



HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

KHÓA HỌC

HỌC MÁY CHO AN TOÀN THÔNG TIN

PHẦN 1: CƠ BẢN VỀ HỌC MÁY VÀ HỌC SÂU

Giảng viên: TS. Nguyễn Ngọc Điệp

E-mail: diepnguyenngoc@ptit.edu.vn

Đơn vị: Khoa An toàn thông tin, Học viện Công nghệ BCVT

Nội dung

❖ Python và các thư viện học máy

- Khái quát về Python
- Một số thư viện học máy
 - NumPy
 - Scikit-Learn
 - Keras
 - TensorFlow

❖ Bài tập và mock project

Python trong học máy

❖ Sử dụng Python cho Học máy

- Python là ngôn ngữ lập trình được ưu tiên hàng đầu để phát triển các ứng dụng học máy vì tính linh hoạt của nó.
- Python hỗ trợ nhiều công cụ và gói khác nhau, cho phép các chuyên gia học máy triển khai các thay đổi với sự linh hoạt cao.
- Python là một ngôn ngữ kịch bản, dễ dàng học hỏi và lập trình.
- Tiện lợi khi triển khai các demo thử nghiệm

❖ Cài đặt

- Sử dụng Python 3.x
- Kiểm tra version:

```
import sys  
print ("Python version: { }",format(sys.version))
```

Python trong học máy

❖ Python Interactive Development Environment(IDE)

- Spyder:
 - Dễ dùng cho người mới bắt đầu, dễ debug, dễ xem dữ liệu
- Pycharm
 - Hỗ trợ người dùng chuyên nghiệp. Có phí/miễn phí.
- Visual Code Studio:
 - Hỗ trợ nhiều plugin
 - Hỗ trợ cả Jupyter notebook
- Jupyter
 - Môi trường tuyệt vời để phát triển và chia sẻ sổ tay Jupyter, Kết hợp mã, văn bản, hình ảnh liên mạch, Phù hợp cho chia sẻ và cộng tác
 - Không đầy đủ tính năng IDE, hạn chế cho dự án lớn

Spyder

The screenshot displays the Spyder IDE interface with the following components:

- Left Panel (File Explorer):** Shows the project structure with folders like 'plugin.py', 'chart_plot_example.py', and 'IPythonConsole'.
- Central Panel (Code Editor):** Displays the code for the 'Plots' plugin. The code includes comments, imports, and a class definition for 'Plots'.
- Right Panel (Variable Explorer):** Shows a table of variables and their values.
- Bottom Panel (Plots):** Displays a 3D surface plot and a polar plot.

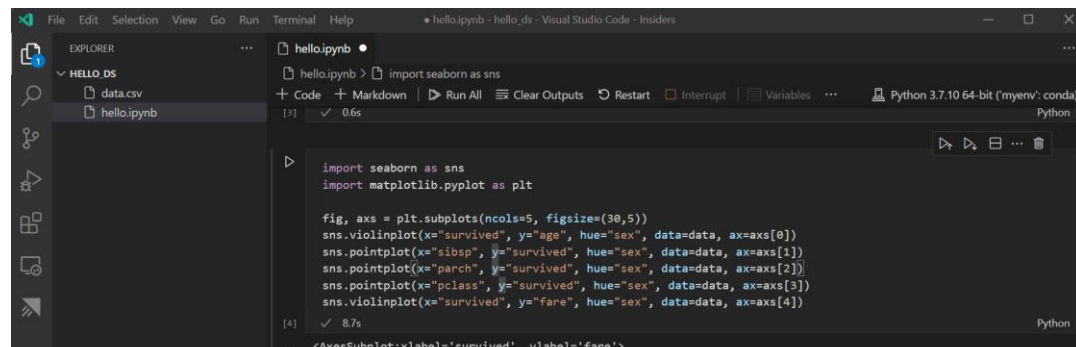
Variable Explorer Table:

