

# Missing Semester for ML

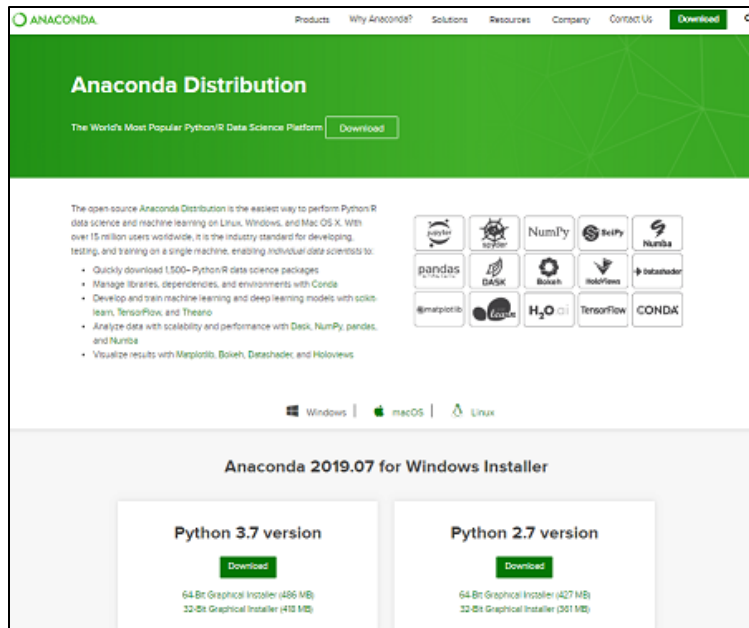
---

## I. 구글 클라우드 개발환경 코랩



# 아나콘다 개발환경

- 머신 러닝 실습을 하기 위해서는 많은 패키지가 필요합니다. 이를 일일이 설치하는 것보다는 필요한 **패키지들을 모아놓은 파이썬 배포판 '아나콘다'를 설치**하는 것을 권장합니다.
- 아나콘다는 Numpy, Pandas, Jupyter Notebook, IPython, scikit-learn, matplotlib, seaborn, nltk 등 이 책에서 사용할 대부분의 패키지를 전부 포함하고 있습니다.



➔ (상대적으로) GPU 설정 복잡

➔ (상대적으로) 설치 시간 오래 걸림

*\*<https://www.anaconda.com/distribution/>*

# 구글 코랩 개발 환경

- 구글 코랩(Colab)은 클라우드 기반의 **무료** Jupyter 노트북 개발 환경임
- 내부적으로는 코랩 + 구글드라이브 + 도커 + 리눅스 + 구글 클라우드의 기술스택으로 이루어져 있음

내 PC도 좋은데.. 굳이 써야 되나요?

장점	주의사항
<ul style="list-style-type: none"><li>○ 공짜다.</li><li>○ 어지간한 개인 PC보다 성능이 좋고 빠르다.</li><li>○ 쉽다.</li><li>○ GPU를 지원하다. 무려 12G~16G !!</li><li>○ 환경설정 및 구동 준비가 5분이면 끝난다.</li><li>○ 클라우드 기반이다.</li><li>○ 모바일도지원된다.</li><li>○ 학습 및 공유에 최고! (Github, 등)</li></ul>	<ul style="list-style-type: none"><li>○ 최대 세션 유지시간은 12시간이다.</li><li>○ 세션이 끊기면? 작업중이던 데이터가 사라진다.</li><li>○ 금융권 등 망 분리 보안 이슈로 법적으로 클라우드에 데이터를 올릴 수 없는 경우는 사내에서 활용하기 어렵다.</li></ul>

# 클라우드 개발환경 비교

## ■ 구글 vs 마이크로소프트

	구글 Colab	마이크로소프트 Azure
장점	<ul style="list-style-type: none"><li>▪ 다른 서비스들이 제공하지 않는 GPU 자원 할당</li><li>▪ 구글 드라이브를 직접 마운트 사용</li><li>▪ 어디서나, 언제나 접근해서 사용 가능, 간결함</li><li>▪ 상대적으로 빠른 속도</li></ul>	<ul style="list-style-type: none"><li>▪ Jupyter Notebook 그대로의 UI</li><li>▪ 언제 어디서든 접근하여 실행할 수 있는 편리함</li><li>▪ Notebook의 관리와 Github와의 연계성</li><li>▪ MS가 제공하는 다양한 강좌들이 포함되어 있어 학습하기에 최적의 장소</li></ul>
단점	<ul style="list-style-type: none"><li>▪ UI가 클래식함</li></ul>	<ul style="list-style-type: none"><li>▪ Preview 버전으로써, 현업 개발로 사용하기에는 다소 느림</li><li>▪ 메모리 할당량이 낮음 (3.8G)</li></ul>

