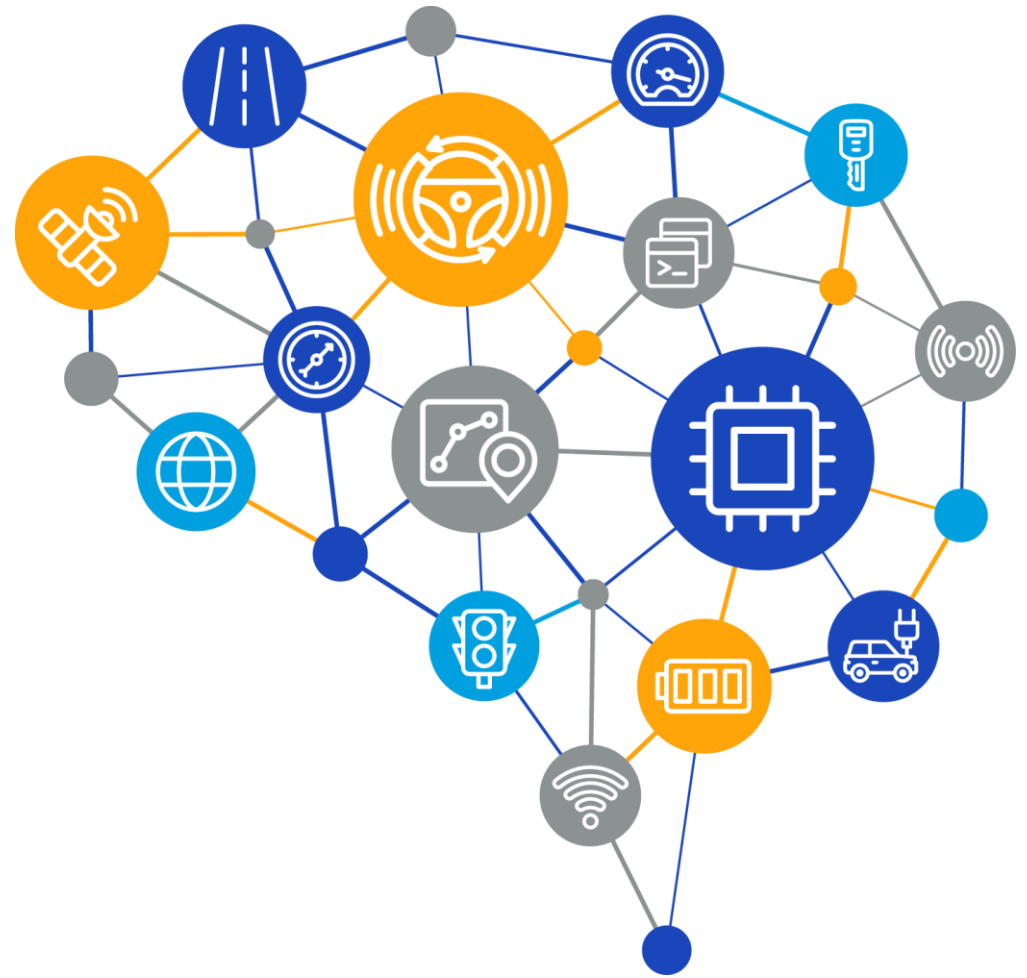


코드스페이스 템플릿 저장소 생성 후 설정 변경 코드스페이스 생성

강환수 교수

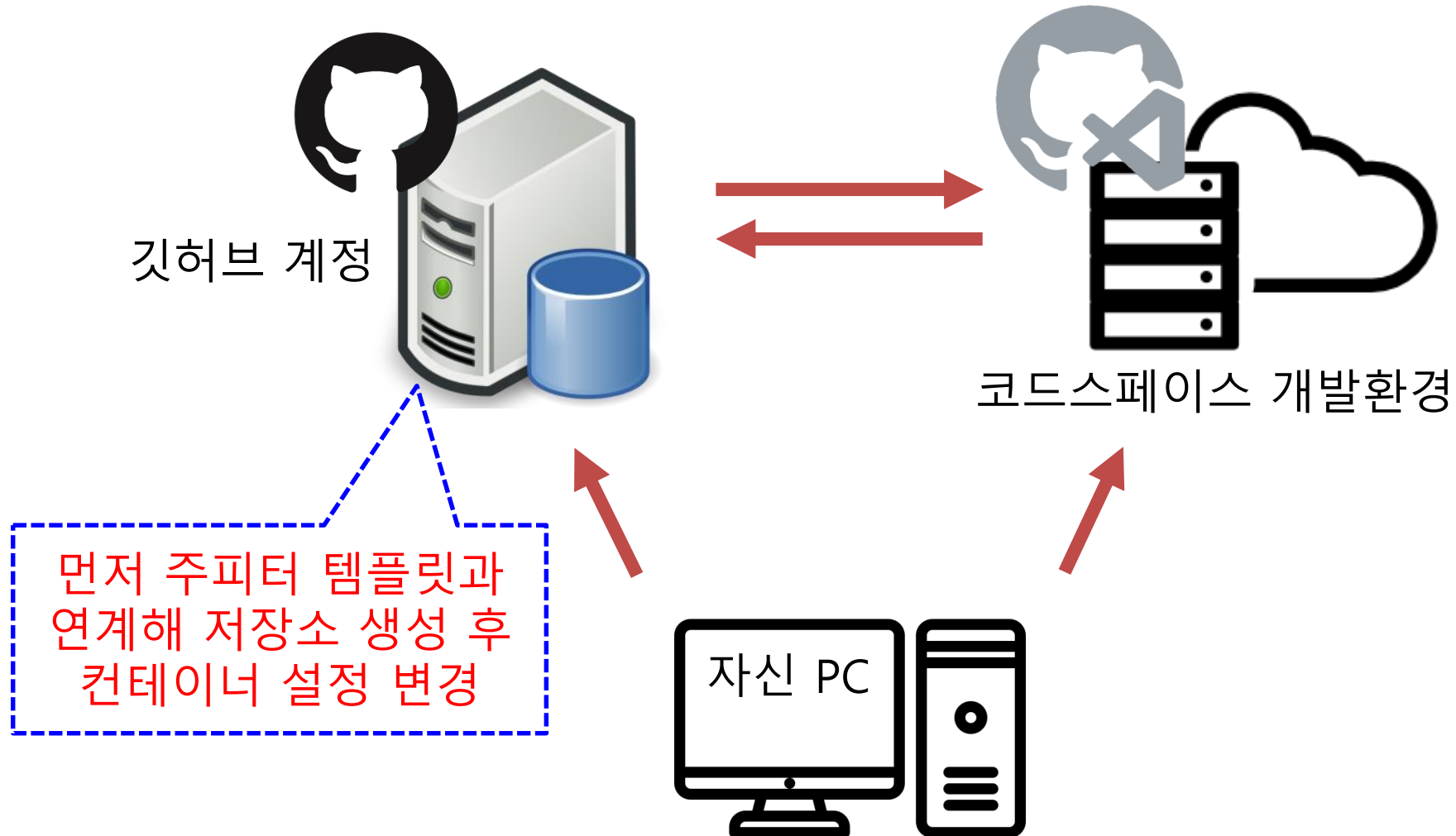


AI Experts
Who Lead
The Future

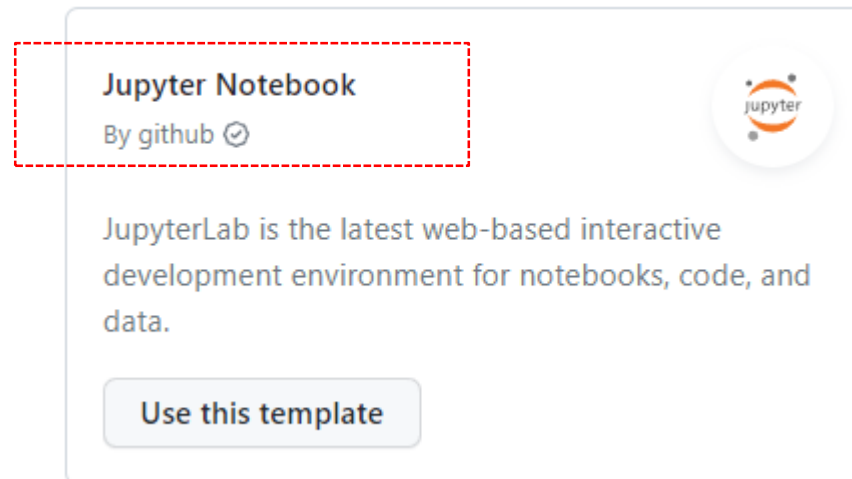
01

코드스페이스 주피터 접속

- 깃허브에서 실행하는 클라우드 개발환경



- 파이썬의 주피터 노트북을 지원하는 템플릿
 - Jupyter Notebook 클릭



- <https://github.com/github/codespaces-jupyter>

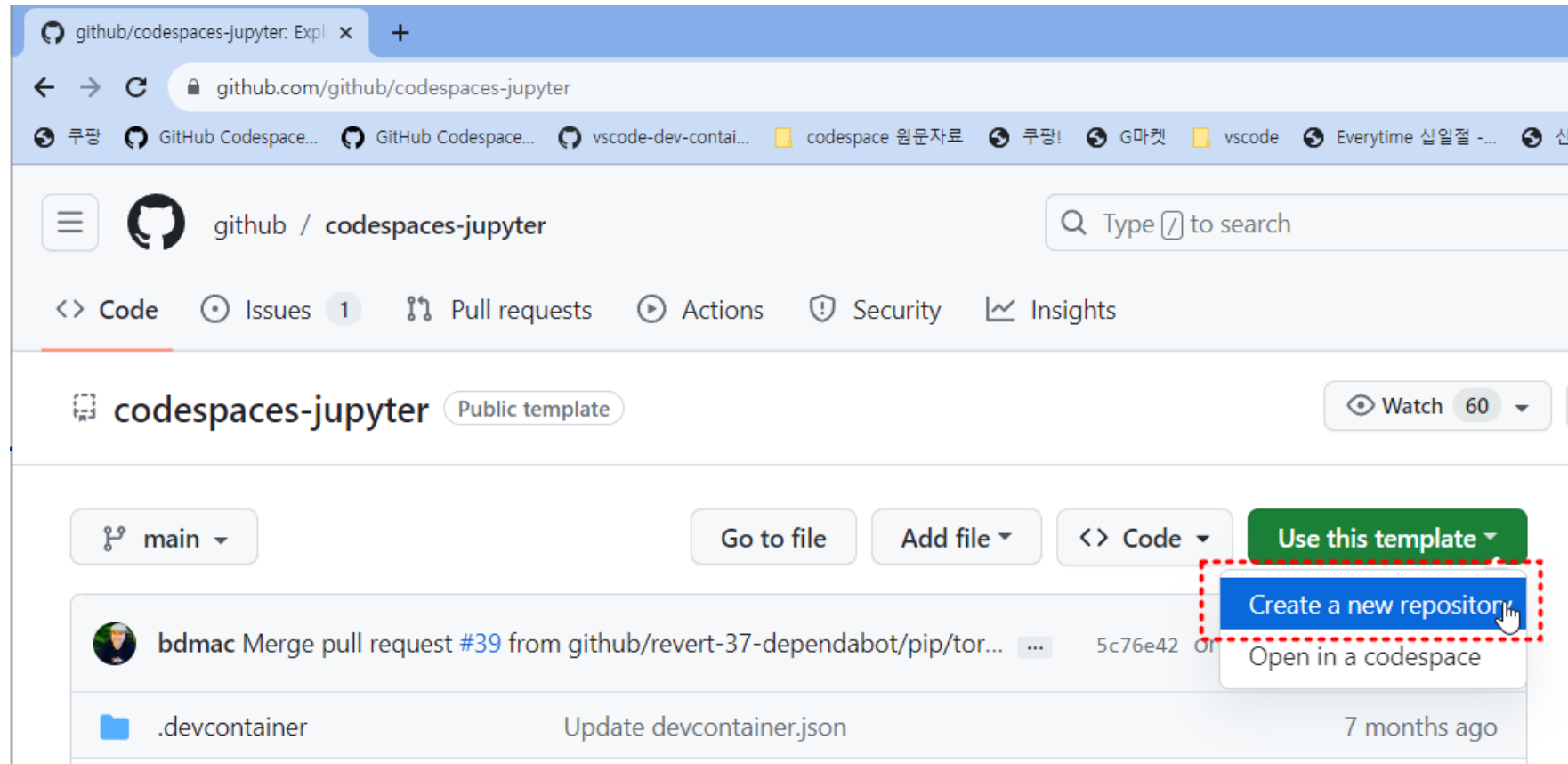
The screenshot shows the GitHub repository page for `codespaces-jupyter`, which is a public template. The repository has 61 watches, 701 forks, and 378 stars. The main branch is selected, showing 2 branches and 0 tags. A recent pull request by `bdmac` is visible. The commit history lists several updates to the devcontainer, notebooks, and README. The README section, titled "GitHub Codespaces ❤️ Jupyter Notebooks", welcomes users and explains how to use the codespace for exploring Python and Jupyter notebooks. It mentions that the codespace is self-contained and can be published as a repository on GitHub.

AI Experts
Who Lead
The Future

02

코드스페이스 템플릿을 사용해
저장소 생성

- **[Create a new repository]를 선택**
 - 템플릿의 내용을 그대로 복제한 저장소를 생성하기 위해



New repository x chatkang-orange-space-trout-g x +

github.com/new?template_name=codespaces-jupyter&template_owner=github

☰ GitHub 🔍 Type / to search >_ + -

Create a new repository

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

Required fields are marked with an asterisk ().*

Repository template

github/codespaces-jupyter

Start your repository with a template repository's contents.

☐ Include all branches

Copy all branches from github/codespaces-jupyter and not just the default branch.

Owner * Repository name *

chatkang / code-jupyter

code-jupyter is available.

