國立清華大學
NATIONAL TSING HUA UNIVERSITY

AIMS - deep learning

Google Colab & Python Introduction

TA course 01

TAs: 蘇時頤, 廖柄淦

- Course website: https://github.com/matteosoo/aimsfellows_DL

Catalog

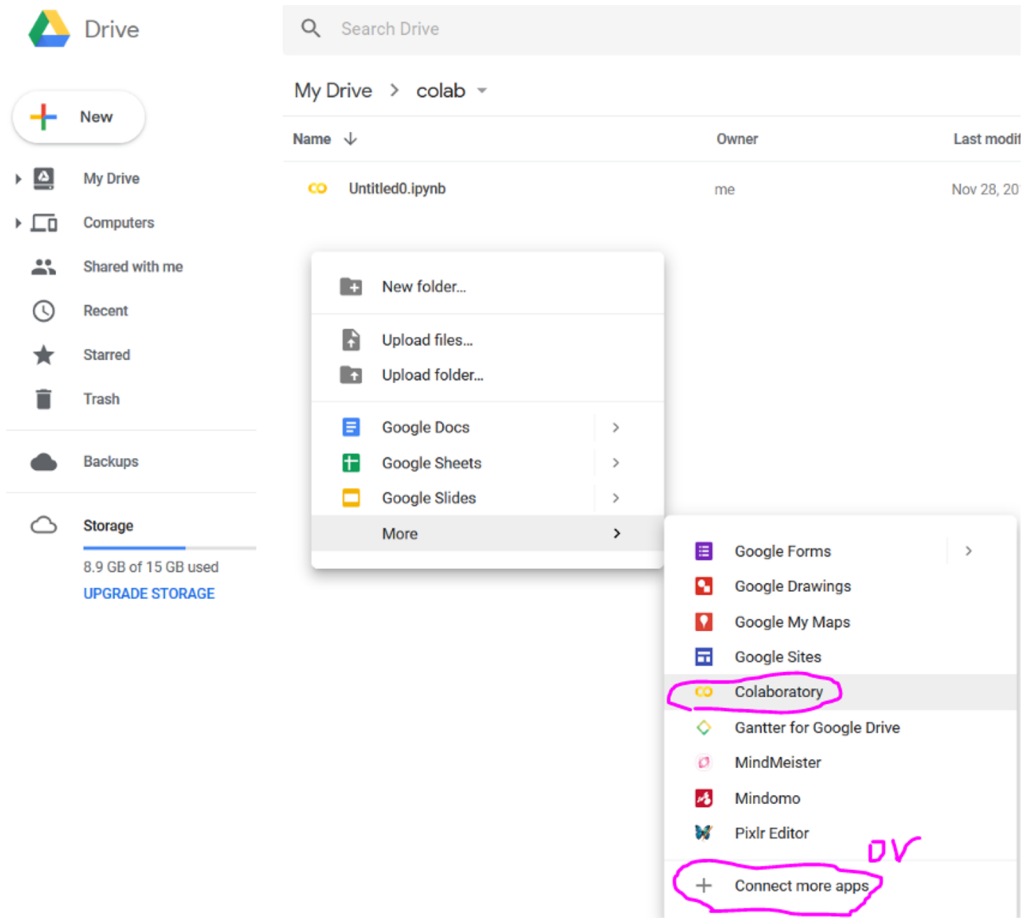
- Google Colab
  - Installation
  - Manipulation
  - Mount
  - Others..
- Basic Python programming
  - Python intro.
  - 變數與資料型態
  - if / else 條件式
  - list 陣列
  - for loop, while loop, break

Google Colab

Free GPUs (Graphics Processing Units) platform tutorial

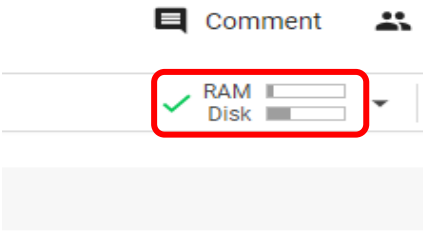
- Installation

- Login your google drive
- Create a new Colab file
  - Right click at empty place, in the menu: More > Colaboratory
  - If not found this option: More > Connect more apps > search and add Colaboratory