# 코랩 실습 환경 세팅

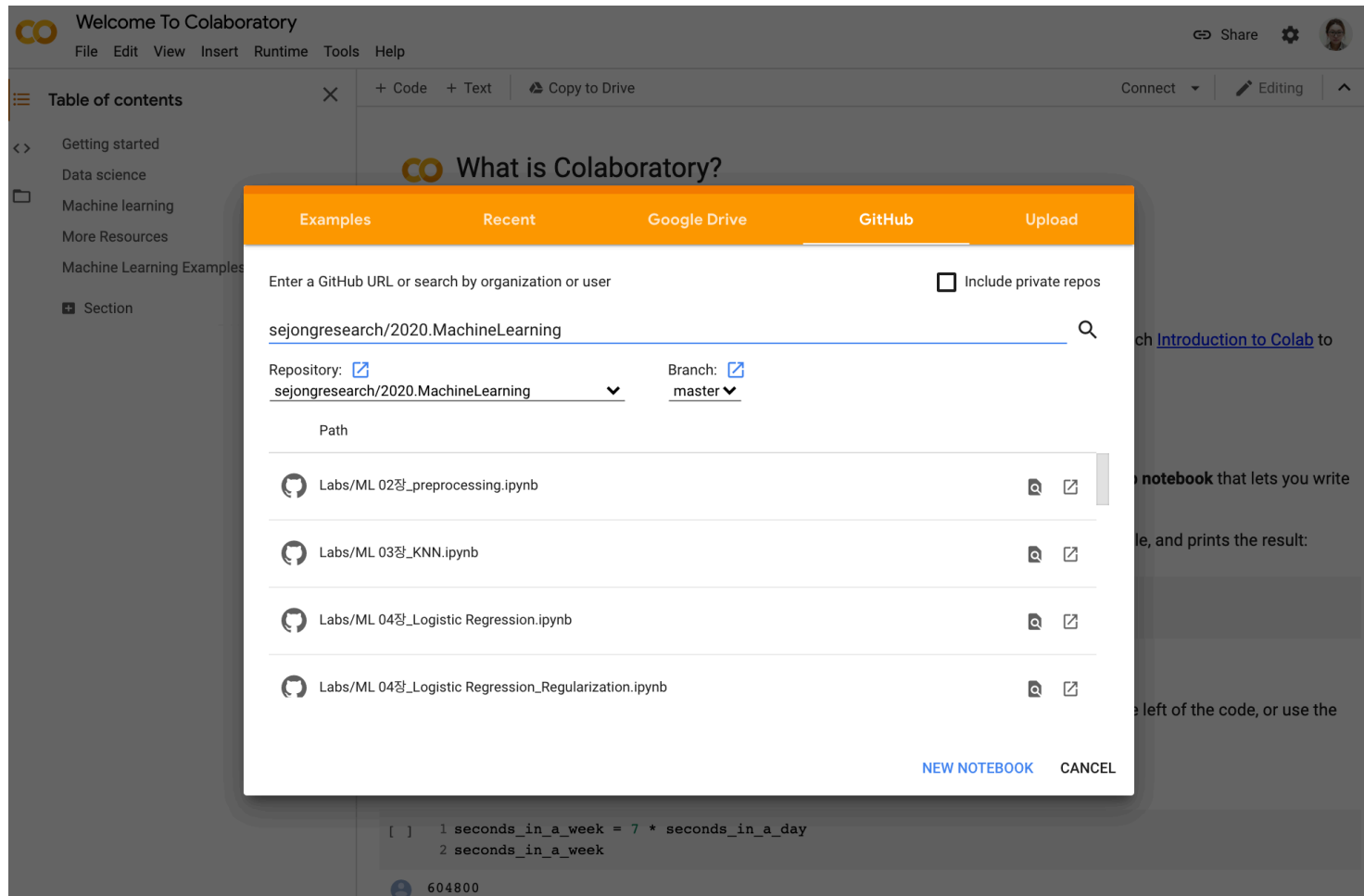
- 코랩 공식 링크: <https://colab.research.google.com>