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

Description (optional)

텐서플로를 사용한 인공지능 코딩

☒ Public

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

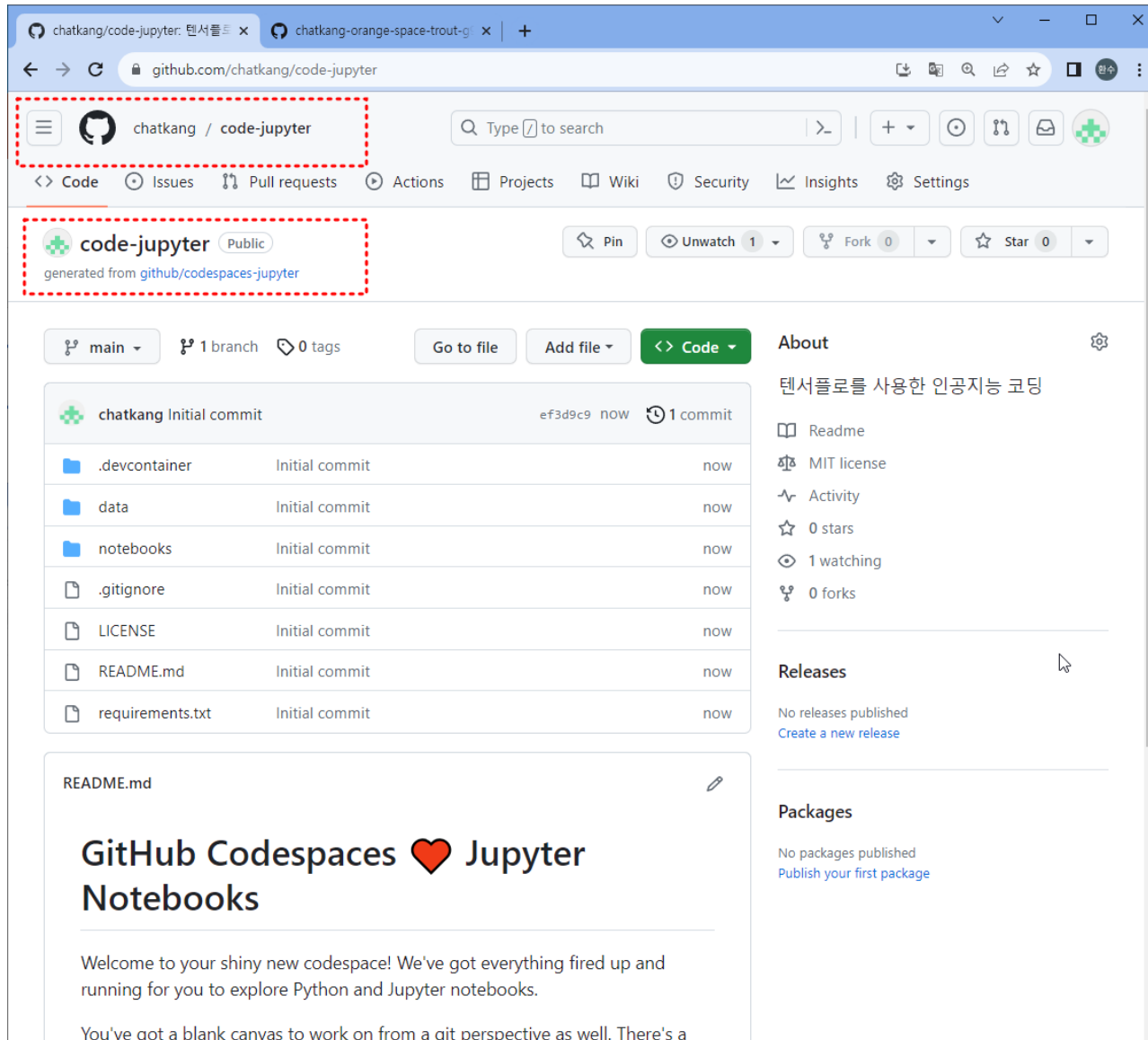
☐ Private

You choose who can see and commit to this repository.

📌 You are creating a public repository in your personal account.

Create repository

- 주인은 자신



- Requirement.txt 파일 수정

```
Code Blame 9 lines (9 loc) · 144 Bytes

1 ipywidgets==7.7.1
2 matplotlib==3.7.0
3 numpy==1.24.2
4 pandas==1.5.3
5 torch==1.12.1
6 torchvision==0.13.1
7 tqdm==4.64.0
8 tensorflow==2.10.0
9 Faker==15.1.0
```

