

Python Cloud IDE | Python Fiddle x

pythonfiddle.com

Run Reset Share Import Login Language

Python Fiddle Python Cloud IDE

G+1 2.6k

Examples

- Chaining comparison operators
- Decorators
- Creating generators objects
- Enumerate
- Function closure
- Lex tokenizer
- Step argument in slice operators
- For Else
- Verbose regular expressions
- In-place value swapping
- Function argument unpacking

Packages

Hotkeys

```
1 print("Hello Python Fiddle")
2
```

Title:

Description:

Tags:

A comma-separated list of tags.

Save

Hello Python Fiddle

Text input and output

```
print("Hello World")
```

```
print("Hello World\nThis is a message")
```

```
x = 3  
print(x)
```

```
x = 2  
y = 3  
print(x, ' ', y)
```

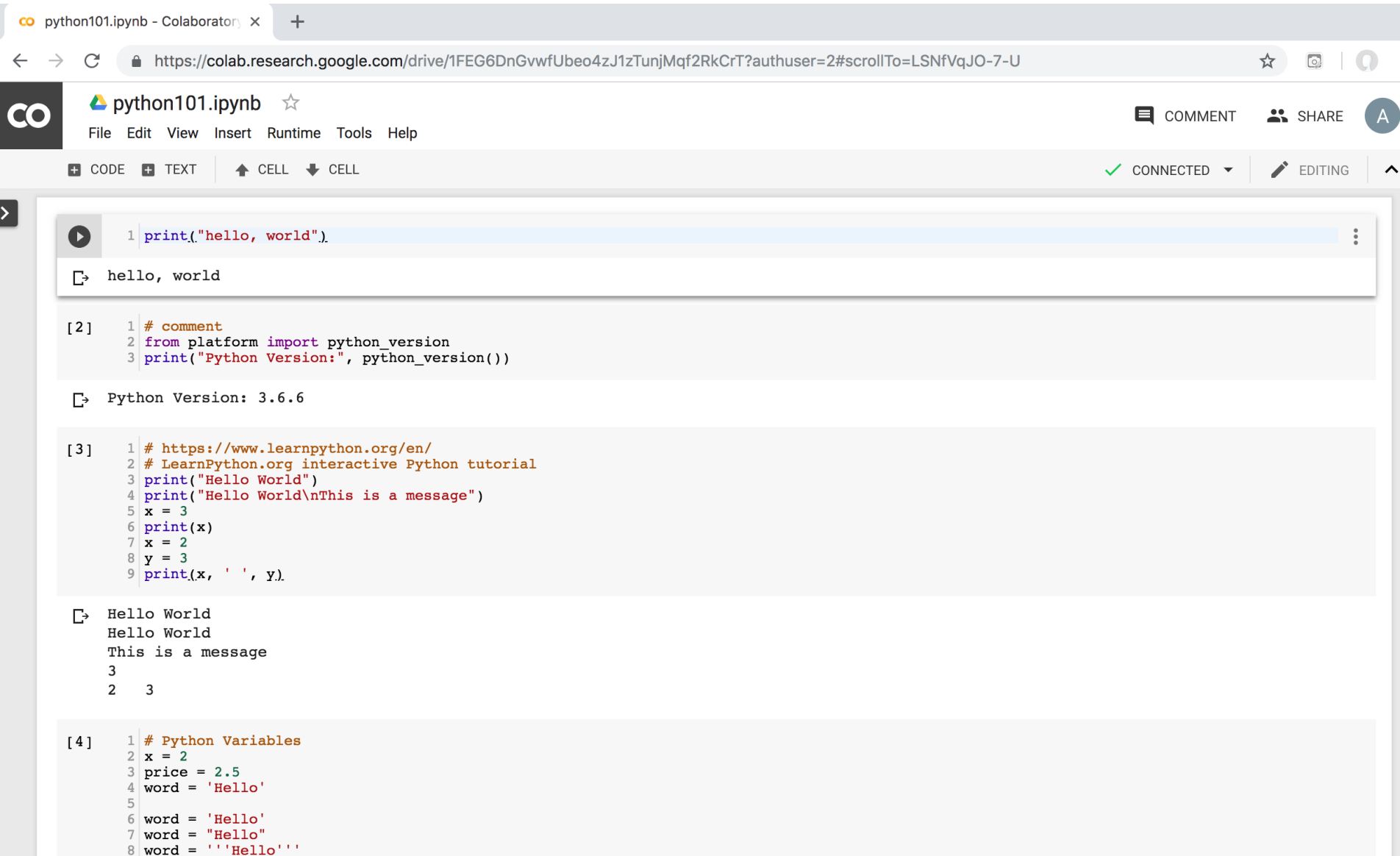
```
name = input("Enter a name: ")
```

```
x = int(input("What is x? "))
```

```
x = float(input("Write a number "))
```