Name	Type	Size	Value
bool	bool	1	True
data	Array of str128	(3, 3)	ndarray object of numpy module
datetime_object	datetime	1	2021-04-14 17:35:14.687085
df	DataFrame	(2, 2)	Column names: Col1, Col2
filename	str	53	/Users/juanitagomez/Local/Dev-Spyder/spyder/spyder/tests/test_dont_use.py
li	list	5	['abcd', 745, 2.23, 'efgh', 70.2]
myset	set	3	{'2', '1', '3'}
r	float	1	6.46567886443
t	tuple	5	('abcd', 745, 2.23, 'efgh', 70.2)
tinylist	list	2	[123, 'efgh']
x	float64	1	1.1235123099439

Code Editor Content (plugin.py - plots):

```
1  -*- coding: utf-8 -*-
2  #
3  # Copyright © Spyder Project Contributors
4  # Licensed under the terms of the MIT License
5  # (see spyder/_init_.py for details)
6
7  """
8  Plots Plugin.
9  """
10
11 # Third party imports
12 from qtpy.QtCore import Signal
13
14 # Local imports
15 from spyder.api.plugins import Plugins, SpyderDockablePlugin
16 from spyder.api.translations import get_translation
17 from spyder.plugins.plots.widgets.main_widget import PlotsWidget
18
19 # Localization
20 _ = get_translation('spyder')
21
22
23
24 class Plots(SpyderDockablePlugin):
25     """
26     Plots plugin.
27     """
28     NAME = 'plots'
29     REQUIRES = [Plugins.IPythonConsole]
30     TABIFY = [Plugins.VariableExplorer, Plugins.Help]
31     WIDGET_CLASS = PlotsWidget
32     CONF_SECTION = NAME
33     CONF_FILE = False
34     DISABLE_ACTIONS_WHEN_HIDDEN = False
35
36     # --- SpyderDockablePlugin API
37
38     def get_name(self):
39         return _('Plots')
40
41     def get_description(self):
42         return _('Display, explore and save console generated plots.')
43
44     def get_icon(self):
45         return self.create_icon('hist')
46
47     def register(self):
48         # Plugins
49         ipyconsole = self.get_plugin(Plugins.IPythonConsole)
50
51         # Signals
52         ipyconsole.sig_shellwidget_changed.connect(self.set_shellwidget)
53         ipyconsole.sig_shellwidget_process_started.connect(
54             self.add_shellwidget)
55         ipyconsole.sig_shellwidget_process_finished.connect(
56             self.remove_shellwidget)
```