#기존 설치된 내용들을 담아서 처리
\$ pip freeze > requirements.txt

한번에 설치하기
\$ pip install -r requirements.txt

파일 수정 후 커밋

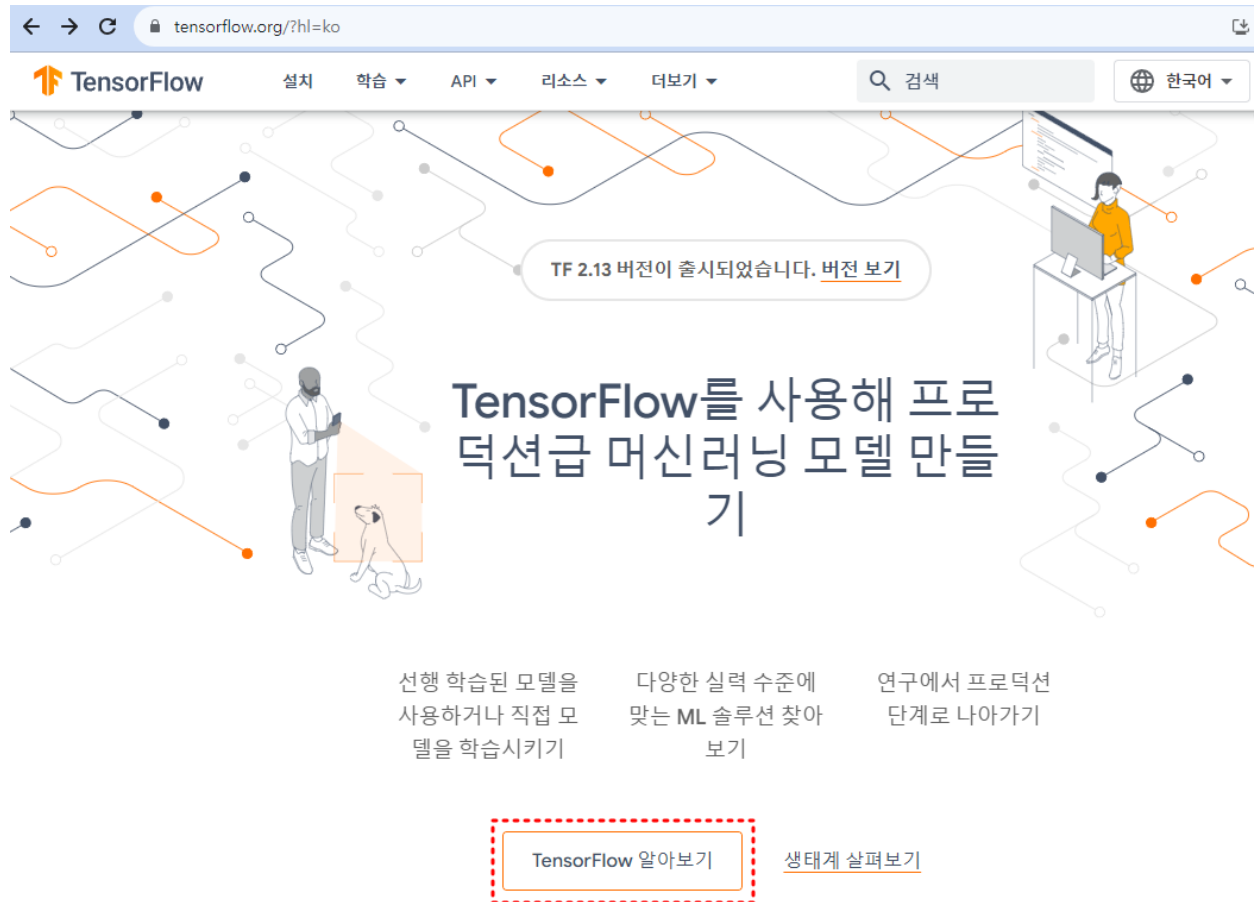
오픈소스 소프트웨어를 위한 깃과 깃허브 Python language

The screenshot shows a GitHub repository named 'chatkang / code-jupyter'. The file 'requirements.txt' is being edited in the 'main' branch. The file content is as follows:

```
1 ipywidgets==7.7.1
2 matplotlib==3.7.0
3 numpy==1.24.2
4 pandas==1.5.3
5 torch==1.12.1
6 torchvision==0.13.1
7 tqdm==4.64.0
8 tensorflow==2.10.0
9 Faker==15.1.0
10
```

A 'Commit changes' dialog box is open, showing the commit message 'Update requirements.txt' and an empty 'Extended description' field. The dialog has two options: 'Commit directly to the main branch' (selected) and 'Create a new branch for this commit and start a pull request'. The 'Commit changes' button is highlighted with a red dashed border.

- Tensorflow.org 접속
 - 알아보기



TensorFlow는 머신러닝을 위한 엔드 투 엔드 오픈소스 플랫폼입니다.

TensorFlow를 사용하면 초보자와 전문가 모두 머신러닝 모델을 쉽게 만들 수 있습니다. 시작하려면 아래의 섹션을 참조하세요.

튜토리얼 보기

튜토리얼에서는 완벽한 엔드 투 엔드 예제와 함께 TensorFlow를 사용하는 방법을 보여줍니다.

가이드 보기

가이드는 TensorFlow의 개념과 구성요소에 대해 설명합니다.



초보자용

사용자에게 친숙한 Sequential API로 시작하는 것이 가장 좋습니다. 구성요소를 연결하여 모델을 만들 수 있습니다. 아래의 'Hello World' 예제를 실행한 다음 튜토리얼을 방문하여 자세한 내용을 알아보세요.

ML에 관해 배워보려면 [교육 페이지](#)를 확인하세요. 엄선된 커리큘럼으로 기본적인 ML 분야의 역량을 키워보세요.

전문가용

Subclassing API는 고급 연구에 사용할 수 있는 실행 시 정의되는(define-by-run) 인터페이스를 제공합니다. 모델 클래스를 만든 다음 전달 패스를 명령형으로 만드세요. 맞춤 레이어, 활성화, 학습 루프를 손쉽게 작성할 수 있습니다. 아래의 'Hello World' 예제를 실행한 다음 [튜토리얼](#)을 방문하여 자세한 내용을 알아보세요.

```
import tensorflow as tf
mnist = tf.keras.datasets.mnist

(x_train, y_train), (x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0

model = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(10, activation='softmax')
])

model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
```

```
class MyModel(tf.keras.Model):
    def __init__(self):
        super(MyModel, self).__init__()
        self.conv1 = Conv2D(32, 3, activation='relu')
        self.flatten = Flatten()
        self.d1 = Dense(128, activation='relu')
        self.d2 = Dense(10, activation='softmax')

    def call(self, x):
        x = self.conv1(x)
        x = self.flatten(x)
        x = self.d1(x)
        return self.d2(x)


model = MyModel()
```

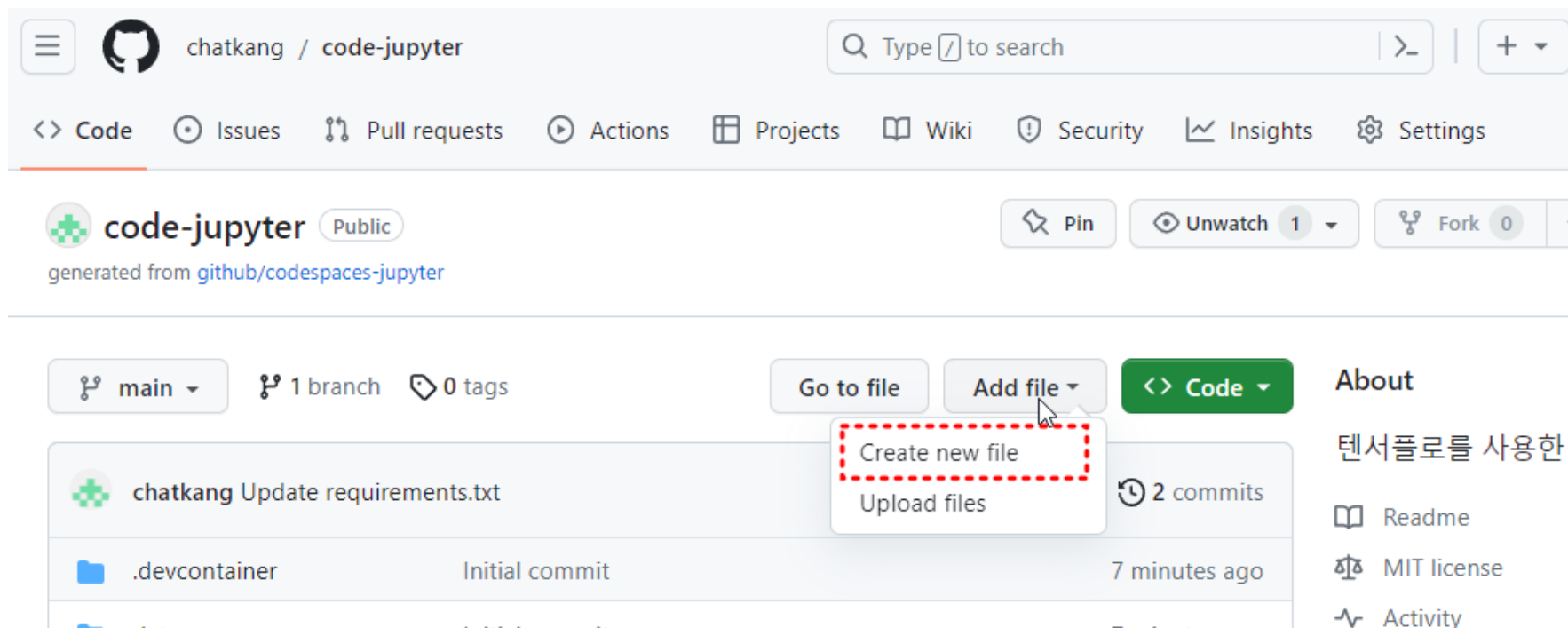
폴더 및 파일 추가

오픈소스 소프트웨어를 위한 깃과 깃허브 Python language

- 입력

- `tf-code/mnist.py`





 `code-jupyter` / in `main`



chatkang / code-jupyter


Type to search

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

 **code-jupyter** Public  Pin  Unwatch 1  Fork 0

generated from [github/codespaces-jupyter](https://github.com/codespaces-jupyter)