<https://www.youtube.com/watch?v=inN8seMm7UI&feature=youtu.be>

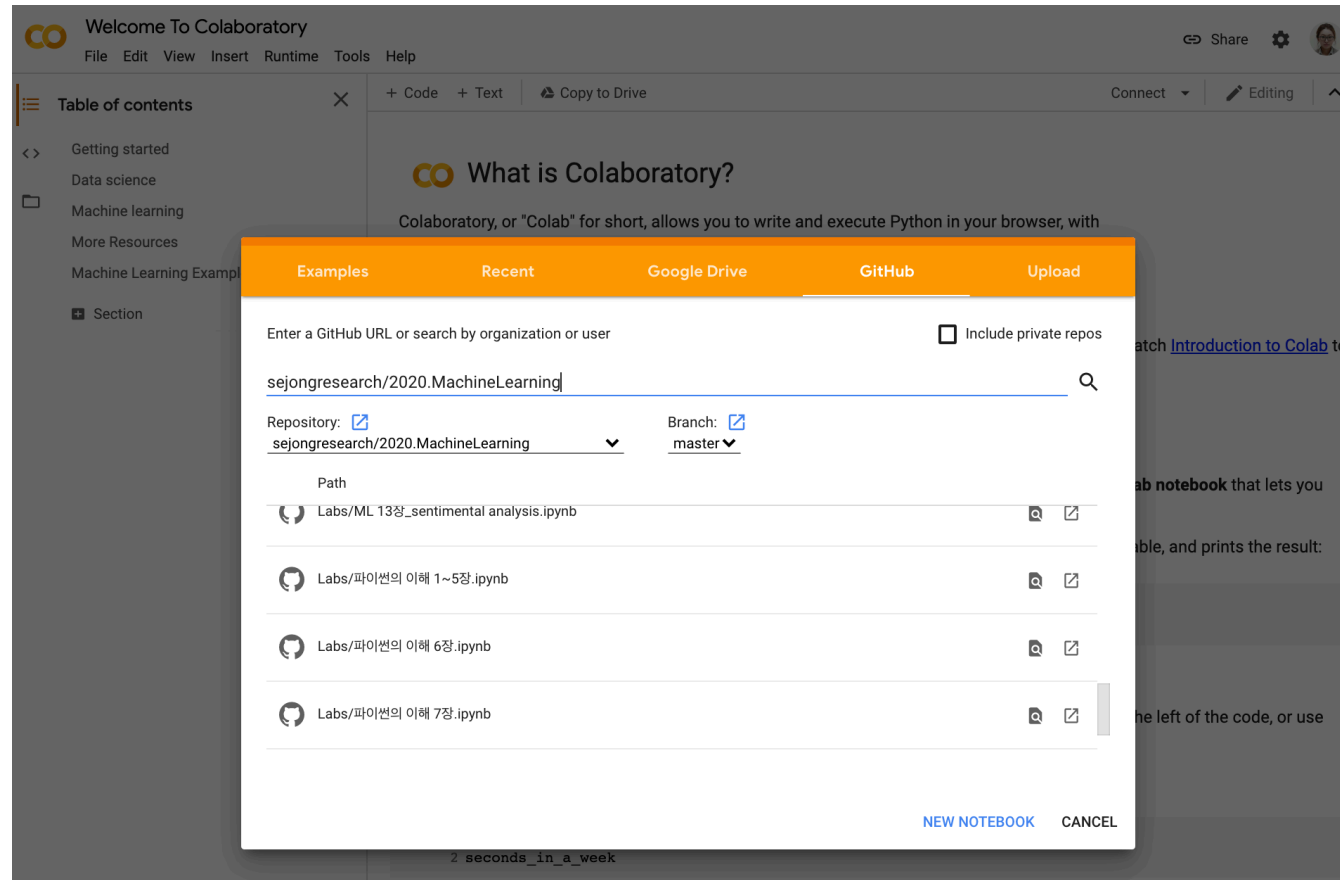
# 코랩 실습 환경 세팅

- 파일 → 노트 열기 → GitHub탭 → sejongresearch/2020.MachineLearning




# 코랩 실습 환경 세팅

- Labs/파이썬의 이해 1~5장.ipynb 열기



# 코랩 실습 환경 세팅

- [Copy to Drive] 클릭
  - 반드시 사용자의 개인 계정 구글 드라이브로 복사 후 실행해야 함

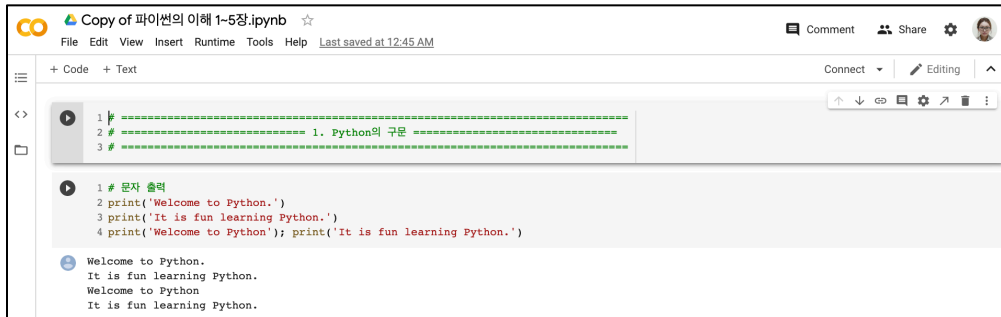


The screenshot displays the Google Colab web interface. At the top, the file name is '파이썬의 이해 1~5장.ipynb'. The menu bar includes 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help'. Below the menu, there are buttons for '+ Code', '+ Text', and 'Copy to Drive', with the latter being highlighted by a red rectangular box. To the right of these buttons are 'Connect', 'Editing', and a user profile icon. The main workspace contains three code blocks. The first block is a comment: `1 # =====`  
`2 # ===== 1. Python의 구문 =====`  
