

03. 파이썬 환경 세팅과 프로그래밍 기초

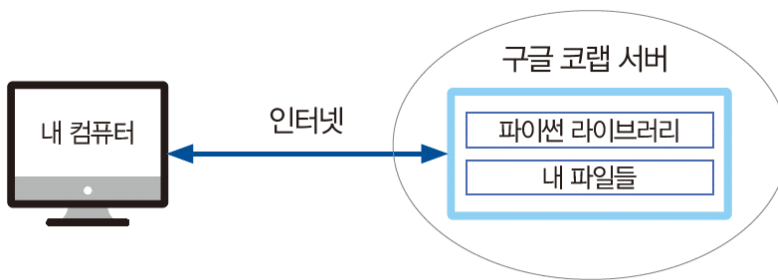
2.2.1 클라우드 방식과 스탠드얼론 방식

■ 클라우드 방식

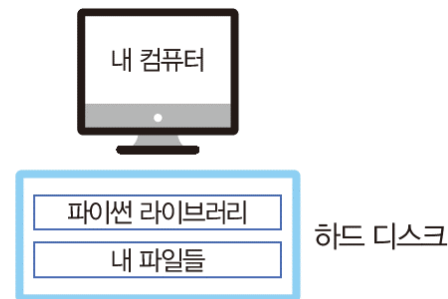
- 프로그램과 데이터가 서버에 저장되고 관리
 - 서버에 환경이 대부분 갖추어져 있어 로그인하면 바로 프로그래밍 가능
 - 인터넷 연결만 있으면 어느 곳에서나 개발 가능. 협업 가능
- 구글의 Colab, 아마존의 SageMaker, 마이크로소프트의 Azure
- 내 프로젝트에 최적의 환경을 갖추 수 없는 한계

■ 스탠드얼론 방식

- 자신에 최적의 환경 구축 가능. 프로그램과 데이터가 자신의 컴퓨터에 저장
- 소프트웨어를 설치하고 환경을 스스로 구축해야 함



(a) 클라우드 방식



(b) 스탠드얼론 방식

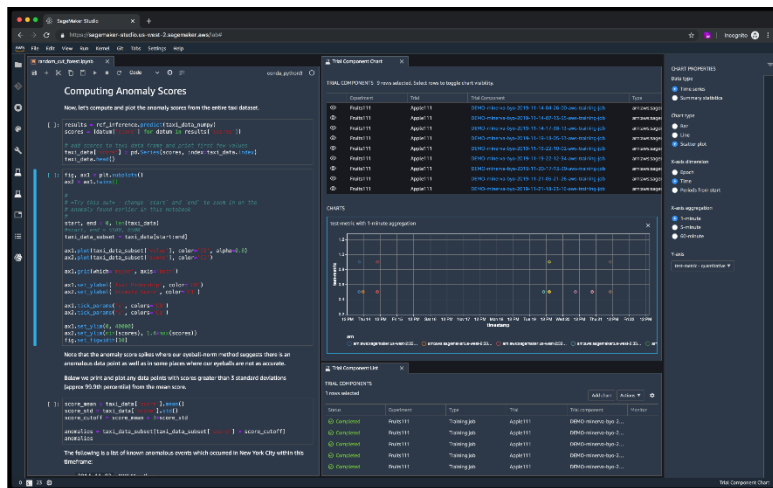
2.2.1 클라우드 방식과 스탠드얼론 방식

■ 아마존의 SageMaker란?

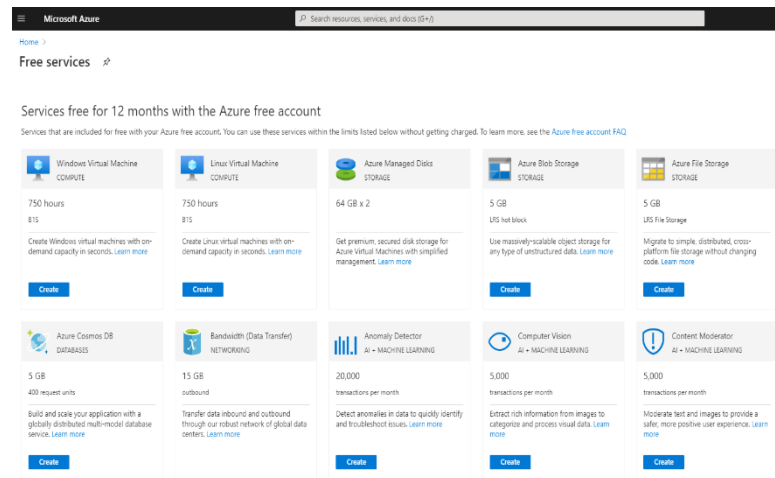
- 데이터 과학자 및 개발자가 모든 규모의 기계 학습 모델을 간편하게 빌드, 학습 및 배포할 수 있도록 하는 완전 관리형 서비스
- A/B 테스트 기능이 내장되어 있어 사용자가 최선의 결과를 내기 위해 모델을 테스트하고 여러 버전을 실험하는 데 용이

■ 마이크로소프트의 Azure란?

- 분석, 컴퓨팅, 데이터베이스, 모바일, 네트워킹, 저장소 및 웹이 통합된 클라우드 서비스
- Linux 컨테이너를 Docker 통합과 함께 실행하고, JavaScript, Python, .NET, PHP, Java 및 Node.js로 앱을 빌드하고, iOS, Android 및 Windows 장치용 백 엔드 구축 가능



SageMaker



Azure

2.2.2 파이썬 시작하기: 클라우드 방식의 colab(코랩)

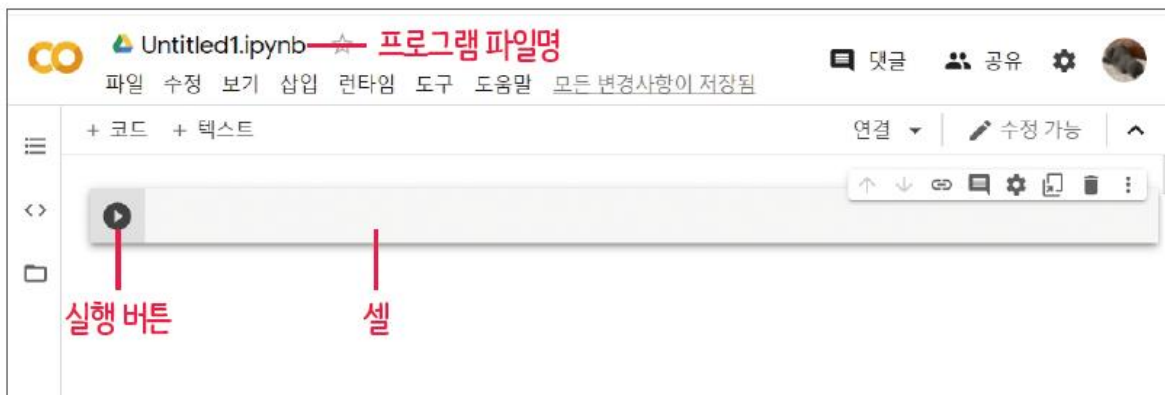
■ Colab의 특성

- 구글 클라우드에서 제공하는 Jupyter 노트북
- GPU를 무료로 사용 가능
- Github 친화적 환경
- 클라우드 기반 협업 용이
- 데이터 사이언스 라이브러리 활용 용이

■ Colab을 사용하는 절차

1. <http://colab.research.google.com>에 접속한다.
2. 구글 계정으로 로그인한다(구글 계정이 없다면 계정을 만든 다음에 로그인한다).
3. 파이썬 프로그래밍을 한다.
4. 프로그래밍을 마치면 프로그램 파일을 구글 드라이브에 저장한다.