Python in Google Colab

<https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT>



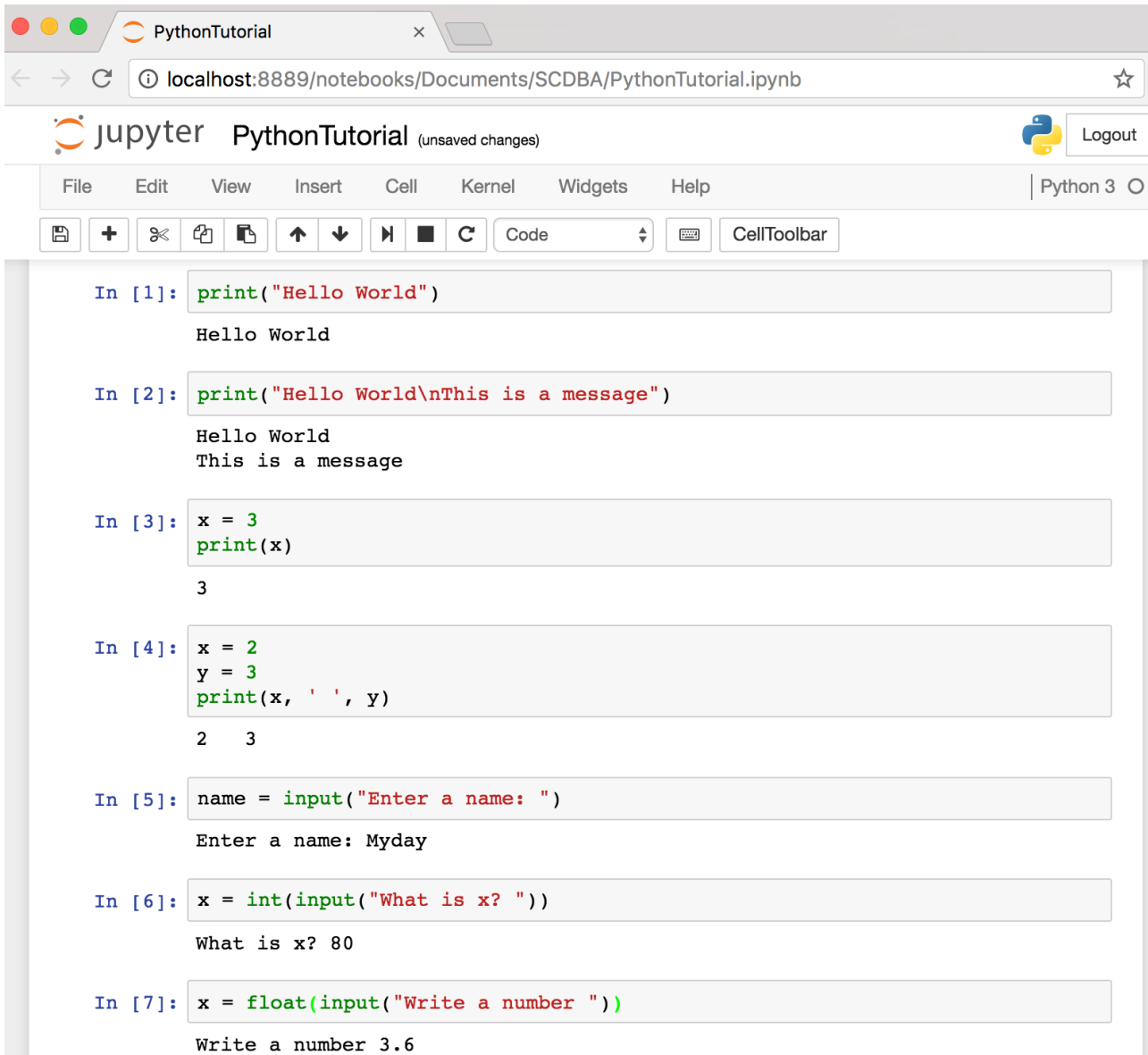
The screenshot shows a Google Colab notebook interface. The browser address bar displays the URL: <https://colab.research.google.com/drive/1FEG6DnGvwfUbeo4zJ1zTunjMqf2RkCrT?authuser=2#scrollTo=LSNfVqJO-7-U>. The notebook title is "python101.ipynb". The interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help) and a toolbar with options for CODE, TEXT, and CELL. The status bar indicates "CONNECTED" and "EDITING".