Visual Code Studio



```

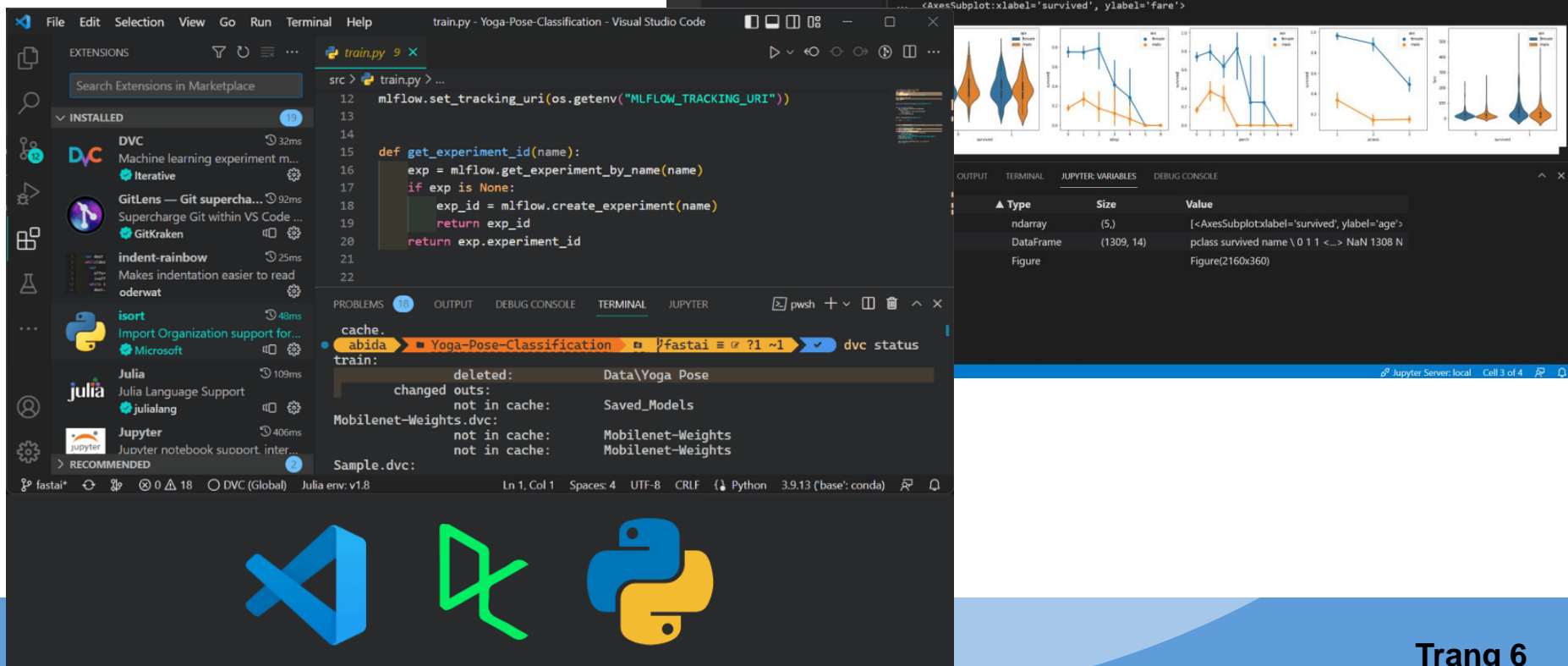
File Edit Selection View Go Run Terminal Help
hello.ipynb - hello_ds - Visual Studio Code - Insiders

EXPLORER
HELLO_DS
data.csv
hello.ipynb

hello.ipynb
import seaborn as sns
import matplotlib.pyplot as plt

fig, axs = plt.subplots(ncols=5, figsize=(30,5))
sns.violinplot(x="survived", y="age", hue="sex", data=data, ax=axs[0])
sns.pointplot(x="sibsp", y="survived", hue="sex", data=data, ax=axs[1])
sns.pointplot(x="parch", y="survived", hue="sex", data=data, ax=axs[2])
sns.pointplot(x="pclass", y="survived", hue="sex", data=data, ax=axs[3])
sns.violinplot(x="survived", y="fare", hue="sex", data=data, ax=axs[4])

<AxesSubplot: xlabel='survived', ylabel='fare'>
  
```



train.py - Yoga-Pose-Classification - Visual Studio Code

```

File Edit Selection View Go Run Terminal Help
train.py 9 x
src > train.py > ...
12 mlflow.set_tracking_uri(os.getenv("MLFLOW_TRACKING_URI"))
13
14
15 def get_experiment_id(name):
16     exp = mlflow.get_experiment_by_name(name)
17     if exp is None:
18         exp_id = mlflow.create_experiment(name)
19         return exp_id
20     return exp.experiment_id
21
22
  
```

PROBLEMS 10 OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

cache.
abida Yoga-Pose-Classification fastai ?1 ~1 dvc status
train:

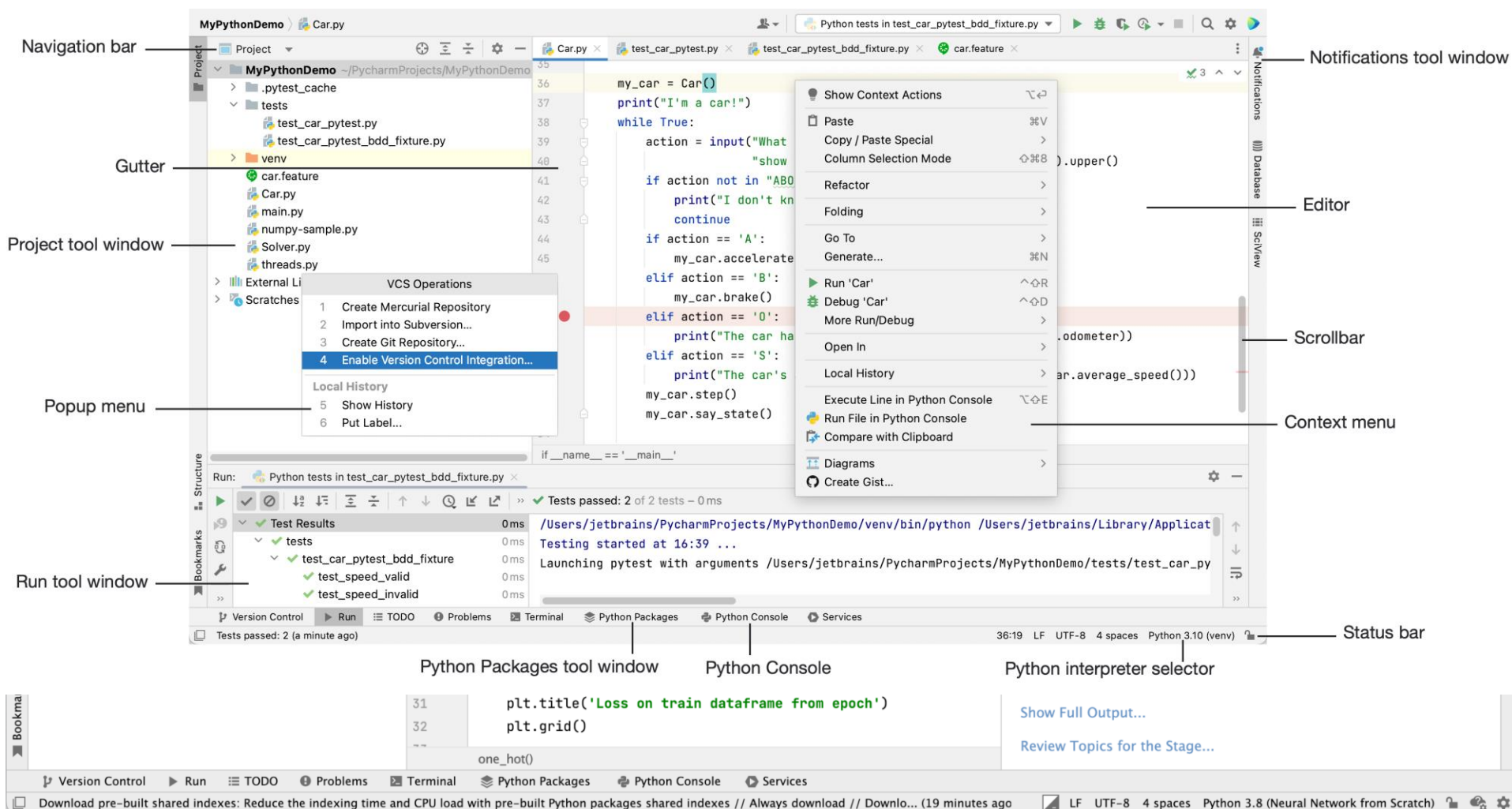
deleted:	Data\Yoga Pose
changed outs:	not in cache: Saved_Models
Mobilenet-Weights.dvc:	not in cache: Mobilenet-Weights
Sample.dvc:	not in cache: Mobilenet-Weights