2.2.2 파이썬 시작하기: 클라우드 방식의 colab



(a) 로그인한 화면



(b) 파이썬 코드를 입력하고 실행한 결과 화면

그림 2-2 코랩 화면

2.2.2 파이썬 시작하기: 클라우드 방식의 colab

■ 프로그래밍 장면

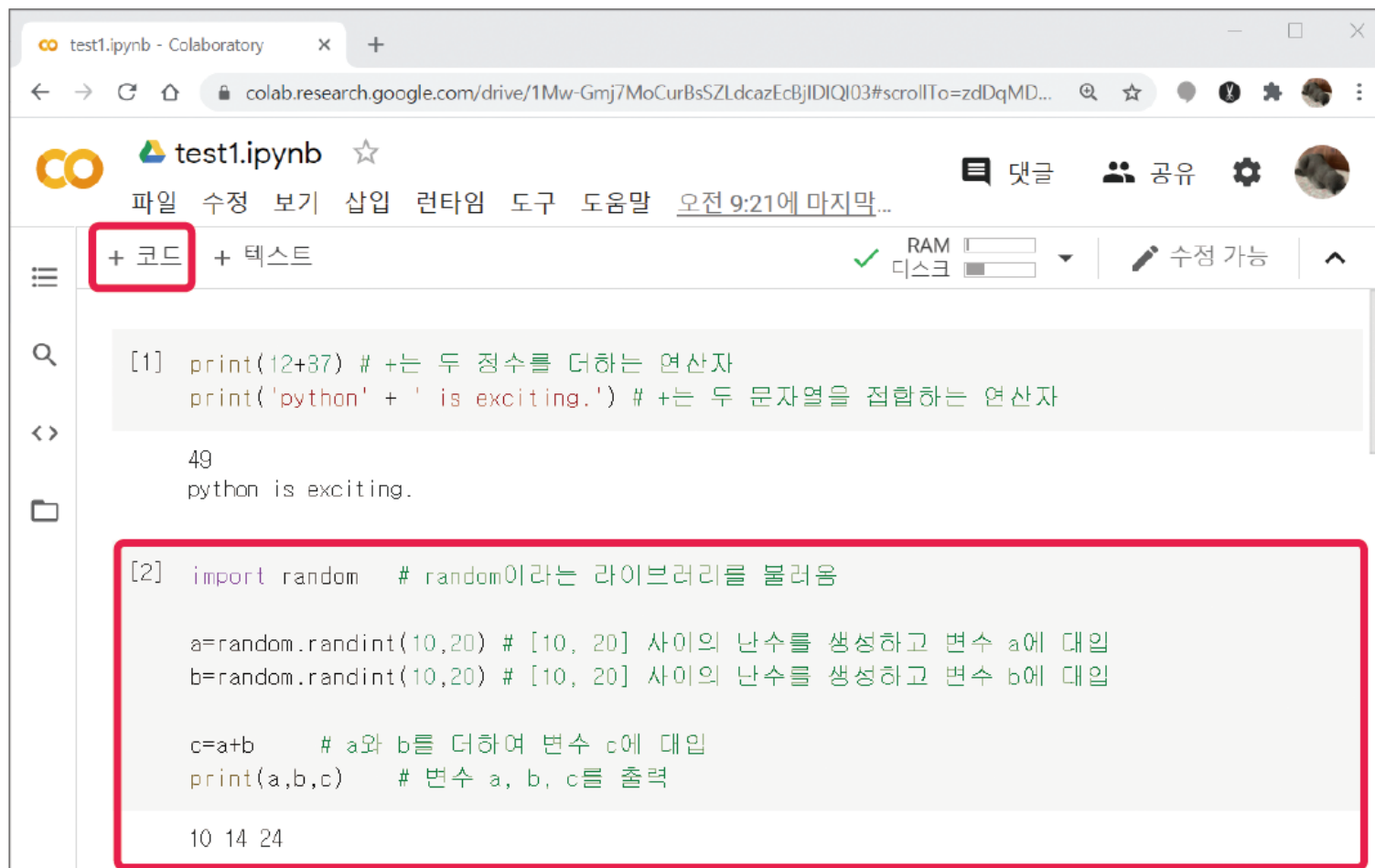


그림 2-3 코랩의 주피터 노트북에서 코드 추가

2.2.3 파이썬 시작하기: 스탠드얼론 방식

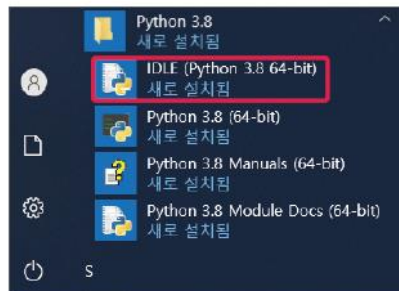
- 설치 장면(<http://www.python.org>에서 설치 파일 다운로드)



(a) 다운로드 화면



(b) 설치 시작 화면

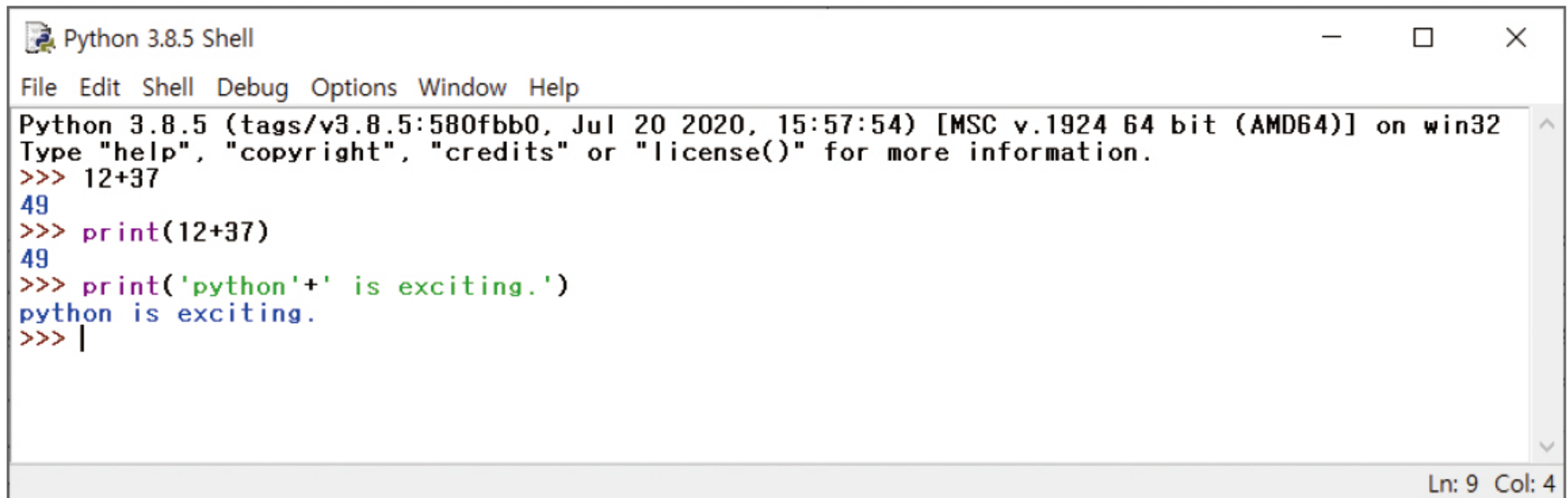


(c) 설치 확인

그림 2-4 파이썬 설치 과정

2.2.3 파이썬 시작하기: 스탠드얼론 방식

- 통합 개발 환경 IDLE로 시범 프로그래밍(셸 창에서 프로그래밍)



The screenshot shows a window titled "Python 3.8.5 Shell". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main text area displays the following content:

```
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:57:54) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 12+37
49
>>> print(12+37)
49
>>> print('python'+ ' is exciting.')
python is exciting.
>>> |
```

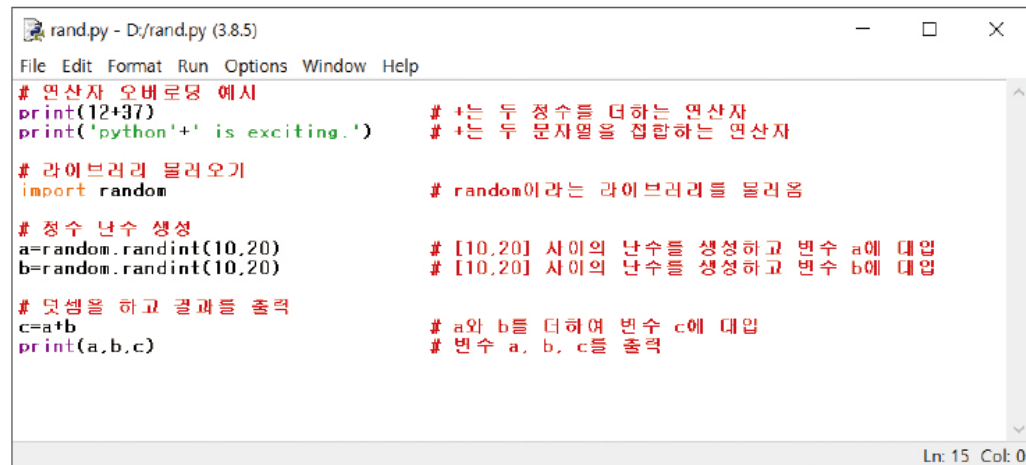
The status bar at the bottom right indicates "Ln: 9 Col: 4".

그림 2-5 파이썬에서 간단한 명령어 수행(셸 창에서 작업)

2.2.3 파이썬 시작하기: 스탠드얼론 방식

■ 스크립트 창에서 프로그래밍

- 셸 창에서 [File]-[New File] 메뉴로 스크립트 창을 열 수 있음
- 스크립트 창에서는 프로그램을 입력하고 여러 번 실행 가능



```

rand.py - D:/rand.py (3.8.5)
File Edit Format Run Options Window Help
# 연산자 오버로딩 예시
print(12+37) # +는 두 정수를 더하는 연산자
print('python'+ ' is exciting.') # +는 두 문자열을 접합하는 연산자

# 라이브러리 불러오기
import random # random이라는 라이브러리를 불러옴

# 정수 난수 생성
a=random.randint(10,20) # [10,20] 사이의 난수를 생성하고 변수 a에 대입
b=random.randint(10,20) # [10,20] 사이의 난수를 생성하고 변수 b에 대입

# 덧셈을 하고 결과를 출력
c=a+b # a와 b를 더하여 변수 c에 대입
print(a,b,c) # 변수 a, b, c를 출력

Ln: 15 Col: 0

```

(a) 스크립트 창에서 [프로그램 2-1]을 입력하고 실행



```

Python 3.8.5 Shell
File Edit Shell Debug Options Window Help
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:57:54) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 12+37
49
>>> print(12+37)
49
>>> print('python'+ ' is exciting.')
python is exciting.
>>>
===== RESTART: D:/rand.py =====
49
python is exciting.
12 17 29
>>>

Ln: 14 Col: 4