The notebook contains four code cells:

- Cell 1:** `print("hello, world").` Output: `hello, world`
- Cell 2:** `# comment`, `from platform import python_version`, `print("Python Version:", python_version())`. Output: `Python Version: 3.6.6`
- Cell 3:** `# https://www.learnpython.org/en/`, `# LearnPython.org interactive Python tutorial`, `print("Hello World")`, `print("Hello World\nThis is a message")`, `x = 3`, `print(x)`, `x = 2`, `y = 3`, `print(x, ' ', y).` Output: `Hello World`, `Hello World`, `This is a message`, `3`, `2 3`
- Cell 4:** `# Python Variables`, `x = 2`, `price = 2.5`, `word = 'Hello'`, `word = 'Hello'`, `word = "Hello"`, `word = '''Hello'''`

<https://tinyurl.com/aintpuython101>

Text input and output



The screenshot shows a Jupyter Notebook window titled "PythonTutorial" with the URL `localhost:8889/notebooks/Documents/SCDBA/PythonTutorial.ipynb`. The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations and execution. The notebook contains seven code cells, each with its input and output:

```
In [1]: print("Hello World")
Hello World
```

```
In [2]: print("Hello World\nThis is a message")
Hello World
This is a message
```

```
In [3]: x = 3
print(x)
3
```

```
In [4]: x = 2
y = 3
print(x, ' ', y)
2  3
```

```
In [5]: name = input("Enter a name: ")
Enter a name: Myday
```

```
In [6]: x = int(input("What is x? "))
What is x? 80
```

```
In [7]: x = float(input("Write a number "))
Write a number 3.6
```

Variables

```
x = 2  
price = 2.5  
word = 'Hello'
```

```
word = 'Hello'  
word = "Hello"  
word = '''Hello'''
```

```
x = 2  
x = x + 1  
x = 5
```

Python Basic Operators

```
print('7 + 2 =', 7 + 2)
print('7 - 2 =', 7 - 2)
print('7 * 2 =', 7 * 2)
print('7 / 2 =', 7 / 2)
print('7 // 2 =', 7 // 2)
print('7 % 2 =', 7 % 2)
print('7 ** 2 =', 7 ** 2)
```

```
print('7 + 2 =', 7 + 2)
print('7 - 2 =', 7 - 2)
print('7 * 2 =', 7 * 2)
print('7 / 2 =', 7 / 2)
print('7 // 2 =', 7 // 2)
print('7 % 2 =', 7 % 2)
print('7 ** 2 =', 7 ** 2)
```

```
7 + 2 = 9
7 - 2 = 5
7 * 2 = 14
7 / 2 = 3.5
7 // 2 = 3
7 % 2 = 1
7 ** 2 = 49
```

BMI Calculator in Python

```
height_cm = float(input("Enter your height in cm: "))
weight_kg = float(input("Enter your weight in kg: "))

height_m = height_cm/100
BMI = (weight_kg/(height_m**2))

print("Your BMI is: " + str(round(BMI,1)))
```

BMI Calculator in Python

```
In [1]: height_cm = float(input("Enter your height in cm: "))
weight_kg = float(input("Enter your weight in kg: "))

height_m = height_cm/100
BMI = (weight_kg/(height_m**2))

print("Your BMI is: " + str(round(BMI,1)))
```

```
Enter your height in cm: 170
Enter your weight in kg: 60
Your BMI is: 20.8
```

```
In [ ]:
```

Future value
of a specified
principal amount,
rate of interest, and
a number of years

Future Value (FV)

```
# How much is your $100 worth after 7 years?  
print(100 * 1.1 ** 7)  
# output = 194.87
```

```
print(100 * 1.1 ** 7)
```

```
194.87171000000012
```

Future Value (FV)

```
pv = 100  
r = 0.1  
n = 7
```

```
fv = pv * ((1 + (r)) ** n)  
print(round(fv, 2))
```

```
pv = 100  
r = 0.1  
n = 7  
  
fv = pv * ((1 + (r)) ** n)  
print(round(fv, 2))
```

```
194.87
```