`3 # =====`. The second block contains Python code: `1 # 문자 출력`  
`2 print('Welcome to Python.')`  
`3 print('It is fun learning Python.')`  
`4 print('Welcome to Python'); print('It is fun learning Python.')`. Below this code, the output is shown: `Welcome to Python.`  
`It is fun learning Python.`  
`Welcome to Python`  
`It is fun learning Python.`. The third block contains Python code: `1 # 줄 바꾸기`  
`2 sum = 10 + 20 + \`  
`3 30 + 40 + \`  
`4 30`  
`5 sum`  
`6`. Below this code, the output is shown: `130`. The bottom block contains Python code: `1 # 블록(Block), 들여쓰기(Indentation)`  
`2 if 10>3:`  
`3 print('This is inside block')`  
`4 print('This is outside of block')`. Below this code, the output is shown: `This is inside block`  
`This is outside of block`.



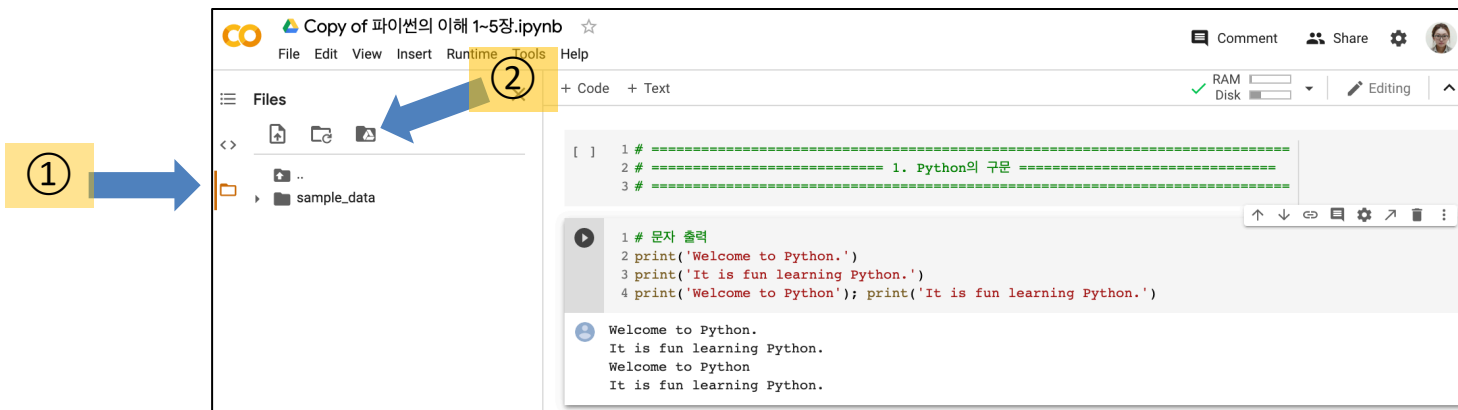
# 코랩 사용법

- Cell 실행하는 법
  - [Play 버튼 클릭] 혹은 [Shift + Enter]



- Google Drive 연동하는 법 (\*개인 클라우드 계정을 사용해야 파일 영구 저장됨)