OUTPUT TERMINAL JUPYTER-VARIABLES DEBUG CONSOLE

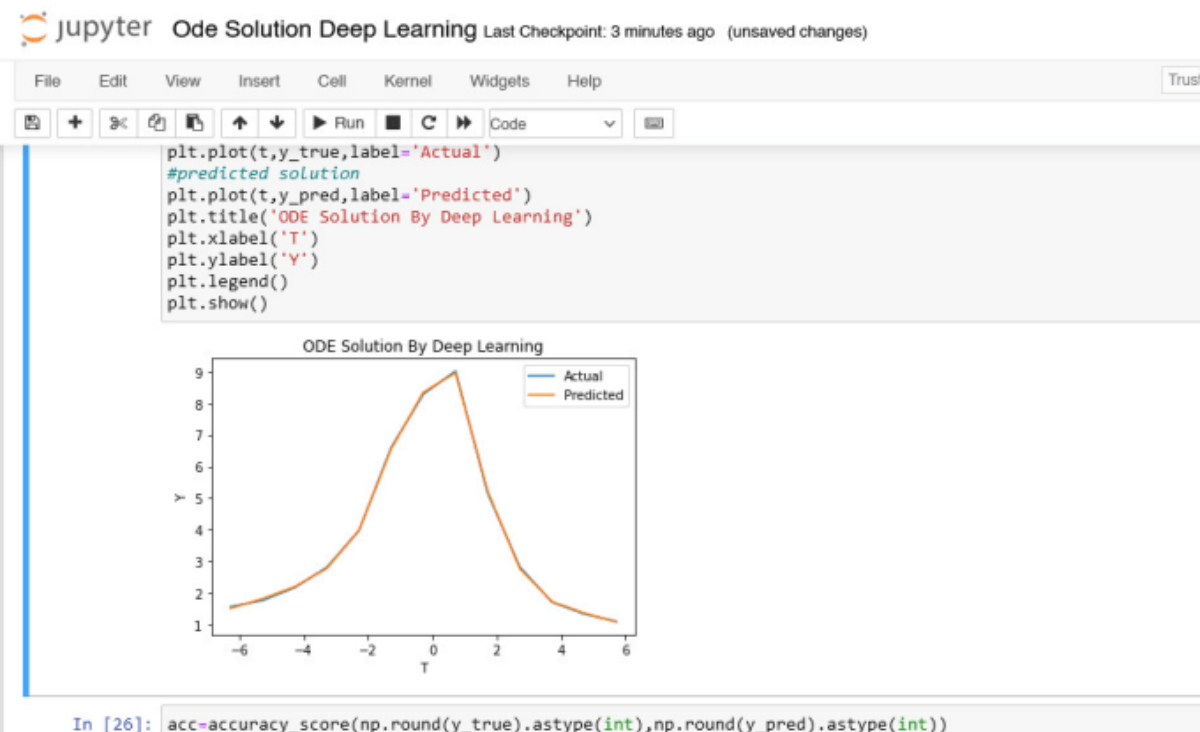
Type	Size	Value
ndarray	(5,)	[<AxesSubplot: xlabel='survived', ylabel='age'>
DataFrame	(1309, 14)	pclass survived name \ 0 1 1 <...> NaN 1308 N
Figure		Figure(2160x360)

fastai* 0 18 DVC (Global) Julia env: v1.8 Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python 3.9.13 (base: conda)

Pycharm



Jupyter notebook



In [18]: `from sklearn.model_selection import train_test_split`

`X_train, X_test, y_train, y_test = train_test_split(X_all, y_all, test_size=0.33, random_state=42)`

`# Show the results of the split`

`print ("Training set has {} samples.".format(X_train.shape[0]))`

`print ("Testing set has {} samples.".format(X_test.shape[0]))`

`print ("Anomaly rate of the training set: {:.2f}%".format(100 * (y_train == 1).mean()))`

`print ("Anomaly rate of the testing set: {:.2f}%".format(100 * (y_test == 1).mean()))`

Training set has 2019 samples.

Testing set has 995 samples.