Future Value (FV)

```
amount = 100
interest = 10 #10% = 0.01 * 10
years = 7

future_value = amount * ((1 + (0.01 * interest)) ** years)
print(round(future_value, 2))
```

```
amount = 100
interest = 10 #10% = 0.01 * 10
years = 7

future_value = amount * ((1 + (0.01 * interest)) ** years)
print(round(future_value, 2))
```

194.87

if statements

> greater than
< smaller than
== equals
!= is not

```
score = 80
if score >=60 :
    print("Pass")
else:
    print("Fail")
```

Pass

```
score = 80
if score >=60 :
    print("Pass")
else:
    print("Fail")
```

if elif else

```
score = 90
grade = ""
if score >=90:
    grade = "A"
elif score >= 80:
    grade = "B"
elif score >= 70:
    grade = "C"
elif score >= 60:
    grade = "D"
else:
    grade = "E"
print(grade)
# grade = "A"
```

A

<http://pythontutor.com/visualize.html>
<https://goo.gl/E6w5ph>

for loops

```
for i in range(1,11):  
    print(i)
```

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10
```

for loops

```
for i in range(1,10):  
    for j in range(1,10):  
        print(i, ' * ', j, ' = ', i*j)
```

```
9 * 1 = 9  
9 * 2 = 18  
9 * 3 = 27  
9 * 4 = 36  
9 * 5 = 45  
9 * 6 = 54  
9 * 7 = 63  
9 * 8 = 72  
9 * 9 = 81
```

while loops

```
age = 10

while age < 20:
    print(age)
    age = age + 1
```

```
10
11
12
13
14
15
16
17
18
19
```


Functions

```
def convertCMtoM(xcm) :  
    m = xcm/100  
    return m
```

```
cm = 180  
m = convertCMtoM(cm)  
print(str(m))
```

1.8

Lists

```
x = [60, 70, 80, 90]
print(len(x))
print(x[0])
print(x[1])
print(x[-1])
```

4

60

70

90

Tuples

A **tuple** in Python is a collection that **cannot be modified**.

A tuple is defined using **parenthesis**.

```
x = (10, 20, 30, 40, 50)
```

```
print(x[0])
```

```
print(x[1])
```

```
print(x[2])
```

```
print(x[-1])
```

10

20

30

50

Dictionary

```
k = { 'EN': 'English', 'FR': 'French' }  
print(k['EN'])
```

Dictionary

'EN' → 'English'

'FR' → 'French'

English

Sets

```
animals = {'cat', 'dog'}
```

```
animals = {'cat', 'dog'}
print('cat' in animals) # Check if an element is in a set; prints "True"
print('fish' in animals) # prints "False"
animals.add('fish') # Add an element to a set
print('fish' in animals) # Prints "True"
print(len(animals)) # Number of elements in a set; prints "3"
animals.add('cat') # Adding an element that is already in the set does nothing
print(len(animals)) # Prints "3"
animals.remove('cat') # Remove an element from a set
print(len(animals)) # Prints "2"
```

```
True
False
True
3
3
2
```

```
animals = {'cat', 'dog'}
print('cat' in animals)
print('fish' in animals)
animals.add('fish')
print('fish' in animals)
print(len(animals))
animals.add('cat')
print(len(animals))
animals.remove('cat')
print(len(animals))
```

File Input / Output

```
with open('myfile.txt', 'w') as file:  
    file.write('Hello World\nThis is Python File Input Output')  
  
with open('myfile.txt', 'r') as file:  
    text = file.read()  
print(text)
```

```
with open('myfile.txt', 'w') as file:  
    file.write('Hello World\nThis is Python File Input Output')
```

```
with open('myfile.txt', 'r') as file:  
    text = file.read()  
print(text)
```

```
Hello World  
This is Python File Input Output
```

```
text
```

```
'Hello World\nThis is Python File Input Output'
```

File Input / Output

```
with open('myfile.txt', 'a+') as file:  
    file.write('\n' + 'New line')
```

```
with open('myfile.txt', 'r') as file:  
    text = file.read()  
print(text)
```

```
with open('myfile.txt', 'a+') as file:  
    file.write('\n' + 'New line')
```

```
with open('myfile.txt', 'r') as file:  
    text = file.read()  
print(text)
```