```

(b) 셸 창에서 프로그램 실행 결과를 확인

그림 2-6 스크립트 창에서 간단하게 작성한 프로그램

2.3 영리한 프로그래밍 환경: 아나콘다

■ 라이브러리 충돌 가능성

- 오픈 소스 특성으로 인해 라이브러리 충돌 가능성 발생(함수 API 변경 또는 환경 변수 값 충돌 등)
- 가상 환경을 사용하여 해결

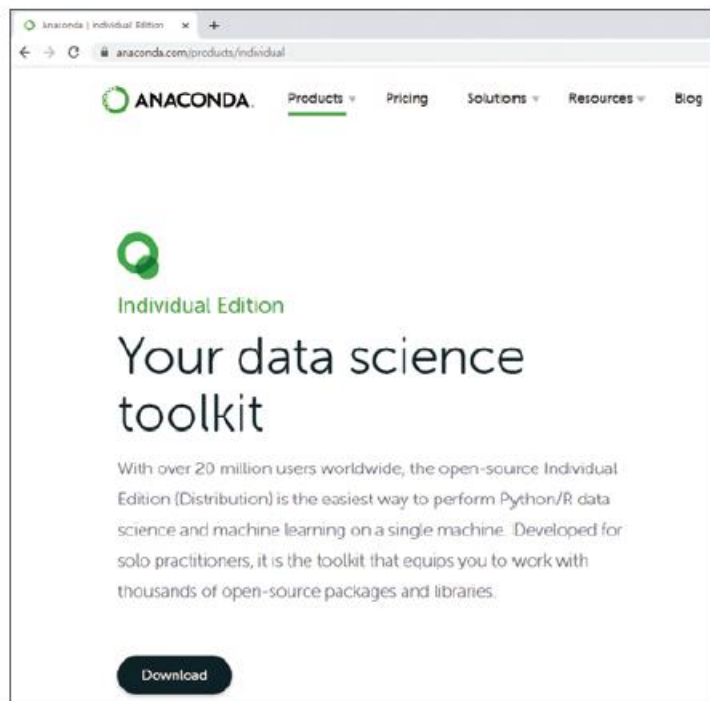
■ 가상 환경

- 프로젝트 별로 별도의 가상 환경을 만들어 일관된 환경 유지
- 아나콘다는 가상 환경을 지원

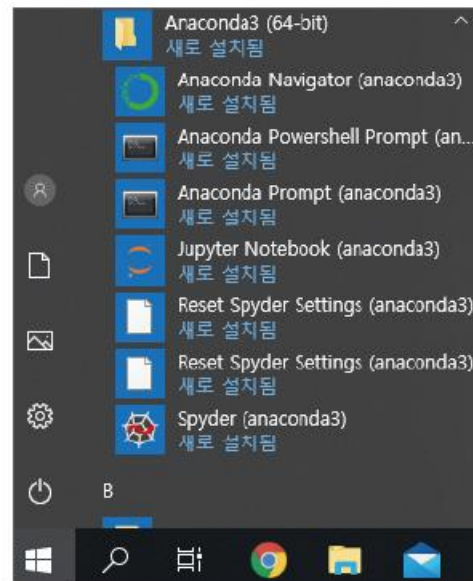
2.3.1 아나콘다 프로그래밍 환경

■ 네 단계의 설치

1. 아나콘다를 설치한다. <http://anaconda.com>



(a) 아나콘다 다운로드 화면



(b) 윈도우 시작 버튼으로 설치 확인

그림 2-7 아나콘다 설치

2.3.1 아나콘다 프로그래밍 환경

■ 네 단계의 설치

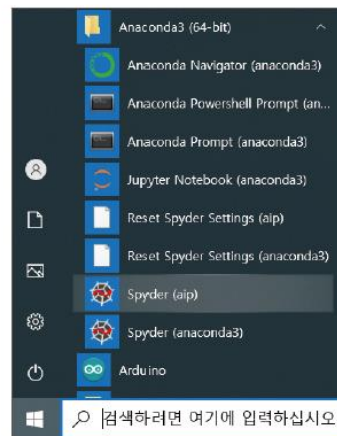
2. 가상 환경을 생성한다(아래 ① 번 명령어).

[aip라는 새로운 가상 환경을 만들고 스파이더와 텐서플로를 설치하는 과정]

(base) C:/> conda create -n aip python=3.7	① aip 가상 환경 생성
(base) C:/> conda activate aip	② aip 가상 환경으로 이동
(aip) C:/> conda install spyder	③ aip 가상 환경에 스파이더 설치
(aip) C:/> conda install tensorflow	④ aip 가상 환경에 텐서플로 설치



(a) 아나콘다의 프롬프트 창



(b) 윈도우의 시작 화면

그림 2-8 아나콘다 프롬프트 화면에서 가상 환경 만들기

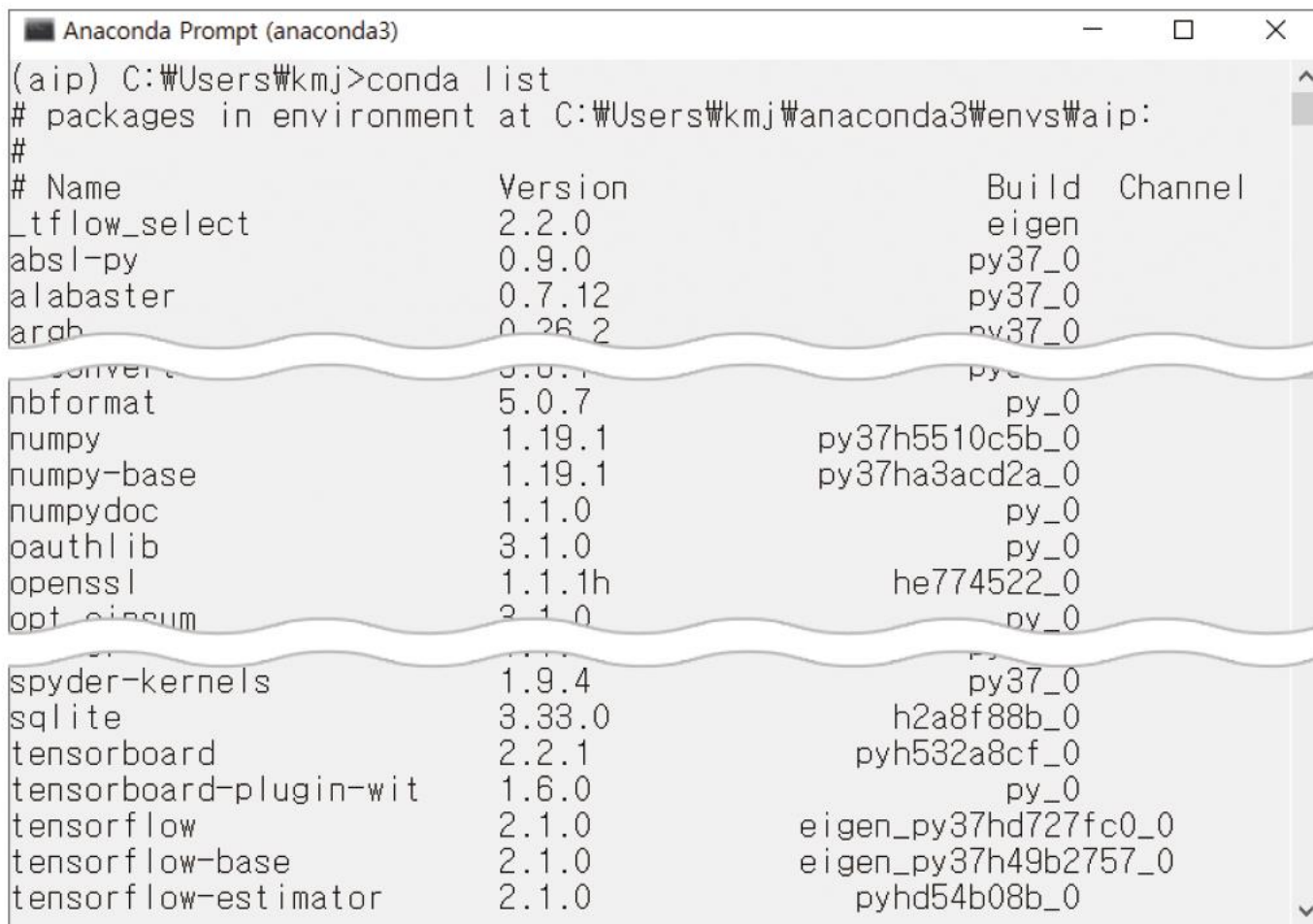
2.3.1 아나콘다 프로그래밍 환경

■ 네 단계의 설치

3. 가상 환경에 Spyder와 텐서플로 모듈을 설치한다(2 3 4 번 명령어).
4. 스파이더에서 작업 디렉토리를 설정한다.
 - 바탕 화면에 aipSources 폴더 만들기
 - 스파이더에서 [Tools]-[Preferences]-[Current working directory]-[Startup]-[The following directory]를 선택하고 aipSources를 브라우징하여 설정

2.3.1 프로그래밍 환경 확인

- > conda list 명령어로 가상 환경에 설치된 모듈 목록 확인



```

Anaconda Prompt (anaconda3)
(aip) C:\Users\Wkmj>conda list
# packages in environment at C:\Users\Wkmj\anaconda3\envs\aip:
#
# Name                                Version                                Build      Channel
tensorflow-select                    2.2.0                                eigen
absl-py                             0.9.0                                py37_0
alabaster                            0.7.12                               py37_0
argh                                 0.26.2                               py37_0
convert                              0.0.1                                py_0
nbformat                             5.0.7                                py_0
numpy                                1.19.1                               py37h5510c5b_0
numpy-base                          1.19.1                               py37ha3acd2a_0
numpydoc                             1.1.0                                py_0
oauthlib                             3.1.0                                py_0
openssl                              1.1.1h                               he774522_0
opt_einsum                           2.1.0                                py_0
spyder-kernels                       1.9.4                                py37_0
sqlite                                3.33.0                               h2a8f88b_0
tensorboard                          2.2.1                                pyh532a8cf_0
tensorboard-plugin-wit               1.6.0                                py_0
tensorflow                           2.1.0                                eigen_py37hd727fc0_0
tensorflow-base                      2.1.0                                eigen_py37h49b2757_0
tensorflow-estimator                 2.1.0                                pyhd54b08b_0