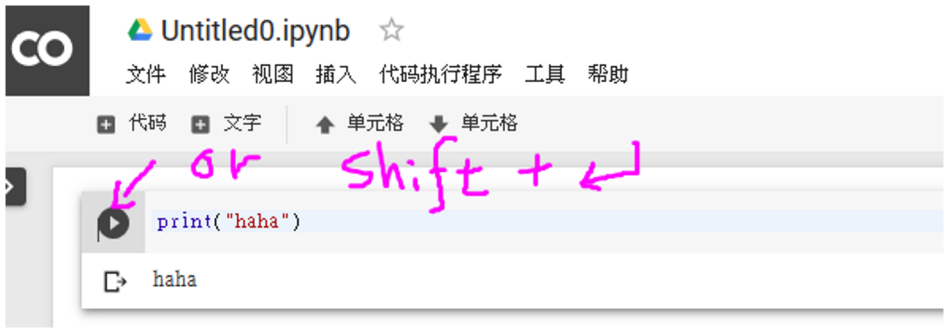


- Manipulation

- Check whether connect successfully



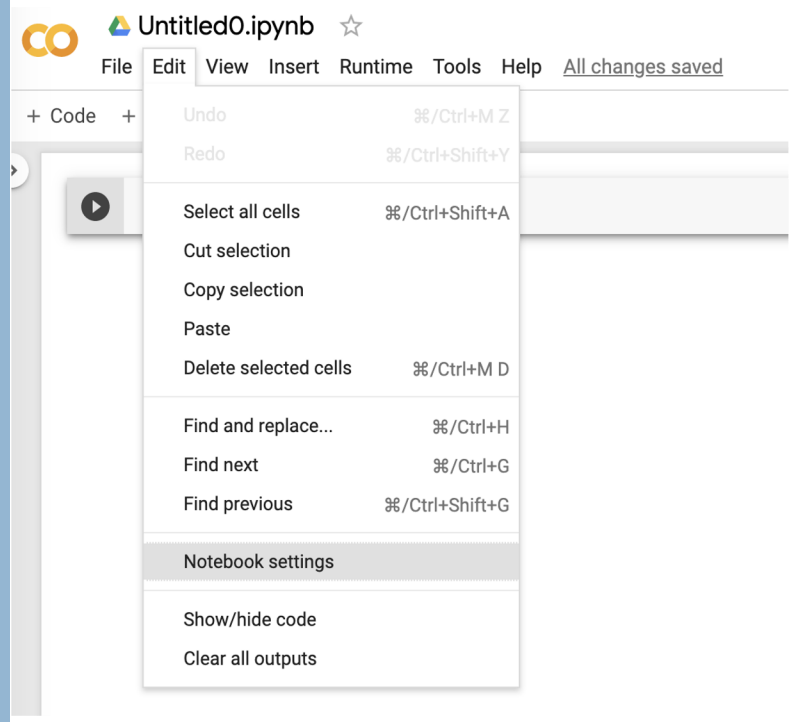
- Here you could input some code, then press the play button or shift + enter to execute the code.