```
Hello World  
This is Python File Input Output  
New line
```

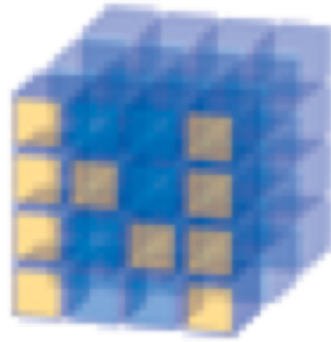
Big Data Analytics

with

Numpy

in Python

NumPy



NumPy
Base

**N-dimensional array
package**

NumPy
is the
fundamental package
for
scientific computing
with **Python.**



NumPy

- NumPy provides a **multidimensional array** object to store homogenous or heterogeneous data; it also provides **optimized functions/methods** to operate on this array object.

NumPy



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.

Getting Started

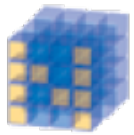
- [Getting NumPy](#)
- [Installing the SciPy Stack](#)
- [NumPy and SciPy documentation page](#)
- [NumPy Tutorial](#)
- [NumPy for MATLAB® Users](#)
- [NumPy functions by category](#)
- [NumPy Mailing List](#)

For more information on the SciPy Stack (for which NumPy provides the fundamental array data structure), see scipy.org.

About NumPy

[License](#)

[Old array packages](#)



NumPy

NumPy ndarray

One-dimensional Array (1-D Array)

0	1			n-1
1	2	3	4	5

Two-dimensional Array (2-D Array)

	0	1			n-1
0	1	2	3	4	5
1	6	7	8	9	10
	11	12	13	14	15
m-1	16	17	18	19	20



NumPy

```
v = list(range(1, 6))
```

```
v
```

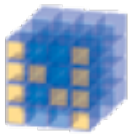
```
2 * v
```

```
import numpy as np
```

```
v = np.arange(1, 6)
```

```
v
```

```
2 * v
```



NumPy

Base

N-dimensional
array package

```
1 v = list(range(1, 6))  
2 v
```

```
[1, 2, 3, 4, 5]
```

```
1 2 * v
```

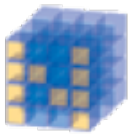
```
[1, 2, 3, 4, 5, 1, 2, 3, 4, 5]
```

```
1 import numpy as np  
2 v = np.arange(1, 6)  
3 v
```

```
array([1, 2, 3, 4, 5])
```

```
1 2 * v
```

```
array([ 2,  4,  6,  8, 10])
```



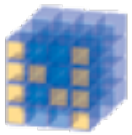
NumPy

NumPy Create Array

```
import numpy as np  
a = np.array([1, 2, 3])  
b = np.array([4, 5, 6])  
c = a * b  
c
```

```
import numpy as np  
a = np.array([1, 2, 3])  
b = np.array([4, 5, 6])  
c = a * b  
c
```

```
array([ 4, 10, 18])
```

NumPy

NumPy

```
1 import numpy as np
2
3 a = np.zeros((2,2)) # Create an array of all zeros
4 print(a)           # Prints "[[ 0.  0.]
5                     #           [ 0.  0.]]"
6
7 b = np.ones((1,2)) # Create an array of all ones
8 print(b)           # Prints "[[ 1.  1.]]"
9
10 c = np.full((2,2), 7) # Create a constant array
11 print(c)             # Prints "[[ 7.  7.]
12                     #           [ 7.  7.]]"
13
14 d = np.eye(2)       # Create a 2x2 identity matrix
15 print(d)            # Prints "[[ 1.  0.]
16                     #           [ 0.  1.]]"
17
18 e = np.random.random((2,2)) # Create an array filled with random values
19 print(e)             # Might print "[[ 0.91940167  0.08143941]
20                     #           [ 0.68744134  0.87236687]]"
```

```
[[0.  0.]
 [0.  0.]]
[[1.  1.]]
[[7 7]
 [7 7]]
[[1.  0.]
 [0.  1.]]
[[0.66258211 0.65552598]
 [0.00429934 0.21695824]]
```

Numpy Quickstart Tutorial

Quickstart tutorial

Prerequisites

Before reading this tutorial you should know a bit of Python. If you would like to refresh your memory, take a look at the [Python tutorial](#).

If you wish to work the examples in this tutorial, you must also have some software installed on your computer. Please see <http://scipy.org/install.html> for instructions.

The Basics

NumPy's main object is the homogeneous multidimensional array. It is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers. In NumPy dimensions are called *axes*. The number of axes is *rank*.

For example, the coordinates of a point in 3D space `[1, 2, 1]` is an array of rank 1, because it has one axis. That axis has a length of 3. In the example pictured below, the array has rank 2 (it is 2-dimensional). The first dimension (axis) has a length of 2, the second dimension has a length of 3.