```

그림 2-9 conda list 명령어로 모듈이 제대로 설치되었는지 확인

2.3.1 프로그래밍 환경 확인

■ 아나콘다 내비게이터 화면

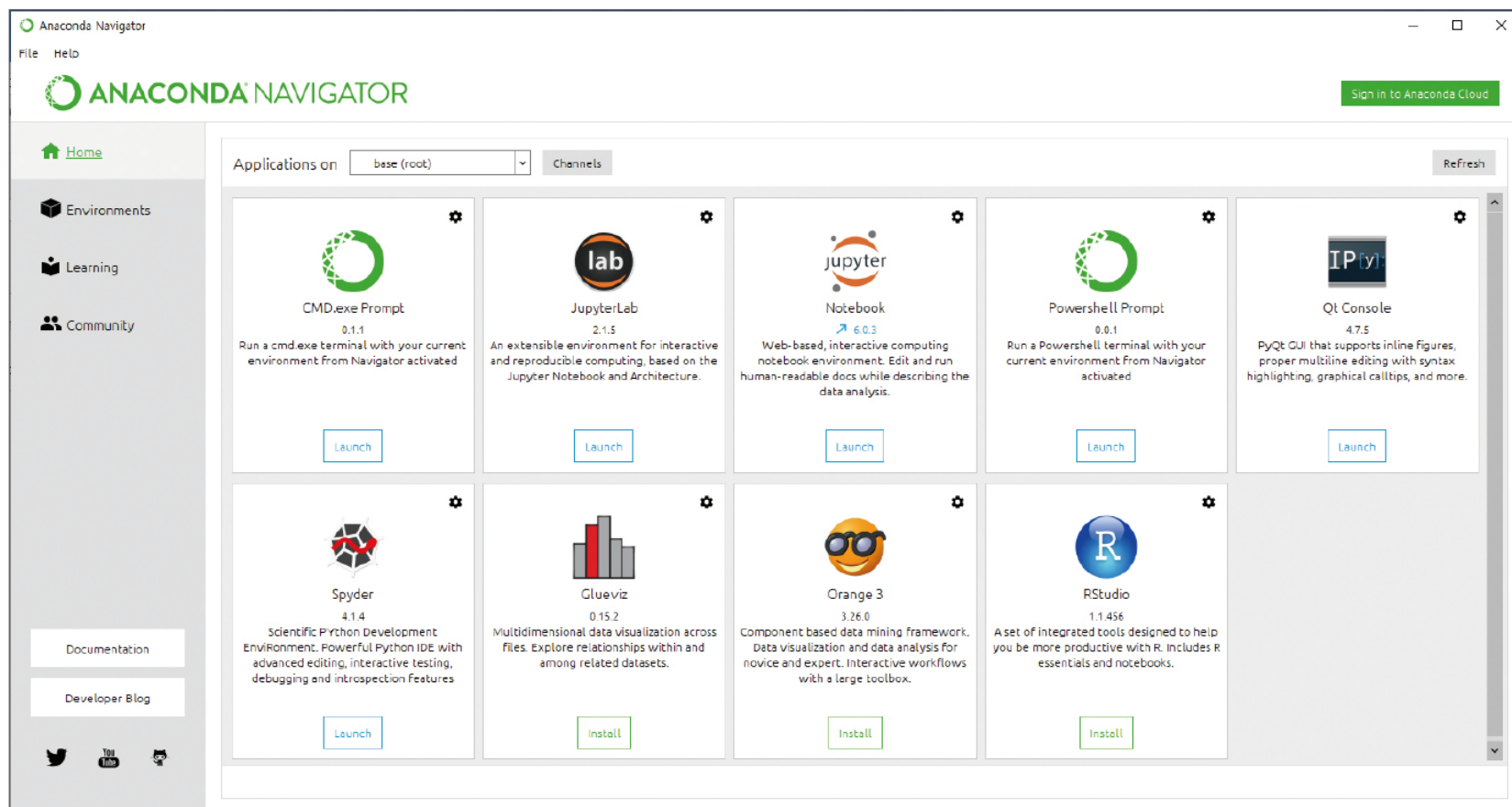
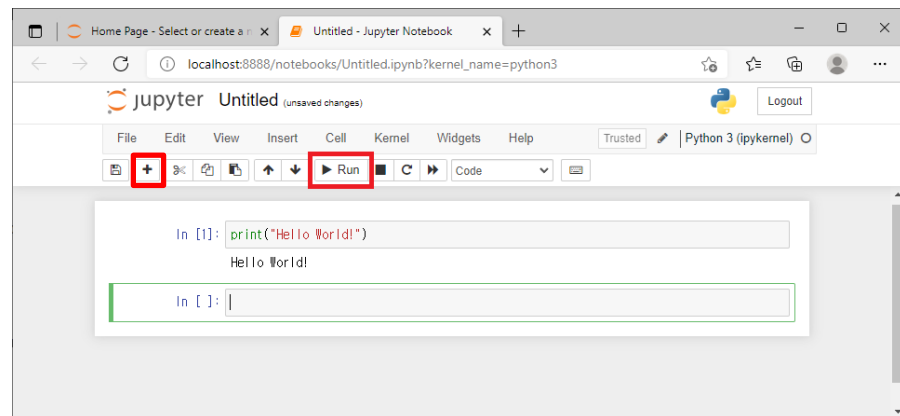
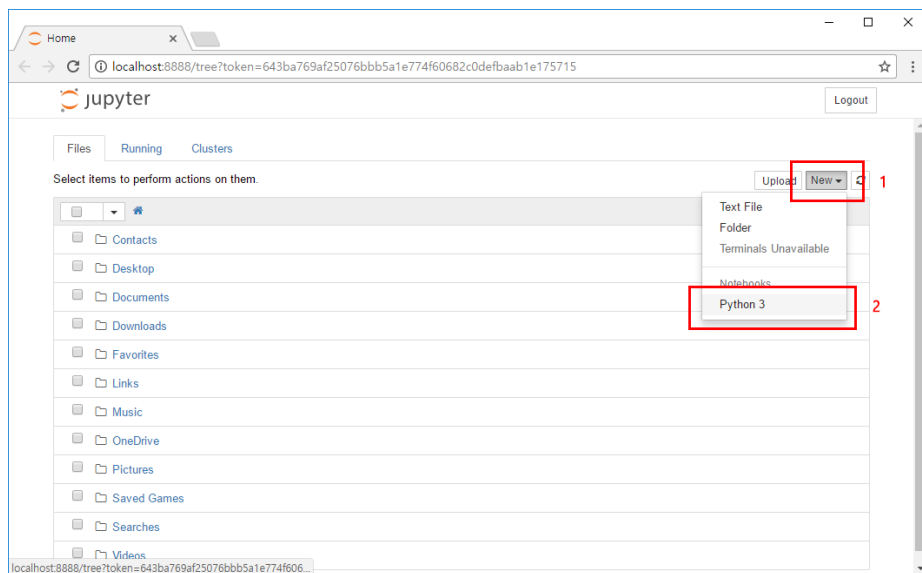


그림 2-10 아나콘다 내비게이터

2.3.2 주피터 노트북 사용하기

- 시작 > Anaconda3 (64-bit) > Jupyter Notebook 클릭
- 오른쪽 New 버튼을 클릭한 뒤 Python 3 클릭
- 스파이더 화면



- * Ctrl+Enter : 해당 Cell만 실행됨
- * Shift+Enter (= Run 버튼) : 해당 Cell 실행 후 다음 Cell로 넘어감
- * Ctrl+Enter : 해당 Cell이 실행된 후 아래에 새로운 Cell이 생성됨

2.3.3 클라우드 방식과 스탠드얼론 방식의 선택 기준

■ 간편한 실습에서는 colab

- 특히 단기 프로그래밍 과정에서는 설치 과정에 대한 시간 절약

■ 본격적인 개발에서는 스탠드얼론 방식

- 특히 인공지능 공학자로 성장하려 하는 경우에는 스스로 최적 환경을 구축할 수 있는 스탠드얼론 방식이 적절함

2.3.4 colab 기본 사용법 시연

■ Colab과 github의 사용 시연

- Colab에서 작성한 코드를 손쉽게 github에 업로드 하고 공유 및 수정할 수 있음

The screenshot shows the Google Colab interface with a notebook named 'colab_tutorial.ipynb'. The code cell contains the following Python code:

```
1 3+5
9

[] 1 import tensorflow as tf

/usr/local/lib/python3.7/dist-packages/requests/__init__.py:91: RequestsDependencyWarning: urllib3 (1.25.12)
RequestsDependencyWarning)

[] 1 from google.colab import drive
2 drive.mount('/content/drive')

Mounted at /content/drive

[] 1 import keras

1 import boto3
```

The output shows a `ModuleNotFoundError` for `boto3`. The error message states: `ModuleNotFoundError: No module named 'boto3'`. A note below the error suggests installing dependencies manually using `!pip` or `!apt`.

The screenshot shows the GitHub repository page for 'c-karl / DA_DS_Book001'. The repository is public and has 1 contributor. The file '10.1.2.탐색적 데이터 분석(EDA).ipynb' is selected. The code cell contains the following Python code:

```
In [1]: # 필요한 패키지 설치
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
sns.set(color_codes=True)
%matplotlib inline

In [2]: # 데이터 불러오기
# https://www.kaggle.com/datasets/jassemostipak/hotel-booking-demand
df = pd.read_csv("datasets/hotel_bookings.csv")

# 데이터 샘플 확인
df.head()
```

The output shows the first few rows of the `df` DataFrame:

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_nights
0	Resort Hotel	0	342	2015	July	27	1	0
1	Resort Hotel	0	737	2015	July	27	1	0
2	Resort Hotel	0	7	2015	July	27	1	0
3	Resort	0	13	2015	July	27	1	0

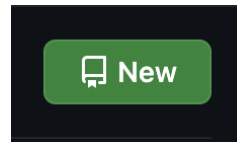
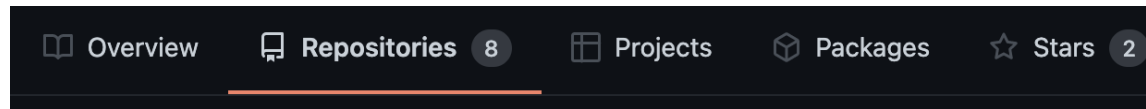
2.3.4 colab 기본 사용법 실습

- 새로운 Colab 노트에서 kaggle의 데이터셋을 업로드하여 확인한다.
- 데이터셋의 각 통계치를 확인한다.
- 해당 노트를 Github에 사본으로 저장한다.
- Github에 접속하여 저장된 노트를 확인 및 Colab 링크로 접속해본다.

2.3.5 colab과 github 연동 실습

■ 연동하고 싶은 Github Repository 생성




- 본인 Github에서 "Repositories" 탭 클릭
- 오른쪽 상단 "New" 초록색 버튼 클릭



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner ^{*} Repository name ^{*}

  Test_Repo 

Great repository names are short and memorable. Need inspiration? How about [fantastic-system](#)?

Description (optional)

☒ **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐ **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☒ **Add a README file**
This is where you can write a long description for your project. [Learn more.](#)

☐ **Add .gitignore**
Choose which files not to track from a list of templates. [Learn more.](#)

☐ **Choose a license**
A license tells others what they can and can't do with your code. [Learn more.](#)

This will set `main` as the default branch. Change the default name in your [settings](#).