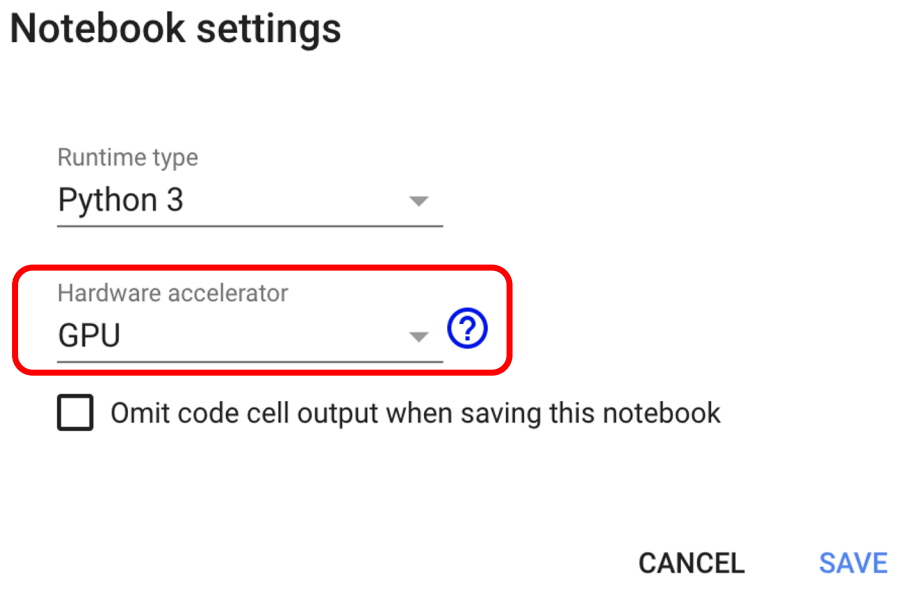


- You could just consider it as a cloud Jupyter Notebook with TensorFlow in your google drive.

- Select GPU to Train Model

- Edit > Notebook setting
  - Select Hardware accelerator -> GPU





- Constraint of the GPU (TPU) resource

	standards	details
GPU	Tesla P100 PCIE 16GB 或 Tesla K80 (可以用!nvidia-smi確認是使用哪一張卡)	提供至少50G儲存空間和12GB的記憶體，使用時間僅可連續12小時(超過時間便會清掉使用中內容)。
TPU	Designed by Google.	TPU runtime environment 不支援 local filesystem (也就是不能mount到GoogleDrive)，只能用 cloud storage，所以必須要用 GCP account 開 GCS bucket (Google Cloud Storage)。

- Install some packages on Colab

- For example, install a package – ***pandas***:

just input:

```
!pip install {package name}
```

then execute.

ex. install pandas:

```
[12] !pip install pandas

Requirement already satisfied: pandas in /usr/local/lib/python3.6/dist-packages (0.22.0)
Requirement already satisfied: pytz>=2011k in /usr/local/lib/python3.6/dist-packages (from pandas) (2018.7)
Requirement already satisfied: numpy>=1.9.0 in /usr/local/lib/python3.6/dist-packages (from pandas) (1.14.6)
Requirement already satisfied: python-dateutil>=2 in /usr/local/lib/python3.6/dist-packages (from pandas) (2.5.3)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-packages (from python-dateutil>=2->pandas) (1.11.0)
```

- Mount the cloud storage to Colab

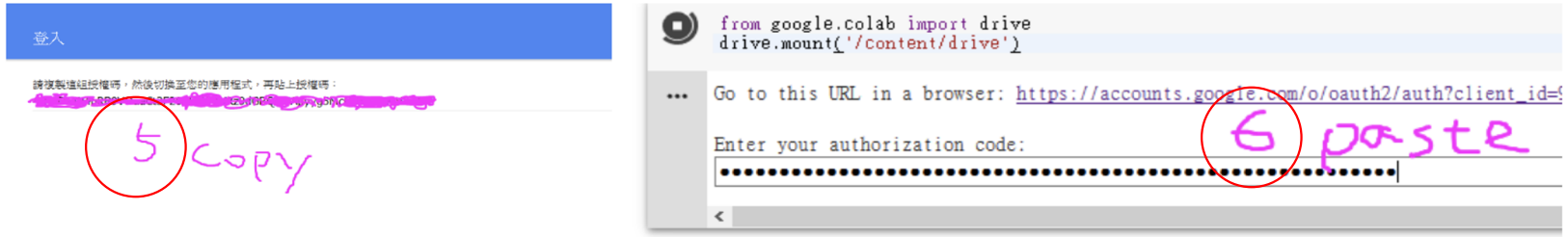
- Mount the Google Drive folder to Colab as follow:

1. Input this and execute

```
from google.colab import drive
drive.mount('/content/drive')
```

2.~4. follow the picture

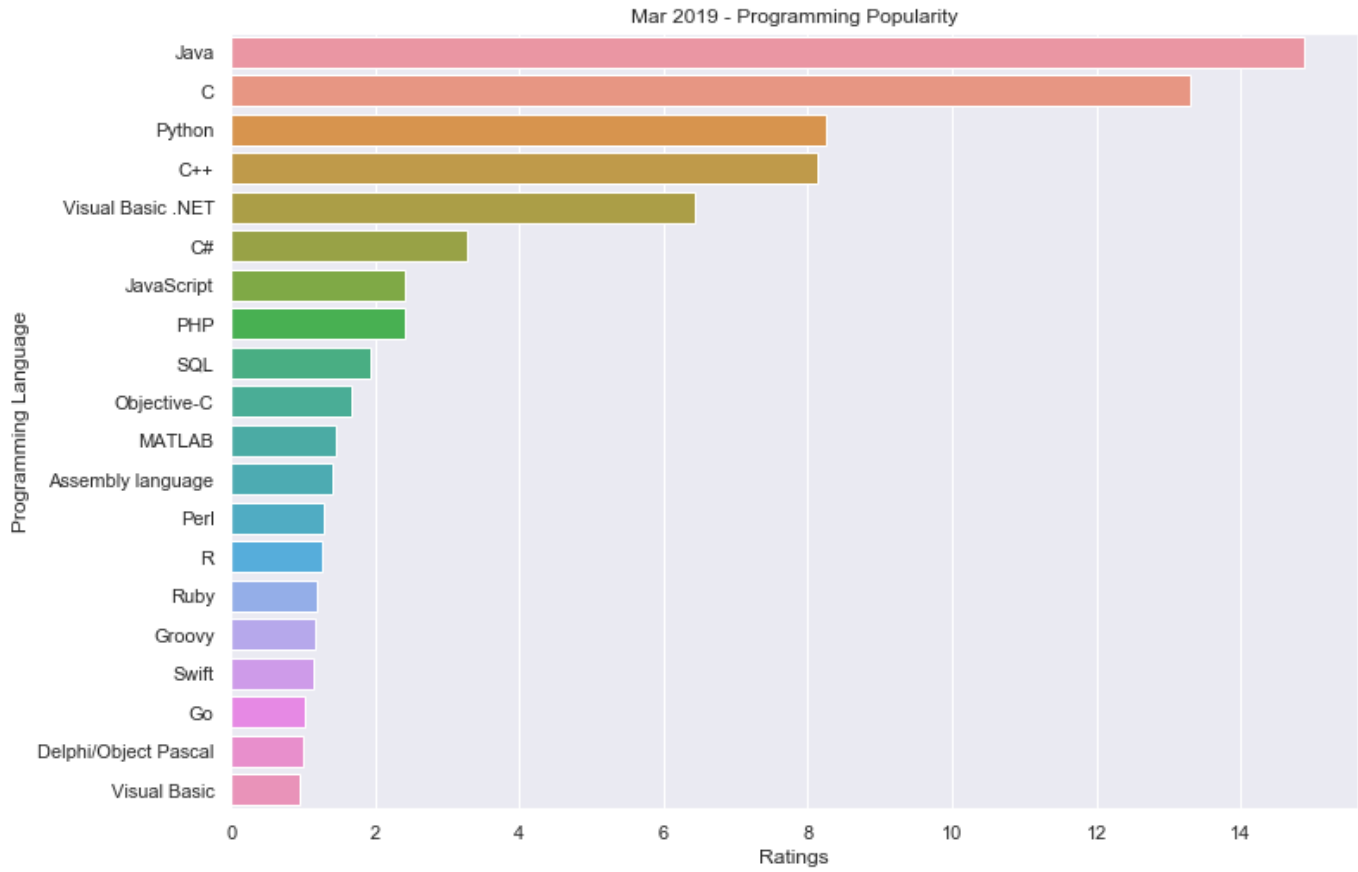
5.&6. **Paste auth code** to colab and **press enter**