```
[[ 1., 0., 0.],  
 [ 0., 1., 2.]]
```

NumPy's array class is called `ndarray`. It is also known by the alias `array`. Note that `numpy.array` is not the same as the Standard Python Library class `array.array`, which only handles one-dimensional arrays and offers less functionality. The more important attributes of an `ndarray` object are:

`ndarray.ndim`

the number of axes (dimensions) of the array. In the Python world, the number of dimensions is referred to as *rank*.

`ndarray.shape`

Table Of Contents

- Quickstart tutorial
 - Prerequisites
 - The Basics
 - An example
 - Array Creation
 - Printing Arrays
 - Basic Operations
 - Universal Functions
 - Indexing, Slicing and Iterating
 - Shape Manipulation
 - Changing the shape of an array
 - Stacking together different arrays
 - Splitting one array into several smaller ones
 - Copies and Views
 - No Copy at All
 - View or Shallow Copy
 - Deep Copy
 - Functions and Methods Overview
 - Less Basic
 - Broadcasting rules
 - Fancy indexing and index tricks
 - Indexing with Arrays of

```
import numpy as np
```

```
a = np.arange(15).reshape(3, 5)
```

```
a.shape
```

```
a.ndim
```

```
a.dtype.name
```

```
import numpy as np  
a = np.arange(15).reshape(3, 5)  
a
```

```
array([[ 0,  1,  2,  3,  4],  
       [ 5,  6,  7,  8,  9],  
       [10, 11, 12, 13, 14]])
```