Anomaly rate of the training set: 2.72%

Anomaly rate of the testing set: 2.91%

Cài đặt

❖ Jupyter notebook/Pycharm/Spyder

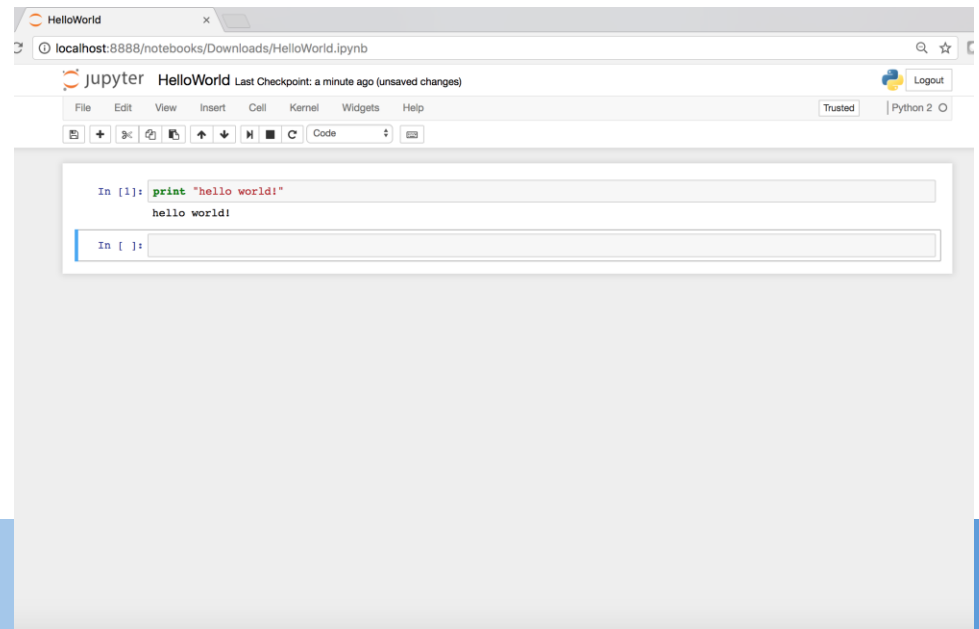
- Cài đặt Anaconda với 150 package mặc định, cùng hơn 250 open source package khác.
- <https://www.anaconda.com/download/>

❖ Nếu chỉ cần: Jupyter notebook

- `pip install --upgrade pip`
- `pip3 install jupyter`

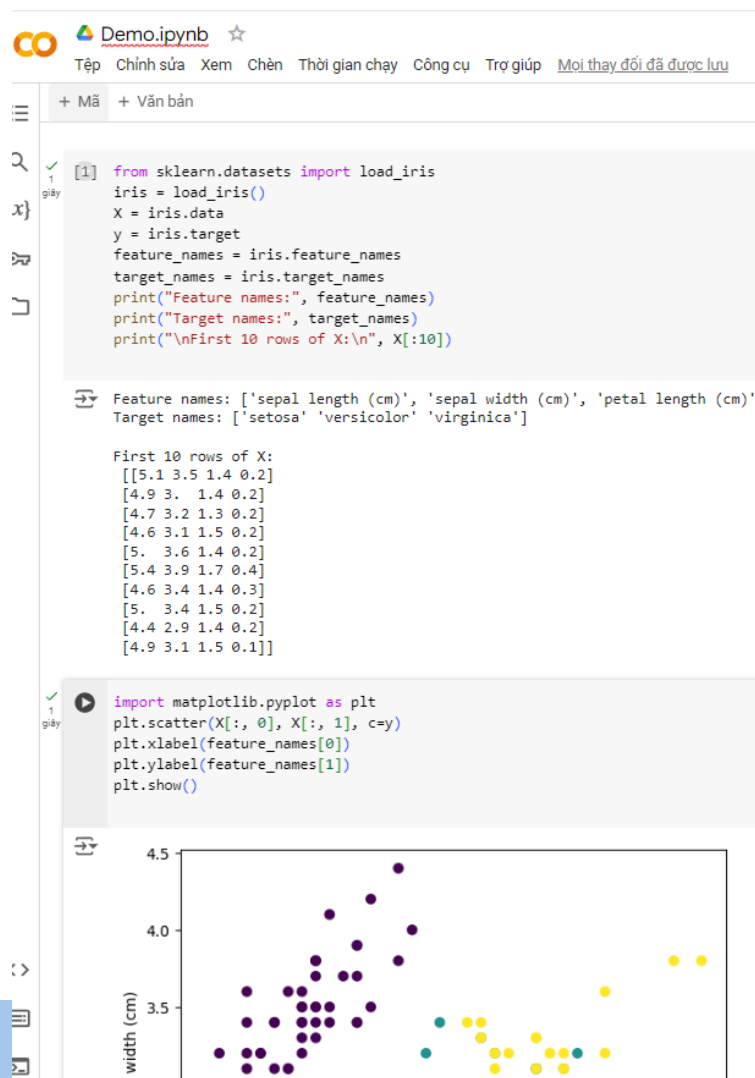
❖ Chạy:

- Gõ: `jupyter notebook`



Google Colab

- ❖ Các hệ thống tương tự Jupyter notebook dựa trên Cloud
- ❖ Tiện cho người dùng thử nghiệm
- ❖ Bất tiện cho các ứng dụng phức tạp



Nội dung

❖ Một số thư viện học máy

- NumPy
- Scikit-Learn
- Keras
- TensorFlow

NumPy

❖ NumPy:

- NumPy cực kỳ quan trọng trong phân tích thống kê hoặc học máy.
- NumPy chứa các hàm tính vi để giải quyết đại số tuyến tính, biến đổi Fourier và các phân tích số khác.
- NumPy có thể được cài đặt bằng cách chạy lệnh sau:

```
pip install numpy
```

- Để cài đặt NumPy thông qua Jupyter, sử dụng lệnh:

```
import sys
```

```
!{sys.executable} -m pip install numpy
```

scikit-learn - Thư viện Học máy của Python

❖ scikit-learn:

- Là một thư viện Python miễn phí được viết bằng Python, cung cấp các thuật toán học máy phổ biến cho các tác vụ phân lớp, phân cụm (clustering), hồi quy, v.v.
- Thư viện này đặc biệt hữu ích cho người mới bắt đầu học về học máy.
- scikit-learn đi kèm với một vài bộ dữ liệu được tích hợp sẵn, chẳng hạn như: Iris dataset, Breast cancer dataset, Boston price house, ...

❖ Cài đặt và thử chạy:

pip install scikit-learn

Python

```
import sklearn

if sklearn:
    print("scikit-learn đã được cài đặt thành công!")
else:
    print("Có lỗi xảy ra trong quá trình cài đặt.")
```

Pandas: Thư viện Phân tích dữ liệu

❖ Pandas:

- Pandas là một thư viện mã nguồn mở, mạnh mẽ trong Python, cung cấp các cấu trúc dữ liệu hiệu quả để thao tác và phân tích dữ liệu.
- Dễ sử dụng:
 - Pandas cung cấp các cấu trúc dữ liệu trực quan và dễ sử dụng như DataFrame và Series, giúp đơn giản hóa việc làm việc với dữ liệu dạng bảng.
 - Hỗ trợ nhiều định dạng dữ liệu: đọc và ghi dữ liệu từ CSV, Excel, SQL, JSON, v.v.
 - Linh hoạt với nhiều kiểu dữ liệu: DataFrame có thể lưu trữ các cột với các kiểu dữ liệu khác nhau (số nguyên, số thực, chuỗi, v.v.) trong cùng một cấu trúc
 - Cung cấp nhiều hàm xử lý dữ liệu: lọc, nhóm, kết hợp, chuyển đổi và thao tác với dữ liệu theo nhiều cách khác nhau
 - Tích hợp với các thư viện khác: tương thích với NumPy, scikit-learn, Matplotlib, v.v.