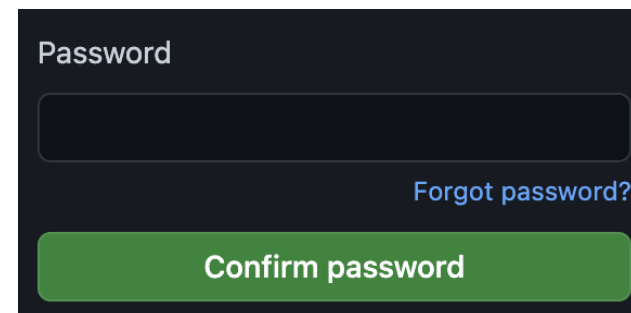
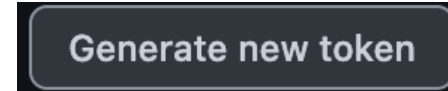
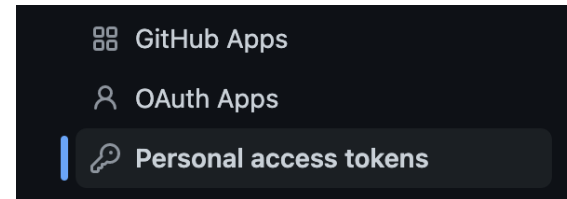
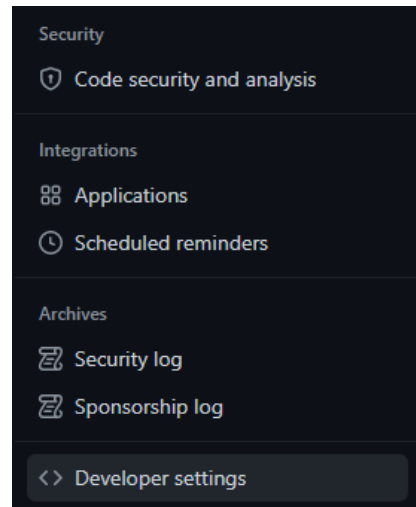
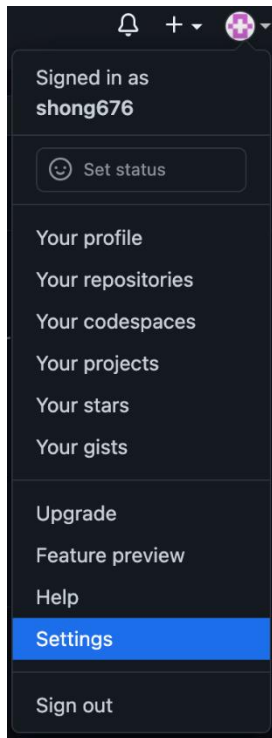
[Create repository](#)

- **Repository name:** 생성하고싶은 폴더명(혹은 프로젝트명)
- **Description:** 해당 Repository 메인페이지 설명글
- **Public:** 해당 Repository를 공개적으로 설정
- **Private:** 해당 Repository를 비공개적으로 설정(개인적/팀 프로젝트일 경우)
- **Add a README file:** Repository 에 대한 설명글 파일 생성
- **Add a .gitignore:** 필요하지 않은 파일을 업로드에서 제외

2.3.5 colab과 github 연동 실습

■ Repository 접근을 위한 Access Token 생성

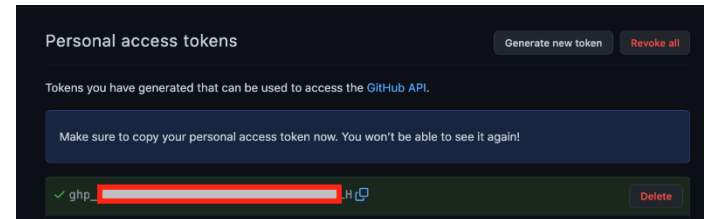
- 오른쪽 상단의 Setting → 왼쪽 메뉴들 중 가장 아래에 있는 "Developer settings" 클릭
- 왼쪽 메뉴 하단 "Developer settings" → "Personal access tokens" 클릭
- 오른쪽 상단 "Generate new token" → 비밀번호 확인



2.3.5 colab과 github 연동 실습

Repository 접근을 위한 Access Token 생성

- 하단 그림과 같이 설정하여 토큰 생성
- 토큰은 다시 볼 수 없으니 꼭 복사
- 노출되지 않도록 주의



Note

메모해 둘 토큰 이름

What's this token for?

Expiration *

30 days The token will expire on Tue, Apr 19 2022

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes.](#)

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input checked="" type="checkbox"/> repo:status	Access commit status
<input checked="" type="checkbox"/> repo_deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input checked="" type="checkbox"/> repo:invite	Access repository invitations
<input checked="" type="checkbox"/> security_events	Read and write security events
<input type="checkbox"/> workflow	Update GitHub Action workflows
<input type="checkbox"/> write:packages	Upload packages to GitHub Package Registry
<input type="checkbox"/> read:packages	Download packages from GitHub Package Registry
<input type="checkbox"/> delete:packages	Delete packages from GitHub Package Registry
<input type="checkbox"/> admin:org	Full control of orgs and teams, read and write org projects
<input type="checkbox"/> write:org	Read and write org and team membership, read and write org projects
<input checked="" type="checkbox"/> read:org	Read org and team membership, read org projects
<input type="checkbox"/> admin:public_key	Full control of user public keys
<input type="checkbox"/> write:public_key	Write user public keys
<input type="checkbox"/> read:public_key	Read user public keys

<input type="checkbox"/> admin:repo_hook	Full control of repository hooks
<input type="checkbox"/> write:repo_hook	Write repository hooks
<input type="checkbox"/> read:repo_hook	Read repository hooks
<input type="checkbox"/> admin:org_hook	Full control of organization hooks
<input checked="" type="checkbox"/> gist	Create gists
<input type="checkbox"/> notifications	Access notifications
<input type="checkbox"/> user	Update ALL user data
<input type="checkbox"/> read:user	Read ALL user profile data
<input type="checkbox"/> user:email	Access user email addresses (read-only)
<input type="checkbox"/> user:follow	Follow and unfollow users
<input type="checkbox"/> delete_repo	Delete repositories
<input type="checkbox"/> write:discussion	Read and write team discussions
<input type="checkbox"/> read:discussion	Read team discussions
<input type="checkbox"/> admin:enterprise	Full control of enterprises
<input type="checkbox"/> manage_runners:enterprise	Manage enterprise runners and runner-groups
<input type="checkbox"/> manage_billing:enterprise	Read and write enterprise billing data
<input type="checkbox"/> read:enterprise	Read enterprise profile data
<input type="checkbox"/> admin:pgp_key	Full control of public user GPG keys (Developer Preview)
<input type="checkbox"/> write:pgp_key	Write public user GPG keys
<input type="checkbox"/> read:pgp_key	Read public user GPG keys

Generate token Cancel

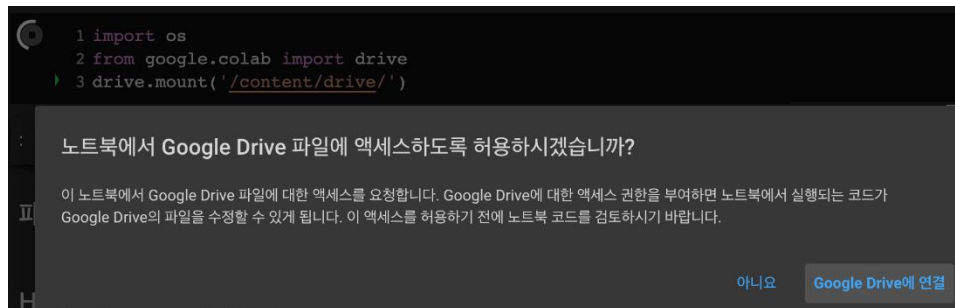
2.3.5 colab과 github 연동 실습

■ 코랩의 드라이브 마운트

- 커밋할 폴더 내 코랩 소스파일에서 아래의 코드를 작성하고 실행

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

- 팝업창에서 Google Drive 연결 버튼 클릭



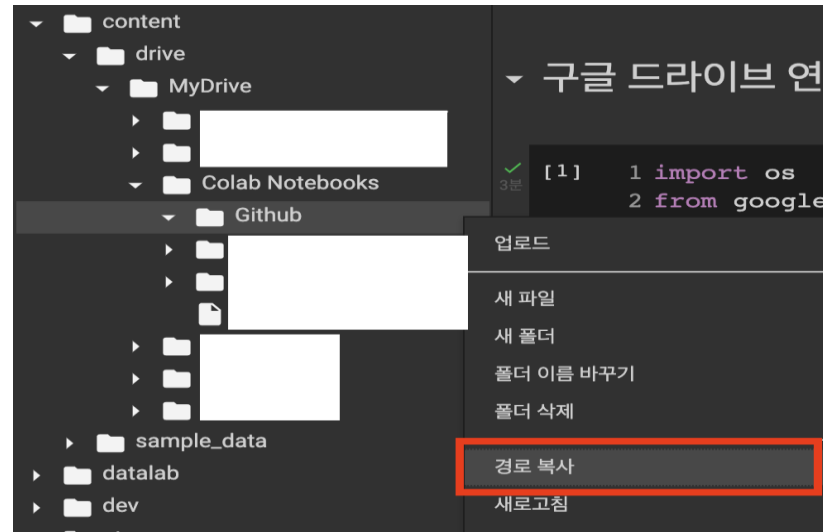
- Github 커밋용 폴더 생성

A screenshot of a "새 폴더" (New Folder) dialog box. It features a text input field containing the word "Github". Below the input field, there are two buttons: "취소" (Cancel) and "만들기" (Create).