```
print(a.shape)
```

```
(3, 5)
```

```
a.ndim
```

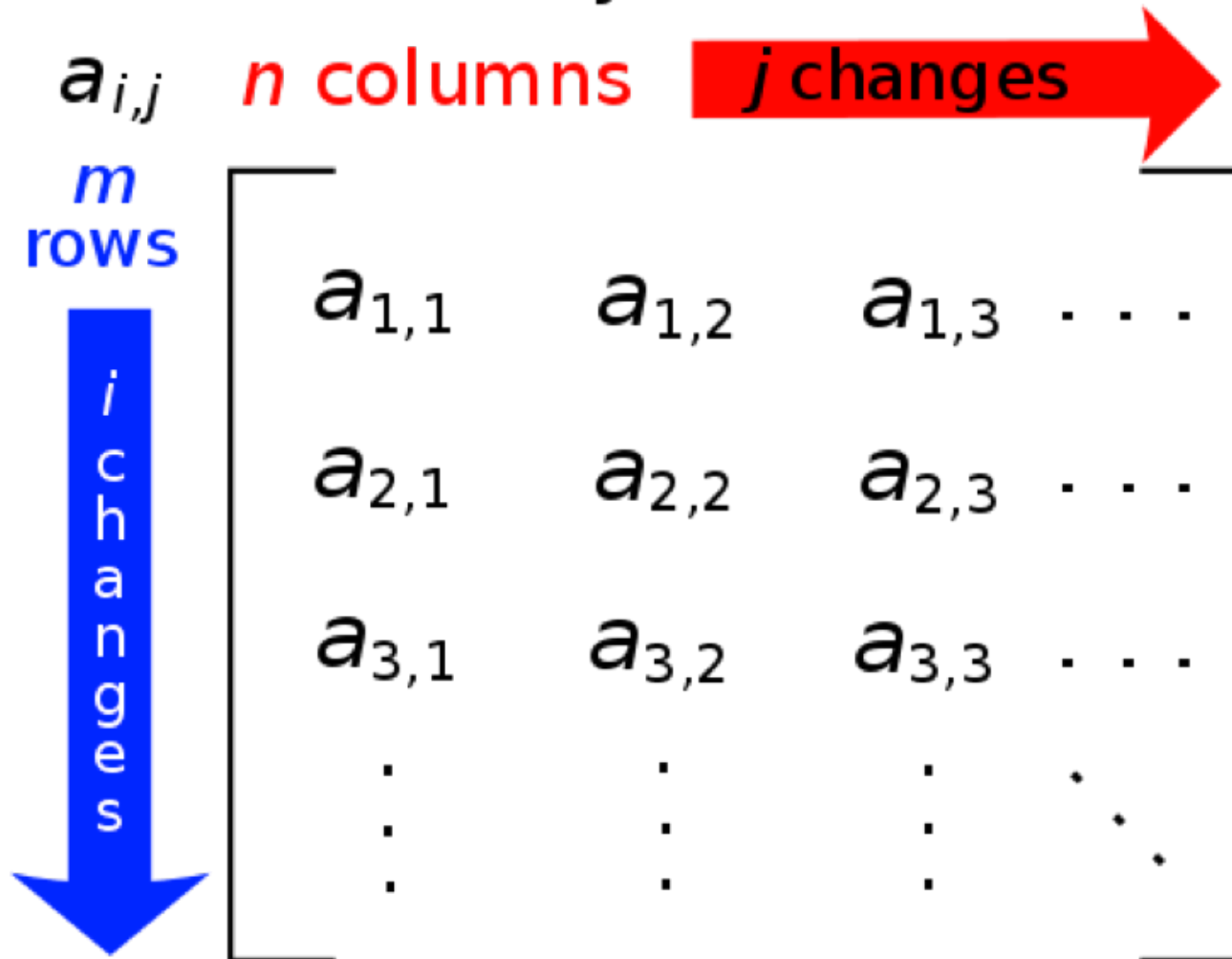
```
2
```

```
a.dtype.name
```

```
'int64'
```

Matrix

m -by- n matrix



NumPy ndarray: Multidimensional Array Object

NumPy ndarray

One-dimensional Array (1-D Array)

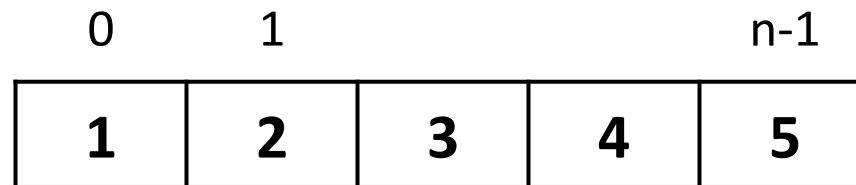
0	1			n-1
1	2	3	4	5

Two-dimensional Array (2-D Array)

	0	1		n-1	
0	1	2	3	4	5
1	6	7	8	9	10
	11	12	13	14	15
m-1	16	17	18	19	20

```
import numpy as np
a = np.array([1,2,3,4,5])
```

One-dimensional Array (1-D Array)



```
a = np.array([1,2,3,4,5])
a
```

```
array([1, 2, 3, 4, 5])
```

```
a = np.array([ [1,2,3,4,5],[6,7,8,9,10],[11,12,13,14,15],[16,17,18,19,20] ] )
```

Two-dimensional Array (2-D Array)

	0	1		n-1	
0	1	2	3	4	5
1	6	7	8	9	10
	11	12	13	14	15
m-1	16	17	18	19	20

```
a = np.array([[1,2,3,4,5],[6,7,8,9,10],[11,12,13,14,15],[16,17,18,19,20]])  
a
```

```
array([[ 1,  2,  3,  4,  5],  
       [ 6,  7,  8,  9, 10],  
       [11, 12, 13, 14, 15],  
       [16, 17, 18, 19, 20]])
```



```
import numpy as np
a = np.array([[0, 1, 2, 3],
              [10, 11, 12, 13],
              [20, 21, 22, 23]])
a
```

0	1	2	3
10	11	12	13
20	21	22	23

```
a = np.array ([[0, 1, 2, 3], [10, 11, 12, 13], [20, 21, 22, 23]])
```

```
a = np.array([[0, 1, 2, 3], [10, 11, 12, 13], [20, 21, 22, 23]])  
a
```

```
array([[ 0,  1,  2,  3],  
       [10, 11, 12, 13],  
       [20, 21, 22, 23]])
```