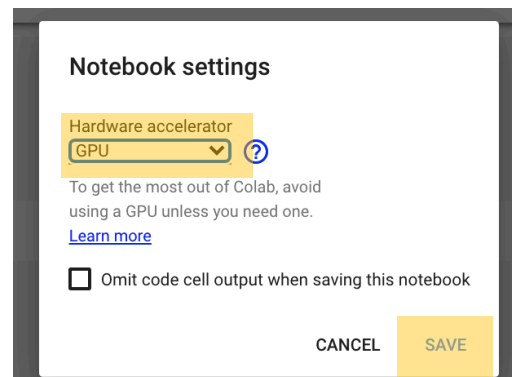
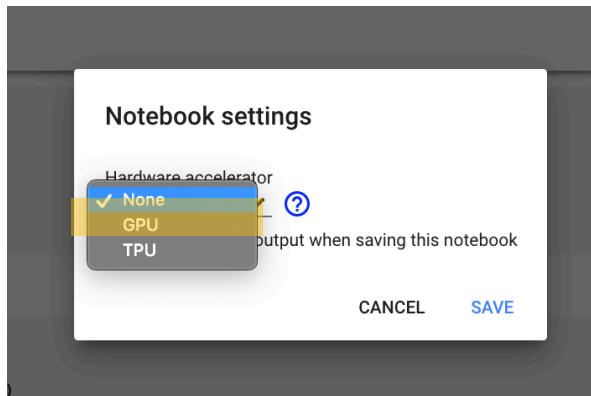
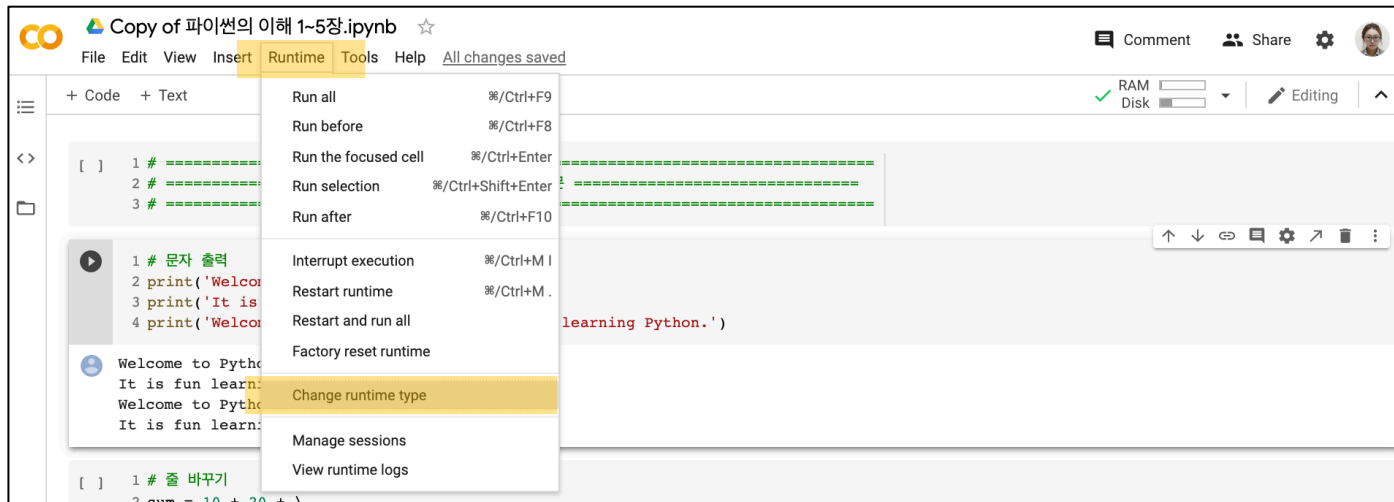
- ① → ② 클릭



# 코랩 사용법

- GPU 사용을 위한 런타임 설정

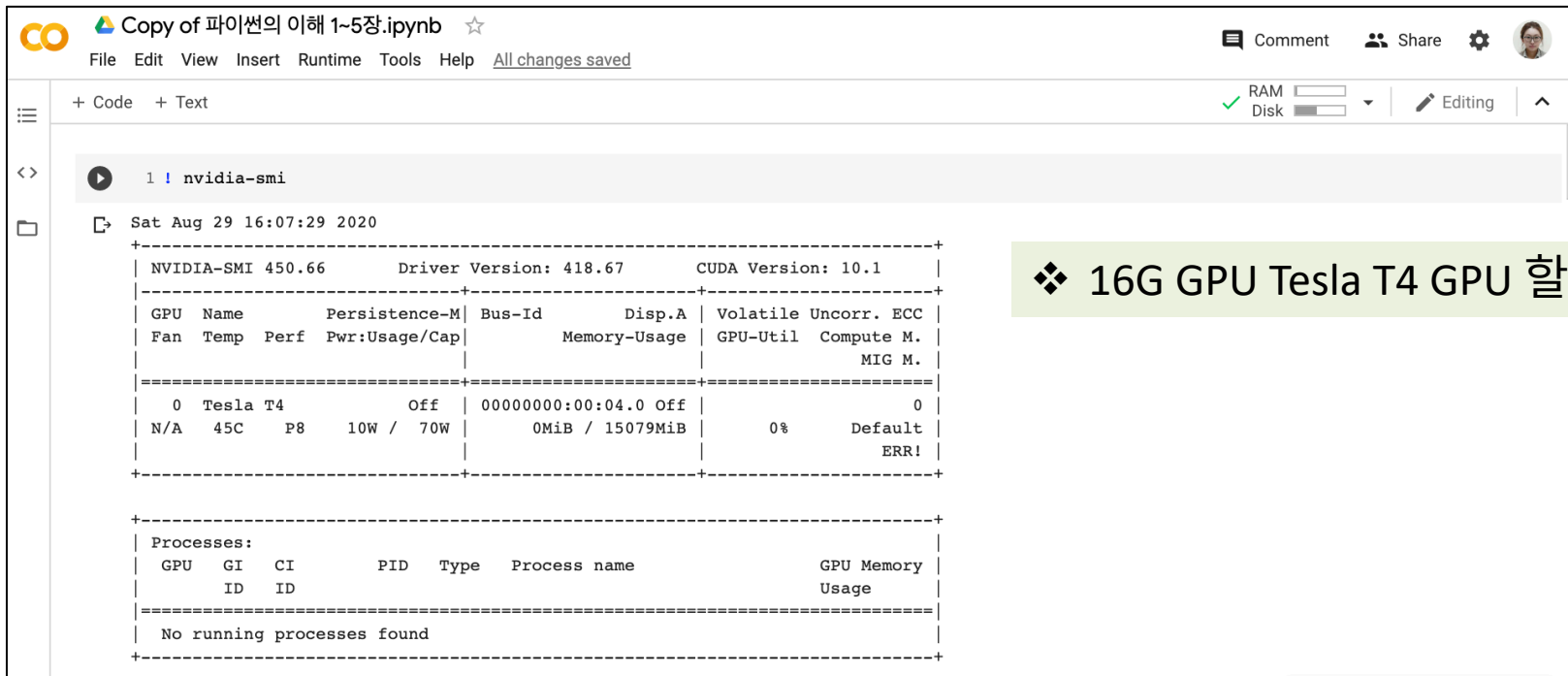
- Runtime → Change runtime type → Hardware accelerator → GPU → Save