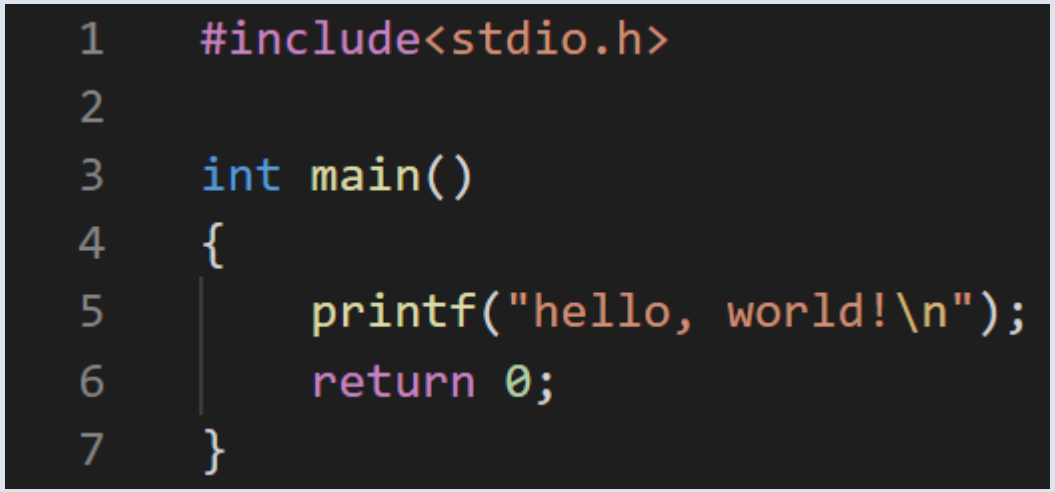
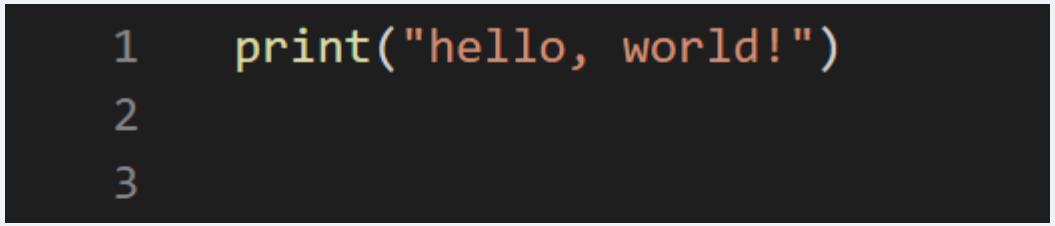
Basic Python programming

- Why Python?

- 學習成本低，初學者容易上手
- 大量函式庫 (library)，節省時間
- 用途廣:
  - Web design, web crawler(爬蟲), artificial intelligence (especially for deep learning), test automation



- Python特性

Preview in IDE* (*Integrated Development Environment)	
	C語言 • 語法複雜 • 功能強大且效能高
	Python • 語法簡潔容易閱讀 • 效能相對其他語言較差

- print()

- What's different?

```
print('hello world!')
```

```
print(1234)
```

- 變數與資料型態

- 數字(Number)
- 字串(String)
- 布林(Boolean)
- 陣列(List)
- 元組(Tuple)
- 集合(Set)
- 字典(Dictionary)

- 字串

- 單引號或雙引號:
  - 'I have a dream.'
  - "I'm lovin' it."
  - 'I'm lovin' it.' (X)
- 跳脫特殊字元 (\)
  - 'I\'m lovin\' it.' (V)