main 1 branch 0 tags

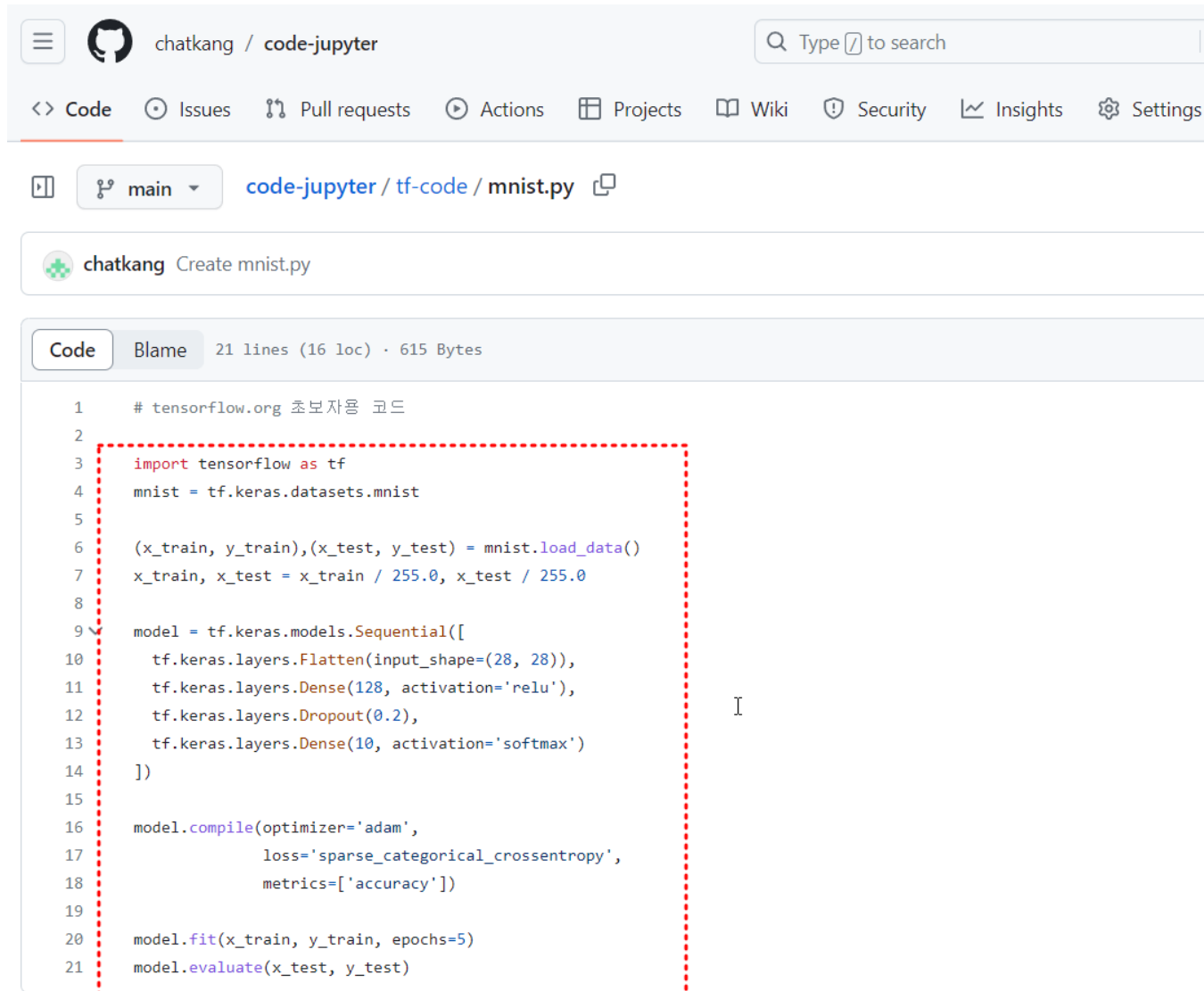
Go to file Add file 

Create new file
Upload files

chatkang Update requirements.txt 2 commits 7 minutes ago

.devcontainer Initial commit

About
텐서플로를 사용한
Readme
MIT license
Activity



```
1 # tensorflow.org 초보자용 코드
2
3 import tensorflow as tf
4 mnist = tf.keras.datasets.mnist
5
6 (x_train, y_train), (x_test, y_test) = mnist.load_data()
7 x_train, x_test = x_train / 255.0, x_test / 255.0
8
9 model = tf.keras.models.Sequential([
10     tf.keras.layers.Flatten(input_shape=(28, 28)),
11     tf.keras.layers.Dense(128, activation='relu'),
12     tf.keras.layers.Dropout(0.2),
13     tf.keras.layers.Dense(10, activation='softmax')
14 ])
15
16 model.compile(optimizer='adam',
17               loss='sparse_categorical_crossentropy',
18               metrics=['accuracy'])
19
20 model.fit(x_train, y_train, epochs=5)
21 model.evaluate(x_test, y_test)
```


The screenshot shows the GitHub repository page for 'chatkang / code-jupyter'. The repository is public and was generated from 'github/codespaces-jupyter'. The main branch is 'main', and there is 1 branch and 0 tags. The repository has 3 commits, with the most recent commit '76e8ddd' made 4 minutes ago. The commit history table shows three commits: 'Create mnist.py' (4 minutes ago), '.devcontainer' (21 minutes ago), and 'data' (21 minutes ago). The 'About' section on the right lists the README, MIT license, and 0 stars.

Commit	Author	Time
76e8ddd	chatkang	4 minutes ago
	Initial commit	21 minutes ago
	Initial commit	21 minutes ago
	Initial commit	21 minutes ago

Commits

The screenshot shows the 'Commits' section on the GitHub repository page. It displays a list of commits on the 'main' branch, filtered for July 29, 2023. The commits are: 'Create mnist.py' (4 minutes ago), 'Update requirements.txt' (16 minutes ago), and 'Initial commit' (21 minutes ago). Each commit is marked as 'Verified'. The commit '76e8ddd' is highlighted with a red dashed box, and a tooltip 'View commit details' is shown over it.

Commit	Author	Time	Verified	Commit ID
Create mnist.py	chatkang	4 minutes ago	Verified	76e8ddd
Update requirements.txt	chatkang	16 minutes ago	Verified	c60092f
Initial commit	chatkang	21 minutes ago	Verified	ef3d9c9

• 파일 차이

Commit

Create mnist.py

main

chatkang committed 5 minutes ago Verified

1 parent c60092f commit 76e8ddd

Browse files

Showing 1 changed file with 21 additions and 0 deletions.