# 코랩 사용법

## ■ 코랩 개발 환경 확인 방법

- 코랩 은 Cell 안에서 ! [리눅스 명령어] 를 통한 셀 실행을 지원함
- GPU 개발 환경 확인을 위한 nvidia-smi 명령어 활용



```
Copy of 파이썬의 이해 1~5장.ipynb ☆
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
1 ! nvidia-smi

Sat Aug 29 16:07:29 2020

+-----+
| NVIDIA-SMI 450.66                Driver Version: 418.67       CUDA Version: 10.1         |
+-----+-----+
| GPU   Name           Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf    Pwr:Usage/Cap|      Memory-Usage | GPU-Util  Compute M. |
|=====+=====+
| 0 Tesla T4             Off        | 00000000:00:04:0 Off |                    0 |
| N/A   45C    P8      10W /  70W |  0MiB / 15079MiB |      0%      Default |
+-----+-----+

+-----+
| Processes:                       |
| GPU   GI   CI        PID   Type   Process name                      GPU Memory |
| ID   ID                                   |             Usage                   |
+-----+-----+
| No running processes found      |
+-----+
```

❖ 16G GPU Tesla T4 GPU 할당됨

# 코랩 사용법

## ■ 코랩 개발 환경 확인 방법

- CPU 스펙 확인을 위한 `cpuinfo` 명령어 실행
- 메모리 사용/미사용 정보 확인을 위한 `free` 명령어 실행
- 서버 운영체제 확인법 실행

```
+ Code + Text
1 !cat /proc/cpuinfo

processor      : 0
vendor_id     : GenuineIntel
cpu family    : 6
model         : 79
model name    : Intel(R) Xeon(R) CPU @ 2.20GHz
```

❖ Xeon CPU 2개 할당됨

```
+ Code + Text
[3] 1 !free -h

             total        used        free      shared  buff/cache   available
Mem:          12G          535M          10G          972K          2.0G          11G
Swap:           0B           0B           0B
```

❖ 메모리 12G 할당됨

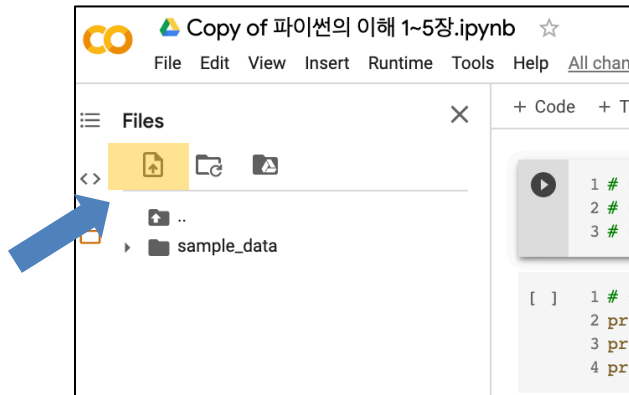
```
+ Code + Text
[1] 1 import platform
    2 platform.platform()

'Linux-4.19.112+-x86_64-with-Ubuntu-18.04-bionic'
```