- Assign

- assign (指派變數)

```
x = 5
```

```
name = "David"
print(name)
```

```
price = 120
print('價錢: ', price)
```

```
價錢:  120
```

```
price = 120
name = 'Tommy'
print(name, '需要付', price, '元')
```

```
Tommy 需要付 120 元
```

-四則運算

```
x = 5
y = 3
z = x + y
```

```
x = 5
y = 3
x + y = z
```

```
  File "<ipython-input-5-7cd33024320f>", line 3
    x + y = z
          ^
SyntaxError: can't assign to operator
```

```
a = 5
a = a + 1
print(a)
```

```
b = 10
b += 1
print(b)
```

```
11
```

-四則運算

```
coffee = 40
tea = 30
price = coffee + tea
```

```
print(2*4)
print(10/3)
print(10//3)
print(10%3)
```

```
8
3.3333333333333335
3
1
```

- Boolean (布林)

- Equal vs assign

- Boolean
  - TRUE (not 0) / FALSE (0)

- Equal vs assign
  - 數學上的等於: ==
  - 指派變數(assign): =

Example:

```
>>> 1 < 2
True

>>> 5 >= 8 + 9
False

>>> '8' + '9' == '17'
False

>>> 'Z' > 'B'
True

>>> not 1>2
True
```

- if / else 條件式

- Example:

```
if salary > spend :
    print("了不起，負責!")

elif salary == spend:
    print("剛好花完!")

else:
    print("買不起!")
```

- Note: ⟷ 請按下"tab"，因為Python 對於縮排 (indent)非常嚴謹！
- Princeple:
  不管使用 if, elif, else, for, while ，當在 ":" 之後的下一行就必須縮排。

- list 陣列

- Example 1 :

```
subject = ['history', 'math', 'science']

lucky_numbers = [5, 12, 20, 49]

student = []
```

- Example 2 :

```
food = ['beef_noodle', 75, 'dumpling', 50]

drinks = [['latte', 'cappuccino'], 'tea', 'coke']
```

-陣列索引 index

```
                0          1          2          3          4          5
subject = ['history', 'math', 'science', 'English', 'geography', 'PE']

print(subject[2:5])
print(subject[1:])
print(subject[:3])
```

```
['science', 'English', 'geography']
['math', 'science', 'English', 'geography', 'PE']
['history', 'math', 'science']
```

```
                -3,        -2,        -1
subject = ['history', 'math', 'science']

print(subject[-1])
print(subject[-2])
print(subject[-3])
```

```
science
math
history
```

- 陣列操作: append, remove

```
subject = ['history', 'math', 'science']
print(subject)

subject.append('English')
print(subject)

subject.remove('math')
print(subject)
```

```
['history', 'math', 'science']
['history', 'math', 'science', 'English']
['history', 'science', 'English']
```

- for loop 迴圈

- **for i in range(**要執行的次數**):**

  要執行的程式

- 執行很多次相同或類似的運算、操作
- 過程中變數可能會改變 (ex: 計數)

- for i in range()

- Example 1:

```
for i in range(5):
    print(i)
```

```
0
1
2
3
4
```

```
for i in range(2, 7):
    print(i)
```

```
2
3
4
5
6
```

- Example 2:

```
subject = ['history', 'math', 'science']

for i in subject:
    print(i)
```

```
history
math
science
```

- while loop

- **while 條件式:**
  要執行的程式

- 條件式為真(True)時: 無限循環
- 已知執行次數，適合使用 **for**
- 未知執行次數，適合使用 **while**

- while loop example

- Example:

```
number = 0
while number < 100:
    print(number)
    number = number + 10
```

```
0
10
20
30
40
50
60
70
80
90
```

- break 中斷

- Example:

```
number = 0
while number < 5:
    print(number)
    number = number+1
```

```
0
1
2
3
4
```

```
number = 0
while True:
    print(number)
    number = number+1
    if number==5:
        break
```

```
0
1
2
3
4
```