2.3.5 colab과 github 연동 실습

■ 디렉토리 변경

- 앞서 생성한 폴더의 경로 복사



- Cd 명령어로 디렉토리 변경 실행

```
[6] 1 cd /content/drive/MyDrive/Colab Notebooks/Github
/content/drive/MyDrive/Colab Notebooks/Github
```


2.3.5 colab과 github 연동 실습

■ Git 클론

- 코랩과 연동할 리포지토리 정보 가져오기

```
!git clone https://[Github 아이디]:[access token]@github.com/[Github 아이디]/[리포지토리 이름].git
```

```
1 !git clone https://park-gb:[redacted]@github.com/park-gb/financial-news-sentiment-classifier.git
```

```
Cloning into 'financial-news-sentiment-classifier'...
remote: Enumerating objects: 23, done.
remote: Counting objects: 100% (23/23), done.
remote: Compressing objects: 100% (17/17), done.
remote: Total 23 (delta 3), reused 20 (delta 3), pack-reused 0
Unpacking objects: 100% (23/23), done.
```

- 앞서 생성한 폴더로 이동해 리포지토리가 정상적으로 clone 되었는지 확인

My Drive > Commit_Test_Folder ▾

Name ↑



Test_Repo

2.3.5 colab과 github 연동 실습

■ Git 접근권한 부여

- 생성한 폴더로 디렉토리 변경

```
cd 리포지토리 이름
```

```
1 cd financial-news-sentiment-classifier/
```

```
↳ /content/drive/MyDrive/Colab Notebooks/Github/financial-news-sentiment-c
```

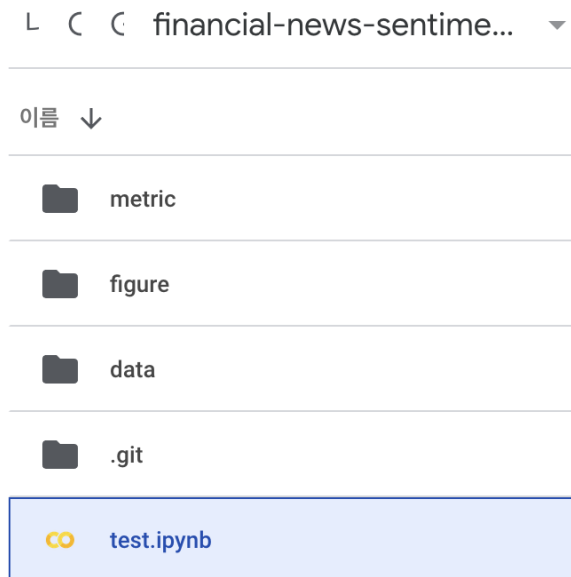
- 아래 코드에서 Github 이메일 주소와 아이디를 각각 입력하고 소스파일에서 실행

```
!git config --global user.email 'Github 이메일'  
!git config --global user.name 'Github 아이디'
```

2.3.5 colab과 github 연동 실습

■ Commit 테스트

- 구글 드라이브에 임의의 파일 생성



- 아래 코드로 Git에 파일 올리기
`!git add (생성한 파일명).ipynb`

- 아래 코드로 커밋 및 푸시 실행

```
!git commit -m '원하는 아무 메시지'
!git push
```

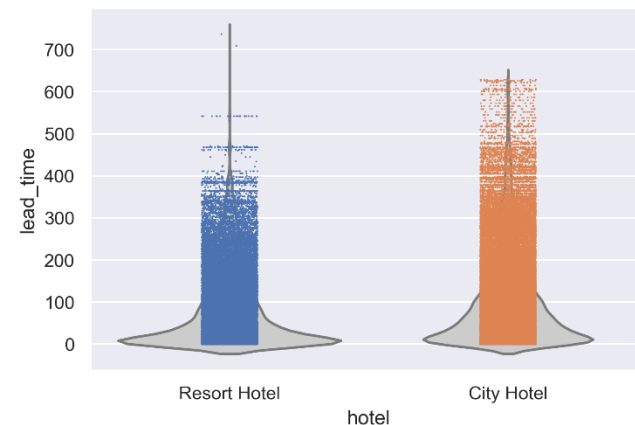
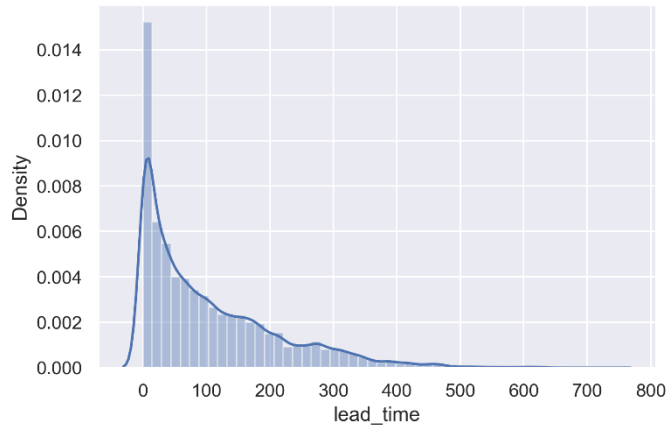
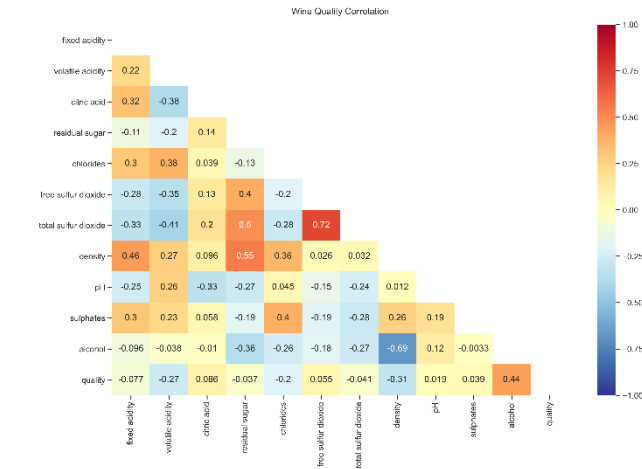
data	initial commit	4 days ago
figure	initial commit	4 days ago
metric	initial commit	4 days ago
.DS_Store	initial commit	4 days ago
LICENSE	Initial commit	4 days ago
README.md	Update README	4 days ago
Sentiment_Classifier_Fi_Ne...	Change a source file	4 days ago
test.ipynb	Connect Google Colab and Drive	3 minutes ago

2.4 탐색적 데이터 분석과 상관분석

■ ML, AI 모델을 만들기에 앞서 데이터를 확인하는 방법

```
[ ] 1 # 각 컬럼의 통계치 확인
    2 df.describe()
```

	is_canceled	lead_time	arrival_date_year	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_nights
count	119390.000000	119390.000000	119390.000000	119390.000000	119390.000000	119390.000000
mean	0.370416	104.011416	2016.156554	27.165173	15.798241	0.927599
std	0.482918	106.863097	0.707476	13.605138	8.780829	0.998613
min	0.000000	0.000000	2015.000000	1.000000	1.000000	0.000000
25%	0.000000	18.000000	2016.000000	16.000000	8.000000	0.000000
50%	0.000000	69.000000	2016.000000	28.000000	16.000000	1.000000
75%	1.000000	160.000000	2017.000000	38.000000	23.000000	2.000000
max	1.000000	737.000000	2017.000000	53.000000	31.000000	19.000000



2.4.1 탐색적 데이터 분석과 상관분석

- 탐색적 데이터 분석
- 시연, 실습
- <https://bit.ly/3xCHGn5>

```
In [1]: # 필요한 패키지 설치

import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
sns.set(color_codes=True)
%matplotlib inline

In [2]: # 데이터 불러오기
# https://www.kaggle.com/jessamostipak/hotel-booking-demand
df = pd.read_csv("datasets/hotel_bookings.csv")

# 데이터 샘플 확인
df.head()
```

```
Out[2]:
```

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_date_week_number	arrival_date_day_of_month	stays_in_weekend_nights	stays_in_week_nights	adults
0	Resort Hotel	0	342	2015	July	27	1	0	0	2
1	Resort Hotel	0	737	2015	July	27	1	0	0	2
2	Resort Hotel	0	7	2015	July	27	1	0	1	1
3	Resort Hotel	0	13	2015	July	27	1	0	1	1
4	Resort Hotel	0	14	2015	July	27	1	0	2	2

5 rows × 11 columns

```
In [6]: df2 = df[['hotel', 'lead_time']]
df2.head()
```

```
Out[6]:
```

	hotel	lead_time
0	Resort Hotel	342
1	Resort Hotel	737
2	Resort Hotel	7
3	Resort Hotel	13
4	Resort Hotel	14

왜도

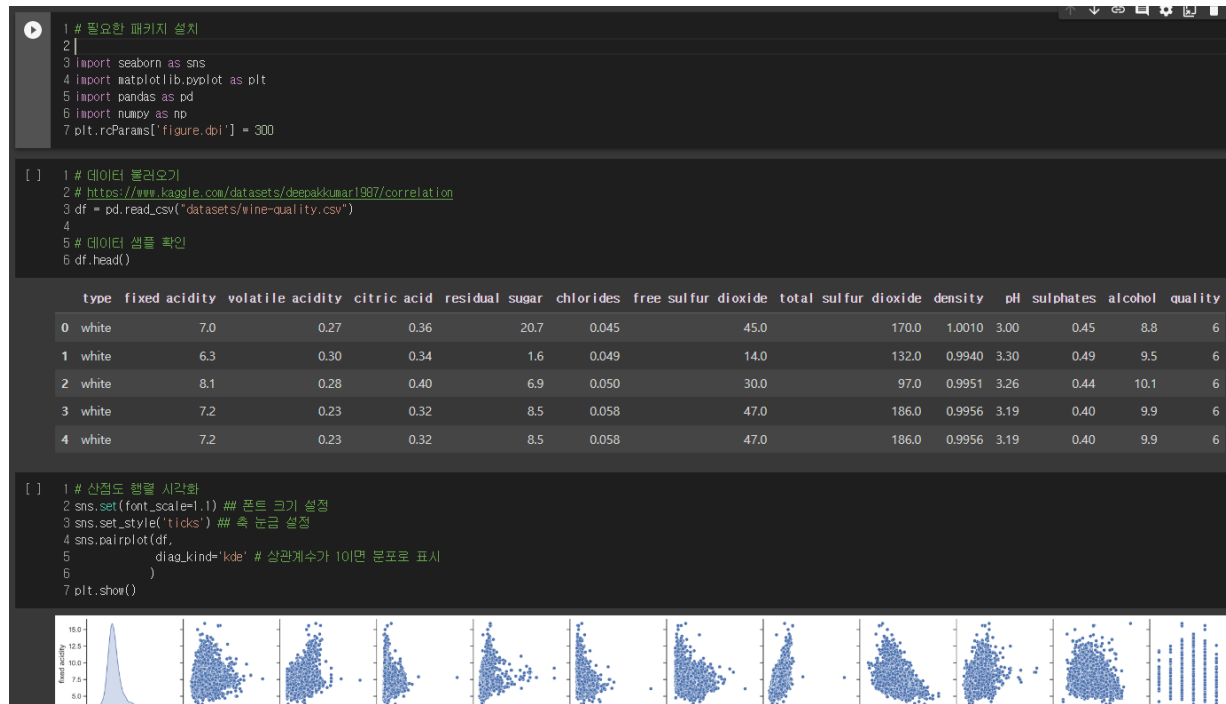
$$\frac{3(\text{평균값} - \text{중앙값})}{\text{표본의 표준편차}} \text{ or } \frac{3(\text{평균값} - \text{최빈값})}{\text{표본의 표준편차}}$$