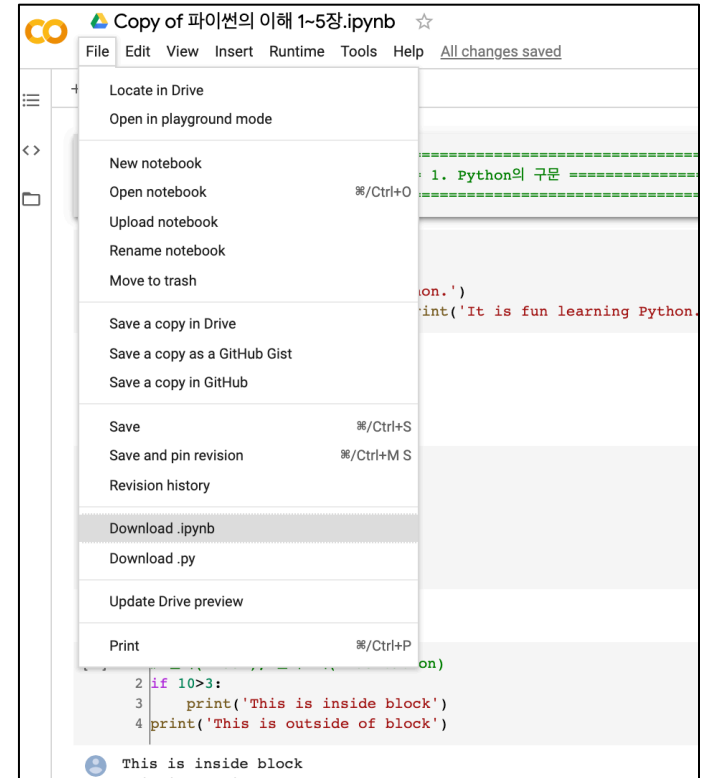
❖ 리눅스 18.04 운영체제

# 코랩 사용법

- 파일 업로드 하기
  - 업로드 버튼 클릭 ➔ 파일 선택



- 작성한 소스 파일 다운로드 하기
  - File ➔ Download .ipynb or .py



# 코랩 사용법: 마크다운 주석

## ■ [1단계] 헤더 (Header)

- 제목, 문단별 제목을 쓰고 싶을 때, 글의 구조(개요) 및 큰 틀을 잡을 때 사용한다.

```
# 제목 1단계
## 제목 2단계
### 제목 3단계
#### 제목 4단계
##### 제목 5단계
##### 제목 6단계
```

### 제목 1단계

### 제목 2단계

### 제목 3단계

### 제목 4단계

### 제목 5단계

### 제목 6단계

## ■ [2단계] 목록 (List)

- 요소를 나열 할 때

```
1. 첫번째
1. 두번째
1. 세번째

+ 순서없음
- 홍길동
* 중대장
+ 프로실망러
```

1. 첫번째

2. 두번째

3. 세번째

• 순서없음

◦ 홍길동

■ 중대장

■ 프로실망러

# 코랩 사용법: 마크다운 주석

## ■ [3단계] 수평선

- 내용을 명시적으로 구분하고 싶을 때

---

## ■ [4단계] 강조

- 문장 내 강조하고 싶은 단어를 눈에 띄게

\_\_볼드(진하게)\_\_  
\_이탤릭체(기울여서)\_  
~~취소선~~  
<u>밑줄</u>  
\_\_볼드로 진하게 만들다가\*이탤릭으로 기울이고\*다시 볼드로\_\_(중복 활용도 가능하다.)