21 tf-code/mnist.py

@@ -0,0 +1,21 @@

```
1 + # tensorflow.org 초보자용 코드
2 +
3 + import tensorflow as tf
4 + mnist = tf.keras.datasets.mnist
5 +
6 + (x_train, y_train), (x_test, y_test) = mnist.load_data()
7 + x_train, x_test = x_train / 255.0, x_test / 255.0
8 +
9 + model = tf.keras.models.Sequential([
10 +     tf.keras.layers.Flatten(input_shape=(28, 28)),
11 +     tf.keras.layers.Dense(128, activation='relu'),
12 +     tf.keras.layers.Dropout(0.2),
13 +     tf.keras.layers.Dense(10, activation='softmax')
14 + ])
15 +
16 + model.compile(optimizer='adam',
17 +               loss='sparse_categorical_crossentropy',
18 +               metrics=['accuracy'])
19 +
20 + model.fit(x_train, y_train, epochs=5)
21 + model.evaluate(x_test, y_test)
```


AI Experts
Who Lead
The Future

03

코드스페이스 생성









코드스페이스 생성

오픈소스 소프트웨어를 위한 깃과 깃허브 Python language

 **Data-Science** Public Pin Unwatch 1

generated from [github/codespaces-jupyter](#)

main 1 branch 0 tags Go to file Add file <> Code

 chatkang	Update README.md
	.devcontainer Initial commit
	data Initial commit
	notebooks Initial commit
	.gitignore Initial commit
	LICENSE Initial commit
	README.md Update README.
	requirements.txt Initial commit

Local **Codespaces**

Codespaces
Your workspaces in the cloud

No codespaces
You don't have any codespaces with this repository checked out

[Create codespace on main](#)

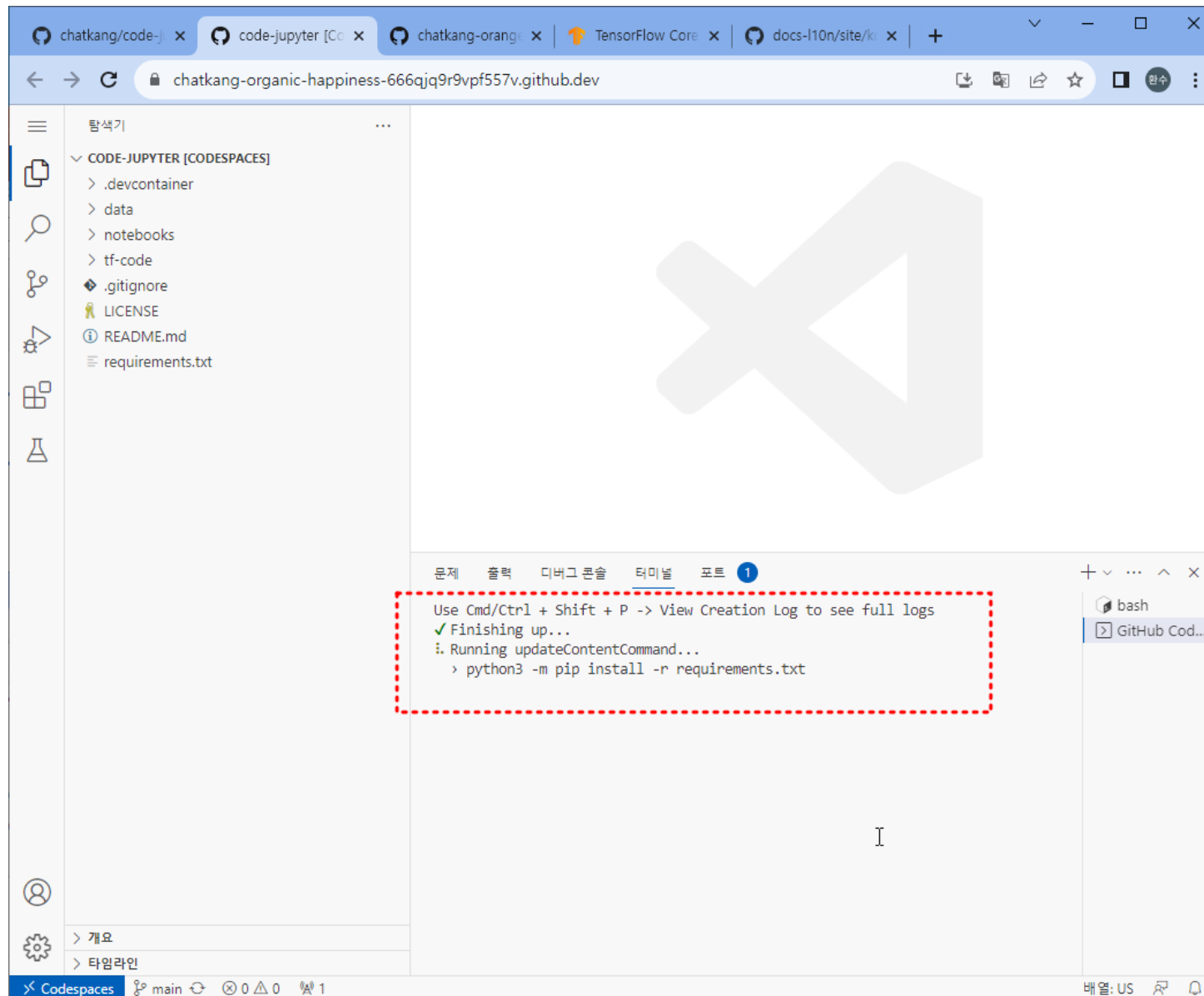
[Learn more about codespaces...](#)