첨도

$$\frac{\sum_{i=1}^n (\text{관측치} - \text{표본평균})^4}{\frac{\text{전체 관측치 수}}{(\text{표본분산})^2}} - 3 \text{ (경우에 따라)}$$

2.4.2 탐색적 데이터 분석과 상관분석

- 공분산과 상관성 분석
- 시연, 실습
- <https://bit.ly/3DM6gG6>



2.4.2 탐색적 데이터 분석과 상관분석

■ 공분산

$$X = \begin{bmatrix} | & | & & | \\ X_1 & X_2 & \dots & X_n \\ | & | & & | \end{bmatrix} \quad X^T X = \begin{bmatrix} \text{---} X_1 \text{---} \\ \text{---} X_2 \text{---} \\ \dots \\ \text{---} X_n \text{---} \end{bmatrix} \begin{bmatrix} | & | & & | \\ X_1 & X_2 & \dots & X_n \\ | & | & & | \end{bmatrix}$$

$$= \begin{bmatrix} \text{dot}(X_1, X_1) & \text{dot}(X_1, X_2) & \dots & \text{dot}(X_1, X_n) \\ \text{dot}(X_2, X_1) & \text{dot}(X_2, X_2) & \dots & \text{dot}(X_2, X_n) \\ \dots & \dots & \ddots & \dots \\ \text{dot}(X_n, X_1) & \text{dot}(X_n, X_2) & \dots & \text{dot}(X_n, X_n) \end{bmatrix}$$

$$\begin{aligned} & COV(X_1, X_2) \\ &= \frac{\sum(\text{각 } X_1 \text{의 편차})(\text{각 } X_2 \text{의 편차})}{n(-1)} \\ &= \frac{1}{n(-1)} \sum (X_{1i} - \bar{x}_1)(X_{2i} - \bar{x}_2) \end{aligned}$$

2.4.2 탐색적 데이터 분석과 상관분석

■ 공분산

A 인터넷 쇼핑몰의 웹사이트 이용정보 예시

고객	웹사이트 접속시간(분)	구매비용(천원)
1	20	12
2	55	50
3	40	33
4	70	85
5	35	25
평균	44	41

- 각 고객의 웹사이트 접속시간 편차: -24, 11, -4, 26, -9
- 각 고객의 구매비용 편차: -29, 9, -8, 44, -16

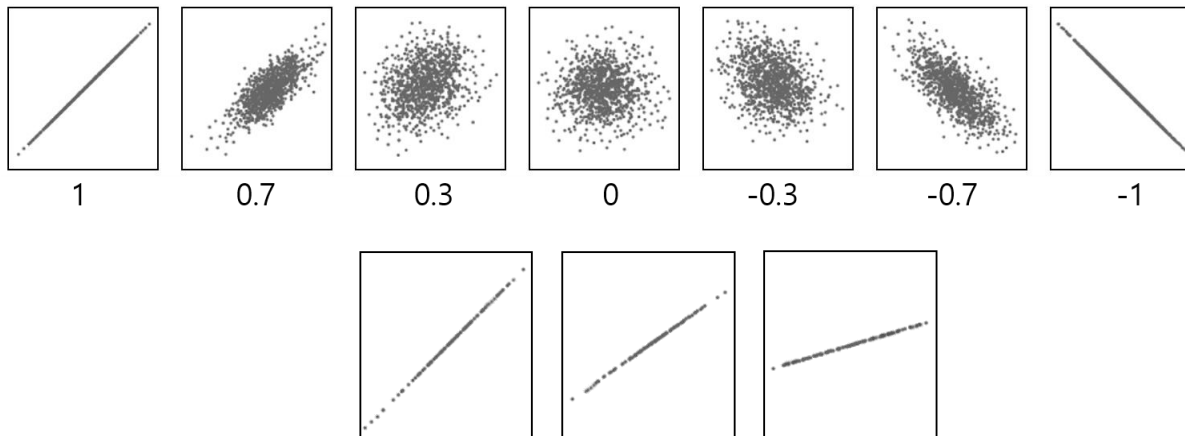
$$\frac{(-24 \times -29) + (11 \times 9) + (-4 \times -8) + (26 \times 44) + (-9 \times -16)}{4} = 528.75$$

2.4.2 탐색적 데이터 분석과 상관분석

■ 피어슨 상관계수

$$P(X_1, X_2) = \frac{COV(X_1, X_2)}{\sqrt{Var(X_1) Var(X_2)}}$$

범위(절댓값)	단계
0	Zero
0.1~0.3	Weak
0.4~0.6	Moderate
0.7~0.9	Strong
1	Perfect



2.5 인공지능 개발에 많이 쓰는 라이브러리

■ 기초 라이브러리

- 판다스(Pandas): 데이터 프레임(DataFrame)이라는 자료구조 지원
- 넘파이(Numpy): 다차원 배열 지원

■ 시각화 라이브러리

- 맷플롯립(Matplotlib): Python에서 가장 많이 쓰이는 데이터 시각화 라이브러리
- 시본(seaborn): matplotlib을 기반으로 다양한 색 테마, 차트 기능을 추가한 라이브러리
- 플롯나인(Plotnine): R의 ggplot2에 기반해 그래프를 그려주는 라이브러리

■ 인공지능 라이브러리

- 사이킷런(Scikit-learn): 고전적인 기계 학습 지원
- 텐서플로(TensorFlow): 딥러닝 지원
- 케라스(Keras): 텐서플로를 한 단계 추상화한 라이브러리
- 파이토치(PyTorch): 딥러닝 라이브러리

2.5.1 판다스(Pandas) 라이브러리

■ 정의

- pandas는 데이터 분석용 라이브러리로 데이터를 다루는 패키지 중 하나
- 데이터 분석을 위한 효율적인 데이터 구조를 제공하며, 1차원 배열 형태의 데이터 구조인 Series와 2차원 배열 형태의 데이터 구조인 Data Frame

■ 특징

- pandas는 파이썬에서 가장 널리 사용되는 데이터 분석 라이브러리로 데이터 프레임(DataFrame)이라는 자료구조
- 데이터 프레임은 엑셀의 스프레드시트와 유사한 형태이며 파이썬으로 데이터를 쉽게 처리할 수 있음
- 데이터를 분석 및 조작을 위한 라이브러리
- 수치형 테이블과 시계열 데이터를 조작하고 운영하기 위한 데이터를 제공
- 시계열 데이터와 비시계열 데이터를 함께 다룰 수 있는 통합 자료 구조
- 누락된 데이터를 유연하게 처리할 수 있는 기능
- SQL 같은 일반 데이터베이스처럼 데이터를 합치고 관계연산을 수행하는 기능

2.5.1 넘파이(Numpy) 라이브러리

■ 정의

- Numerical python의 줄임말, 수치 계산을 위한 파이썬 라이브러리
- 배열(array) 개념으로 변수 사용. 벡터, 행렬 연산 가능

■ 특징

- Python의 선형대수 라이브러리로 C언어 기반으로 작성되어서 다차원 배열이나 행렬의 계산 속도가 압도적으로 빠름.
- 넘파이에(np). arrange(n)를 붙여주면 기존 파이썬의. range(n)처럼 0부터 (n-1)까지 n개의 순차 배열을 만들 수 있음

```
# arange, 10까지 순차적 배열 생성
rangeArr=np.arange(10)
print("range\n",rangeArr,'\n')

# zeros, 0으로 채워진 배열 생성
zeroArr=np.zeros((3,2),dtype=int)
print("zeros\n",zeroArr,'\n')

# ones, 1로 채워진 배열 생성
oneArr=np.ones((4,3))
print("ones\n",oneArr)
```

```
[Output]

range
[0 1 2 3 4 5 6 7 8 9]

zeros
[[0 0]
 [0 0]
 [0 0]]

ones
[[1. 1. 1.]
 [1. 1. 1.]
 [1. 1. 1.]
 [1. 1. 1.]
```

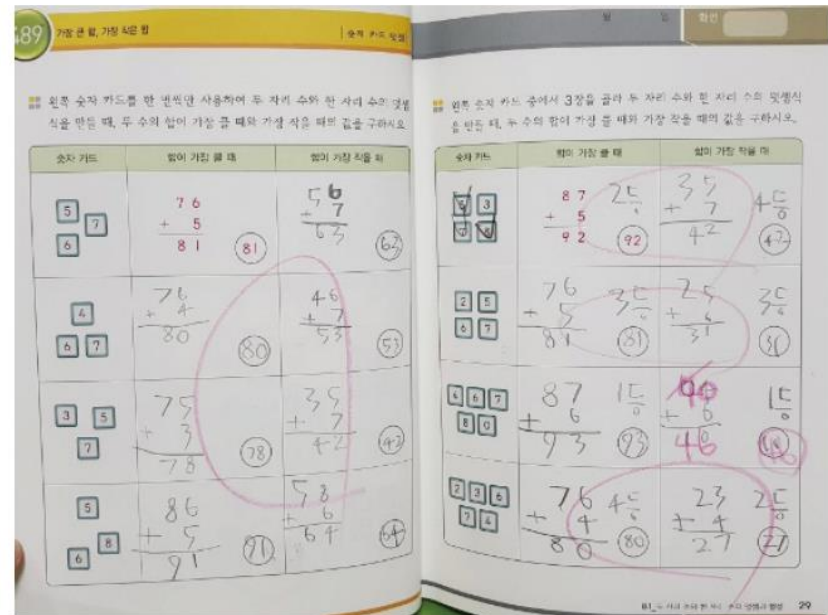
2.6 인공지능 프로그래밍 예제 1: 셈 지능

■ 인간의 셈 지능

- 느리고 정확도에 한계
- 세계주산암산대회(2018) 우승자는 초당 1.3개의 사칙 연산 수행

$$\begin{array}{r} 48 \\ + 34 \\ \hline 82 \end{array}$$

- ① 일의 자리를 더한다. 결과 2를 기록하고 올림수 1을 왼쪽 자리로 보낸다.
- ② 십의 자리를 더한 결과에 올림수를 더한다. 결과 8을 기록한다.