**볼드(진하게)**

*이탤릭체(기울여서)*

취소선

밑줄

**볼드로 진하게 만들다가*이탤릭으로 기울이고*다시 볼드로(중복 활용도 가능하다.)**

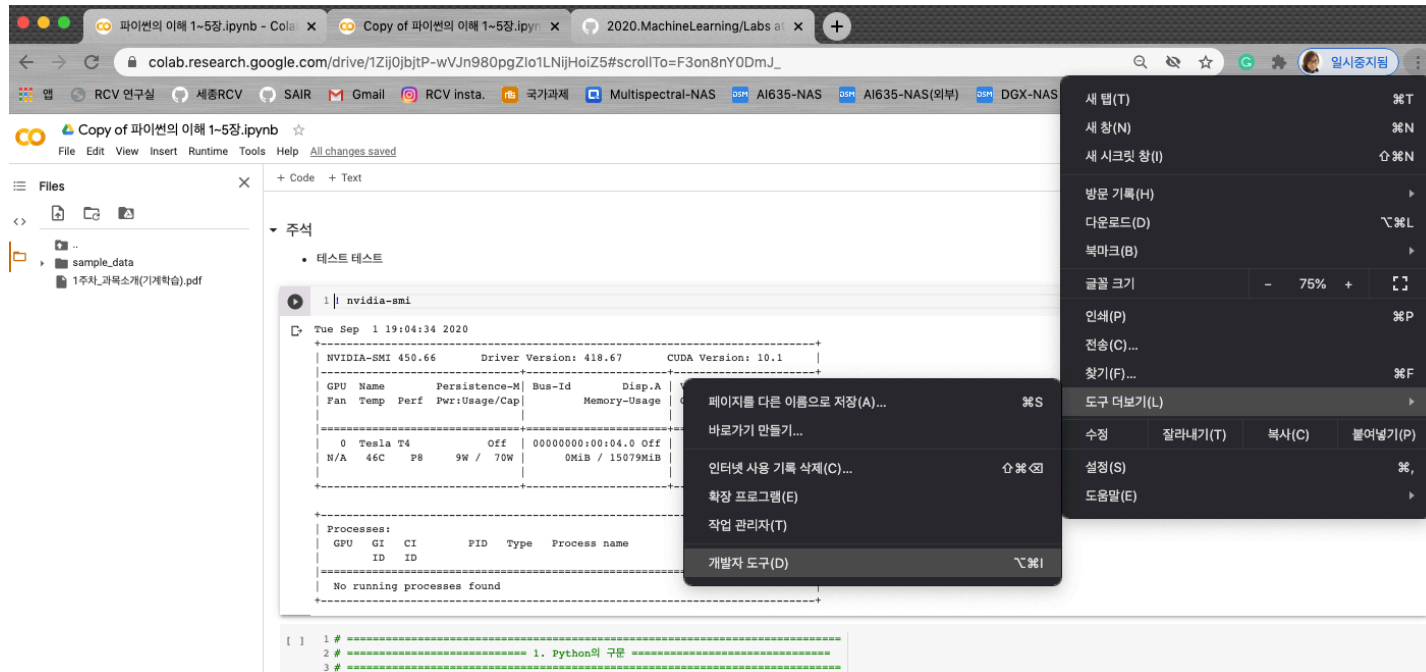
# 코랩 사용법

## ■ 코랩 세션의 단점

- 한 세션의 최대 유지 가능 시간은 **12시간**이지만 **90분** 이상 작업이 없는 경우 강제 세션 종료됨

## ■ 세션 유지 방법

- 브라우저의 개발자 모드(F12)의 console 탭을 통해서 실행시키면 된다.





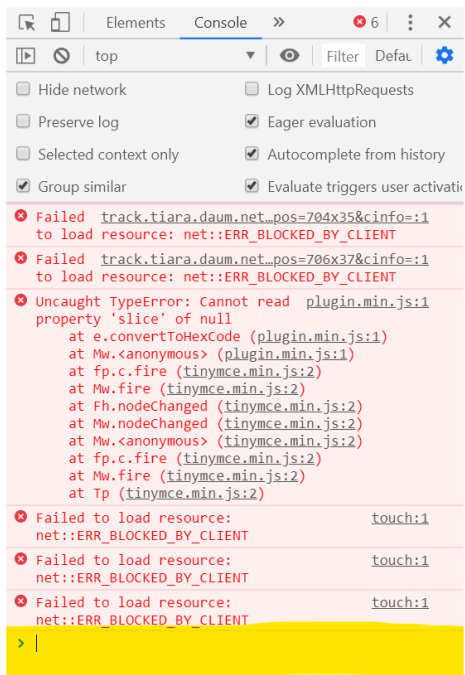
# 코랩 사용법

## ■ 코랩 세션의 단점

- 한 세션의 최대 유지 가능 시간은 12시간이지만 90분 이상 작업이 없는 경우 강제 세션 종료됨

## ■ 세션 유지 방법

- 브라우저의 개발자 모드(F12)의 console 탭을 통해서 실행시키면 된다.



# 코랩 사용법

- 구글 colab에서 90 time out 세션 유지 javascript 코드

```
function ClickConnect() {  
    var buttons = document.querySelectorAll("colab-dialog.yes-no-dialog paper-  
button#cancel");  
    buttons.forEach(function(btn) {  
        btn.click();  
    });  
    console.log("1분마다 자동 재연결");  
    document.querySelector("colab-toolbar-button#connect").click();  
}  
setInterval(ClickConnect,1000*60);
```

# 코랩 사용법

- 구글 colab에서 buffered data was truncated after reaching the output size limit 에러 방지를 위한 현재 출력 창 자동 지우기

```
function CleanCurrentOutput(){
    var btn = document.querySelector(".output-icon.clear_outputs_enabled.output-icon-selected[title$='현재 실행 중...'] iron-icon[command=clear-focused-or-selected-outputs]");
    if(btn) { console.log("30분마다 출력 지우기");
        btn.click();
    }
}
setInterval(CleanCurrentOutput,1000*60*30);
```