```
print(a.ndim)
```

```
2
```

```
print(a.shape)
```

```
(3, 4)
```

0	1	2	3
10	11	12	13
20	21	22	23

NumPy Basics: Arrays and Vectorized Computation

NumPy Array

axis 1

0

1

2

0

0,0

0,1

0,2

axis 0

1

1,0

1,1

1,2

2

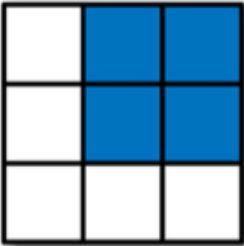

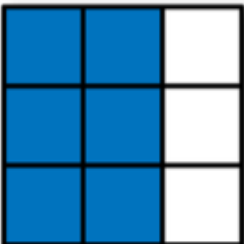
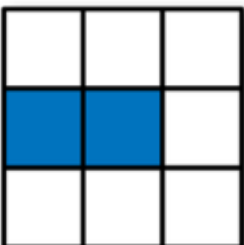
2,0

2,1

2,2

	0	1	2
0	0,0	0,1	0,2
1	1,0	1,1	1,2
2	2,0	2,1	2,2

Numpy Array


	Expression	Shape
	<code>arr[:2, 1:]</code>	<code>(2, 2)</code>
	<code>arr[2]</code> <code>arr[2, :]</code> <code>arr[2:, :]</code>	<code>(3,)</code> <code>(3,)</code> <code>(1, 3)</code>
	<code>arr[:, :2]</code>	<code>(3, 2)</code>
	<code>arr[1, :2]</code> <code>arr[1:2, :2]</code>	<code>(2,)</code> <code>(1, 2)</code>
















Wes McKinney (2017), "Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython", 2nd Edition, O'Reilly Media.

Materials and IPython notebooks for "Python for Data Analysis" by Wes McKinney, published by O'Reilly Media

52 commits 2 branches 0 releases 6 contributors

Branch: 2nd-edition ▾ New pull request Find file Clone or download ▾

 **betatim** committed with **wesm** Add requirements (#71) Latest commit ea47998 5 days ago

 datasets	Add Kaggle titanic dataset	5 months ago
 examples	Remove sex column from tips dataset	4 months ago
 .gitignore	Add gitignore	2 years ago
 COPYING	Use MIT license for code examples	a month ago
 README.md	Add launch in Azure Notebooks button (#70)	19 days ago
 appa.ipynb	Make more cells markdown instead of raw	a month ago
 ch02.ipynb	Make more cells markdown instead of raw	a month ago
 ch03.ipynb	Make more cells markdown instead of raw	a month ago
 ch04.ipynb	Convert all notebooks to v4 format	a month ago
 ch05.ipynb	Make more cells markdown instead of raw	a month ago
 ch06.ipynb	Make more cells markdown instead of raw	a month ago
 ch07.ipynb	Convert all notebooks to v4 format	a month ago
 ch08.ipynb	Make more cells markdown instead of raw	a month ago
 ch09.ipynb	Make more cells markdown instead of raw	a month ago
 ch10.ipynb	Make more cells markdown instead of raw	a month ago

<https://github.com/wesm/pydata-book>


Wes McKinney (2017), "Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython", 2nd Edition, O'Reilly Media.




wesm / pydata-book Watch 640 Star 4,031 Fork 4,348

[Code](#) [Issues 2](#) [Pull requests 0](#) [Projects 0](#) [Insights](#)

Branch: 2nd-edition **pydata-book / ch04.ipynb** Find file Copy path

wesm Convert all notebooks to v4 format c2780a0 on Sep 27

2 contributors 

1857 lines (1856 sloc) | 32.6 KB Raw Blame History   

NumPy Basics: Arrays and

```
In [ ]: import numpy as np
np.random.seed(12345)
import matplotlib.pyplot as plt
plt.rc('figure', figsize=(10, 6))
np.set_printoptions(precision=4, suppress=True)
```

```
In [ ]: import numpy as np
my_arr = np.arange(1000000)
my_list = list(range(1000000))
```

```
In [ ]: %time for _ in range(10): my_arr2 = my_arr * 2
%time for _ in range(10): my_list2 = [x * 2 for x in my_list]
```

The NumPy ndarray: A Multidimensional Array Object

```
In [ ]: import numpy as np
# Generate some random data
data = np.random.randn(2, 3)
data
```

Summary

- **Foundations of Big Data Analysis in Python**
 - **Python**
 - Programming language
 - **Numpy**
 - Scientific computing

References

- Wes McKinney (2017), "Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython", 2nd Edition, O'Reilly Media.
<https://github.com/wesm/pydata-book>
- Ties de Kok (2017), Learn Python for Research,
<https://github.com/TiesdeKok/LearnPythonforResearch>
- Avinash Jain (2017), Introduction To Python Programming, Udemy,
<https://www.udemy.com/pythonforbeginnersintro/>
- Python Programming, <https://pythonprogramming.net/>
- Python, <https://www.python.org/>
- Python Programming Language, <http://pythonprogramminglanguage.com/>
- Numpy, <http://www.numpy.org/>
- Pandas, <http://pandas.pydata.org/>
- Skikit-learn, <http://scikit-learn.org/>
- Data School (2015), Machine learning in Python with scikit-learn,
<https://www.youtube.com/playlist?list=PL5-da3qGB5ICeMbQuqbbCOQWcS6OYBr5A>
- Jason Brownlee (2016), Your First Machine Learning Project in Python Step-By-Step,
<https://machinelearningmastery.com/machine-learning-in-python-step-by-step/>
- Min-Yuh Day (2020), Python 101, <https://tinyurl.com/aintpupython101>