Pandas: Thư viện Phân tích dữ liệu

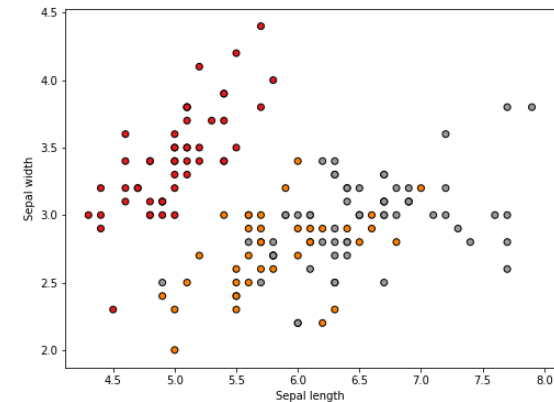
❖ Vai trò của Pandas trong Học máy:

- Pandas đóng vai trò quan trọng trong học máy bằng cách giúp xử lý và chuẩn bị dữ liệu trước khi đưa vào các thuật toán học máy.
- Các tác vụ phổ biến bao gồm:
 - Đọc và tải dữ liệu từ các nguồn khác nhau
 - Kiểm tra và xử lý dữ liệu thiếu, ngoại lai, và các lỗi khác
 - Chuyển đổi và chuẩn hóa dữ liệu để phù hợp với yêu cầu của các thuật toán học máy
 - Khám phá và tóm tắt dữ liệu để hiểu các đặc điểm của nó
 - Chuẩn bị dữ liệu thành các tập hợp huấn luyện và kiểm thử cho các mô hình học máy

matplotlib

❖ Thư viện matplotlib:

- tạo các biểu đồ và hình ảnh hóa dữ liệu hai chiều (2D)
- hỗ trợ nhiều loại biểu đồ khác nhau, chẳng hạn như đường (line), cột (bar), tán (scatter), miền (area), hình tròn (pie), histogram, v.v.
- khả năng kiểm soát chi tiết hầu hết các khía cạnh của biểu đồ, bao gồm nhãn, chú thích, định dạng trục, màu sắc, kiểu đường, v.v.
- tương thích và hoạt động hiệu quả với các thư viện khoa học dữ liệu phổ biến khác trong Python như Pandas, NumPy, scikit-learn, v.v.
- có thể được sử dụng trong nhiều môi trường khác nhau, chẳng hạn như Jupyter Notebook



Các thư viện trong học sâu

- ❖ Keras
- ❖ Tensorflow
- ❖ Pytorch
- ❖ ...

Bài tập: Thử nghiệm các thư viện

- ❖ Thử nghiệm các môi trường IDE khác nhau
- ❖ Ứng dụng trong tiền xử lý dữ liệu:
 - Numpy, Pandas (theo Hướng dẫn thực hành)

Bài tập cuối Phần 1

- ❖ Chọn 1 tập dữ liệu /bài toán trong thực tế
- ❖ Thử nghiệm các kiến thức học máy cơ bản để xây dựng mô hình, tối ưu hiệu năng, với các bước:
 - Xác định bài toán học máy phù hợp
 - Phân tích, khai phá dữ liệu
 - Thử nghiệm mô hình với các kĩ thuật đã học
 - Thử train toàn bộ tập dữ liệu với mô hình, đánh giá khả năng mô hình, độ phù hợp của dữ liệu
 - Train/test và đánh giá mô hình
 - K cross validation tốt hơn train test split như nào?
 - Thử nghiệm các phương pháp tiền xử lý dữ liệu. Phân tích, đánh giá khả năng áp dụng từng kĩ thuật.
 - Thử nghiệm kĩ thuật lựa chọn đặc trưng và đưa ra tập đặc trưng
 - Thử nghiệm các phương pháp xử lý dữ liệu mất cân bằng (nếu dữ liệu mất cân bằng)
 - Thử nghiệm các độ đo phân lớp
 - Thử nghiệm các mô hình khác trên tập dữ liệu
 - Gridsearch để tự động tìm tham số tối ưu cho các mô hình?
- ❖ Thử nghiệm tương tự với mô hình học sâu