2.6 인공지능 프로그래밍 예제 1: 셈 지능

■ 컴퓨터의 셈 지능

- 무척 빠르고 정확함. 컴퓨터 하드웨어에 내장되어 있음
- [프로그램 2-2]는 컴퓨터의 셈 지능 속도를 측정
 - 10억 개 덧셈을 11.8초 만에 수행

```
01 import time                # 시간 측정 라이브러리
02
03 start=time.time()          # 시작 직전 시각을 기록
04 sum=0
05 for i in range(1,100000001): # 1억 번 반복
06     sum=sum+i
07
08 end=time.time()             # 끝난 직후 시각을 기록
09
10 print('1+2+...+100000000=', sum)
11 print('소요 시간은 ', end-start, '초입니다.') # 시간 차이를 계산하여 출력
```

1+2+...+100000000=4999999950000000
소요 시간은 11.80145525932312초입니다.

2.6 인공지능 프로그래밍 예제 2: 인공지능 기자

■ 뉴스 기사를 자동으로 작성하는 인공지능

- <LA 타임스>의 퀘이커봇은 지진 전문 로봇 기자
- <연합뉴스>의 사커봇은 프리미어 리그 속보 로봇 기자
- 현재 높은 수준의 자연어 처리와 추론 등의 기술을 사용하여 인간 기사를 능가하는 수준
- 여기서는 원시적인 로봇 기자 프로그래밍

2.6 인공지능 프로그래밍 예제 2: 인공지능 기자

■ 손흥민의 경기 결과를 기사로 작성하는 '단순한' 인공지능 프로그램

- 04~10행: 경기 결과를 입력하는 곳
- 13~31행: 기사를 합성하는 곳
- 13행은 datetime 함수를 이용하여 속보 입력 시간을 자동으로 붙임

프로그램 2-3

손흥민 선수의 경기 속보를 전담하는 로봇 기자

```
01  from datetime import datetime
02
03  # 경기 결과 입력 받는 곳
04  place=input("경기가 열린 곳은? ")
05  time=input("경기가 열린 시간은? ")
06  opponent=input("상대 팀은? ")
07  goals=input("손흥민은 몇 골을 넣었나요? ")
08  aids=input("도움은 몇 개인가요? ")
09  score_me=input("손흥민 팀이 넣은 골 수는? ")
10  score_you=input("상대 팀이 넣은 골 수는? ")
11
12  # 기사 작성하는 곳
13  news="[프리미어 리그 속보("+str(datetime.now())+")]\n"
14  news=news+"손흥민 선수는 "+place+"에서 "+time+"에 열린 경기에 출전하였습니다. "
15  news=news+"상대 팀은 "+opponent+"입니다. "
```


2.6 인공지능 프로그래밍 예제 2: 인공지능 기자

■ (... 앞에서 계속)

- 17~22행: 두 팀의 골 수에 따라 경우를 나누어 기사 작성
- 24~31행: 손흥민의 골과 도움 수에 따라 변화를 주어 단조로움 해소

```
16
17 if score_me>score_you:
18     news=news+"손흥민 선수의 팀이 "+score_me+"골을 넣어 "+score_you+"골을 넣은 상대 팀
    을 이겼습니다. "
19 elif score_me<score_you:
20     news=news+"손흥민 선수의 팀이 "+score_me+"골을 넣어 "+score_you+"골을 넣은 상대 팀
    에게 졌습니다. "
21 else:
22     news=news+"두 팀은 "+score_me+"대"+score_you+"로 비겼습니다. "
23
24 if int(goals)>0 and int(aids)>0:
25     news=news+"손흥민 선수는 "+goals+"골에 도움 "+aids+"개로 승리를 크게 이끌었습니다. "
26 elif int(goals)>0 and int(aids)==0:
27     news=news+"손흥민 선수는 "+goals+"골로 승리를 이끌었습니다. "
28 elif int(goals)==0 and int(aids)>0:
29     news=news+"손흥민 선수는 골은 없지만 도움 "+aids+"개로 승리하는 데 공헌하였습니다. "
30 else:
31     news=news+"아쉽게도 이번 경기에서 손흥민의 발끝은 침묵을 지켰습니다. "
32
33 print(news)
```

2.6 인공지능 프로그래밍 예제 2: 인공지능 기자

■ 경기 결과 입력과 작성된 기사 예

경기가 열린 곳은? 런던 스타디움

경기가 열린 시간은? 8월 6일 오후 7시

상대 팀은? 맨체스터 유나이티드

손흥민은 몇 골을 넣었나요? 2

도움은 몇 개인가요? 1

손흥민 팀이 넣은 골 수는? 4

상대 팀이 넣은 골 수는? 2

[프리미어 리그 속보(2020-08-07 09:33:14.128724)]

손흥민 선수는 런던 스타디움에서 12월 12일 오후 7시에 열린 경기에 출전하였습니다. 상대 팀은 맨체스터 유나이티드입니다. 손흥민 선수의 팀이 4골을 넣어 2골을 넣은 상대 팀을 이겼습니다. 손흥민 선수는 2골에 도움 1개로 승리를 크게 이끌었습니다.

2.6 인공지능 프로그래밍 예제 2: 인공지능 기자

■ 확장

- 문장을 랜덤 선택하여 단조로움 해소

```

18 wording=["상대 팀을 이겼습니다.", "상대를 대상으로 통쾌한 승리를 거머쥐었습니다.", "상대 팀
   을 꺾었습니다."]
19 news=news+"손흥민 선수의 팀이"+score_me+"골을 넣어"+score_you+"골을 넣은"+ random.
   choice(wording)

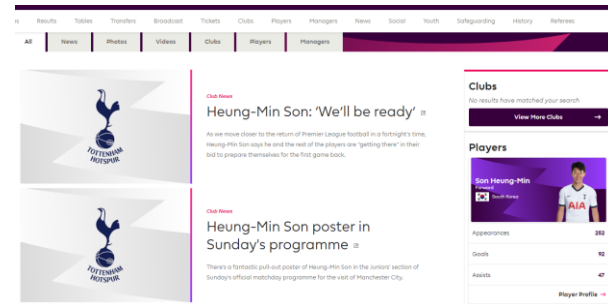
```

- 자동으로 경기 결과를 크롤링하고 기사를 업로드

- 프리미어 웹 사이트 파싱
- 웹 프로그래밍(자바스크립트)

- 수준 높은 기사 작성

- 자연어 처리 기술과 word2vec과 같은 언어 모델 활용



2.6 인공지능 프로그래밍 예제 3: 더 똑똑한 인공지능 기자

■ 언어 처리 라이브러리를 활용하면 더 똑똑한 인공지능 기자

- 이 절은 가장 단순한 축에 속하는 TTS(text-to-speech) 기능 활용(gtts 라이브러리)

```
from datetime import datetime

# 경기 결과 입력 받는 곳
place=input("경기가 열린 곳은? ")
time=input("경기가 열린 시간은? ")
opponent=input("상대 팀은? ")
goals=input("손흥민은 몇 골을 넣었나요? ")
aids=input("도움은 몇 개인가요? ")
score_me=input("손흥민 팀이 넣은 골 수는? ")
score_you=input("상대 팀이 넣은 골 수는? ")

# 기사 작성하는 곳
news="[프리미어 리그 속보 (" +str(datetime.now())+"))]\n"
news=news+"손흥민 선수는 " +place+"에서 " +time+"에 열린 경기에 출전하였습니다. "
news=news+"상대 팀은 " +opponent+"입니다. "

if score_me>score_you:
    news=news+"손흥민 선수의 팀이 " +score_me+"골을 넣어 " +score_you+"골을 넣은 상대 팀을 이겼습니다. "
elif score_me<score_you:
    news=news+"손흥민 선수의 팀이 " +score_me+"골을 넣어 " +score_you+"골을 넣은 상대 팀에게 졌습니다. "
else:
    news=news+"두 팀은 " +score_me+"대 " +score_you+"로 비겼습니다. "

if int(goals)>0 and int(aids)>0:
    news=news+"손흥민 선수는 " +goals+"골에 도움 " +aids+"개로 승리를 크게 이끌었습니다. "
elif int(goals)>0 and int(aids)==0:
    news=news+"손흥민 선수는 " +goals+"골로 승리를 이끌었습니다. "
elif int(goals)==0 and int(aids)>0:
    news=news+"손흥민 선수는 골은 없지만 도움 " +aids+"개로 승리하는 데 공헌하였습니다. "
else:
    news=news+"아쉽게도 이번 경기에서 손흥민의 발끝은 침묵을 지켰습니다. "
print(news)

# 음성으로 들려주는 곳
!pip install gtts playsound
from gtts import gTTS
import playsound

tts=gTTS(text=news,lang='ko') # 문자열 news를 위한 한국어 음성 합성
tts.save("/content/news_Son.mp3")
```

실습시
복사해서
사용

2.6.1 TTS 프로그래밍

- [프로그램 2-4]: 소리를 들려줄 수 있게 [프로그램 2-3] 확장

프로그램 2-4

말하는 로봇 기자

```
01  ... } [프로그램 2-3]
33
34
35  # 음성으로 들려주는 곳
36  from gtts import gTTS
37  import playsound
38
39  tts=gTTS(text=news, lang='ko')      # 문자열 news를 위한 한국어 음성 합성
40  tts.save("news_Son.mp3")
```

2.7 실습

- 실습 1. 난수 2개를 더하는 프로그램을 $[0,1]$ 범위의 실수 난수를 생성해서 더하는 프로그램으로 수정하시오.

- 실습2. 컴퓨터의 지능 센 속도 코드를 $[0,1]$ 범위의 실수 난수를 생성하여 더하는 연산을 1억 번 반복하는 프로그램으로 수정하고 걸리는 시간을 측정하시오.

- 실습 3. 인공지능 기사 코드를
 - 1) 기사 입력시간이 분 단위까지만 기록되도록 수정하시오.
 - 2) 손흥민 선수의 활약 내용이 승리와 패배를 고려하도록 수정하시오.