Codespace usage for this repository is paid for by chatkang

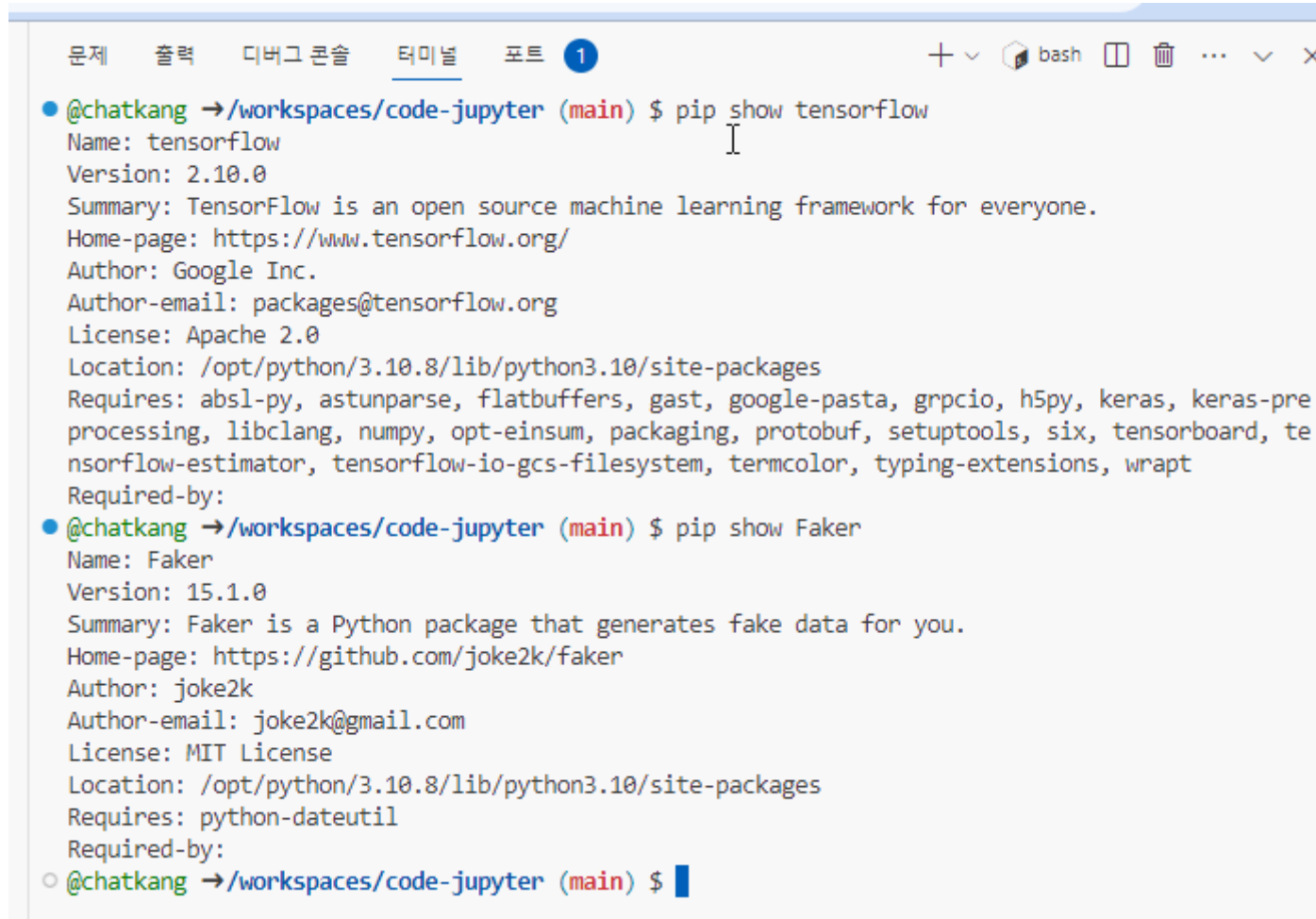
Setting up your codespace

```
✓ Image found.  
🔧 Building container...
```

💡 **Tip** Keep your sensitive information safe with native secret support. [Learn more](#)



- 모듈 검사



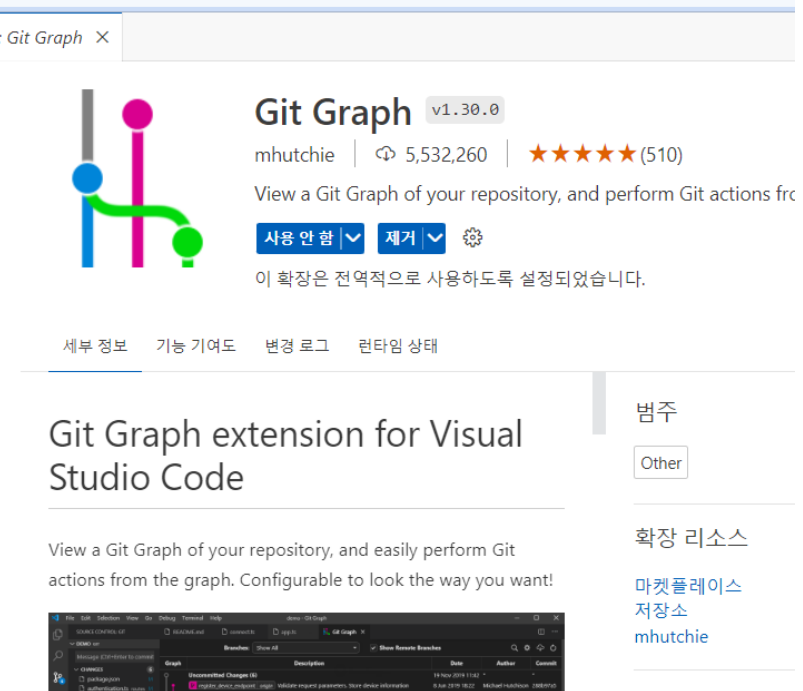
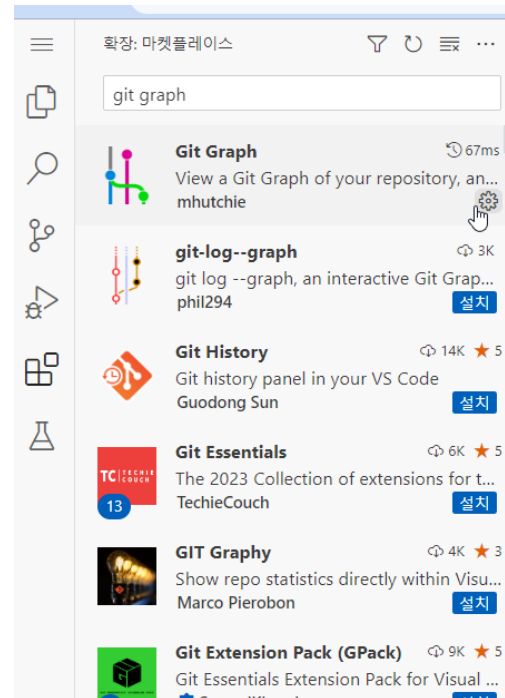
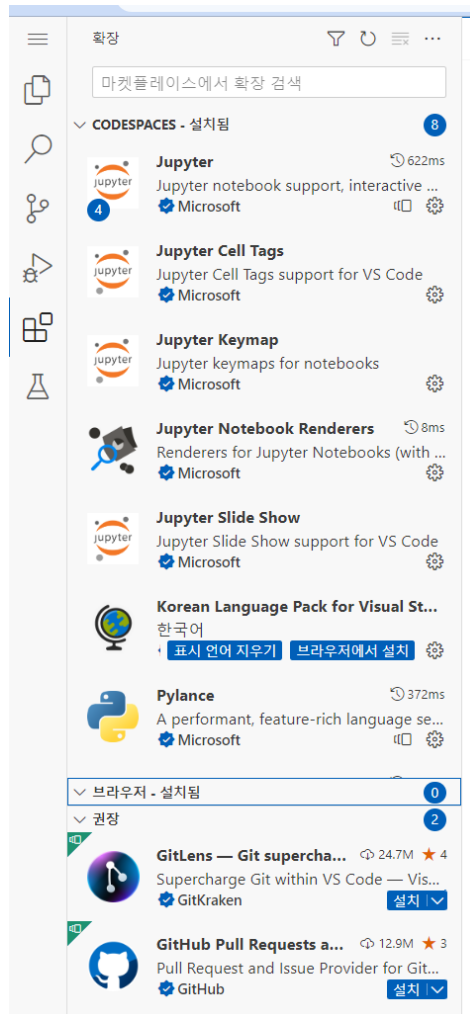
```
문제 출력 디버그 콘솔 터미널 포트 1
+ v bash
• @chatkang →/workspaces/code-jupyter (main) $ pip show tensorflow
Name: tensorflow
Version: 2.10.0
Summary: TensorFlow is an open source machine learning framework for everyone.
Home-page: https://www.tensorflow.org/
Author: Google Inc.
Author-email: packages@tensorflow.org
License: Apache 2.0
Location: /opt/python/3.10.8/lib/python3.10/site-packages
Requires: absl-py, astunparse, flatbuffers, gast, google-pasta, grpcio, h5py, keras, keras-pre
processing, libclang, numpy, opt-einsum, packaging, protobuf, setuptools, six, tensorboard, te
nsorflow-estimator, tensorflow-io-gcs-filesystem, termcolor, typing-extensions, wrapt
Required-by:
• @chatkang →/workspaces/code-jupyter (main) $ pip show Faker
Name: Faker
Version: 15.1.0
Summary: Faker is a Python package that generates fake data for you.
Home-page: https://github.com/joke2k/faker
Author: joke2k
Author-email: joke2k@gmail.com
License: MIT License
Location: /opt/python/3.10.8/lib/python3.10/site-packages
Requires: python-dateutil
Required-by:
○ @chatkang →/workspaces/code-jupyter (main) $
```

- 파일 구성

- 폴더이름: CODE-JUPYTER [CODESPACES]
- 4개의 폴더
- 여러 개의 파일

```
tf-code > mnist.py > ...
1 # tensorflow.org 초보자용 코드
2
3 import tensorflow as tf
4 mnist = tf.keras.datasets.mnist
5
6 (x_train, y_train), (x_test, y_test) = mnist.load_data()
7 x_train, x_test = x_train / 255.0, x_test / 255.0
8
9 model = tf.keras.models.Sequential([
10     tf.keras.layers.Flatten(input_shape=(28, 28)),
11     tf.keras.layers.Dense(128, activation='relu'),
12     tf.keras.layers.Dropout(0.2),
13     tf.keras.layers.Dense(10, activation='softmax')
14 ])
15
16 model.compile(optimizer='adam',
17               loss='sparse_categorical_crossentropy',
18               metrics=['accuracy'])
19
20 model.fit(x_train, y_train, epochs=5)
21 model.evaluate(x_test, y_test)
22
```


- Git graph도 설치



AI Experts
Who Lead
The Future

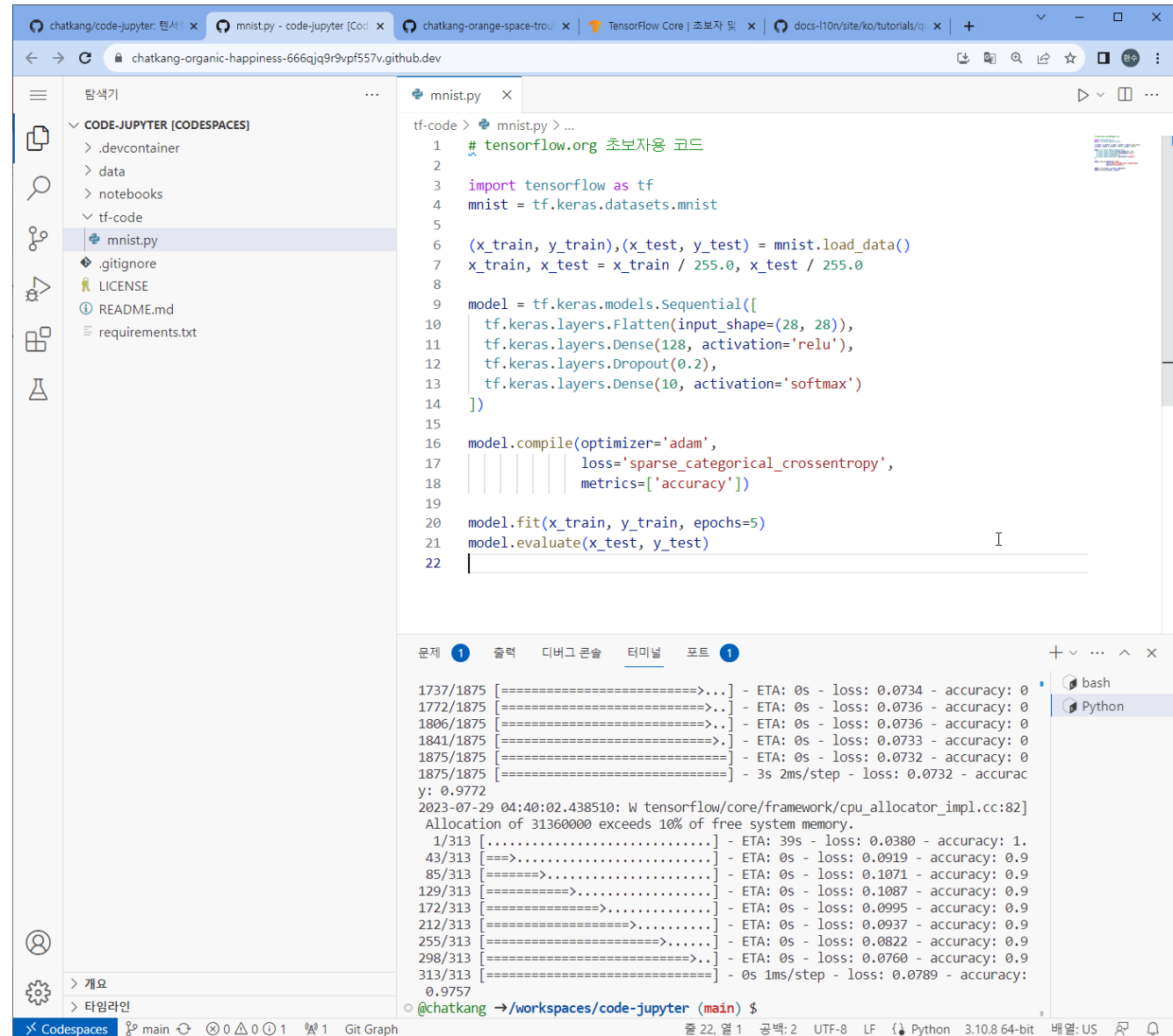
04

파이썬 코드 실행

소스 파일 mnist.py 실행

오픈소스 소프트웨어를 위한 깃과 깃허브 Python language

- 우측 삼각형, ctrl + F5



```
tf-code > mnist.py > ...
1 # tensorflow.org 초보자용 코드
2
3 import tensorflow as tf
4 mnist = tf.keras.datasets.mnist
5
6 (x_train, y_train), (x_test, y_test) = mnist.load_data()
7 x_train, x_test = x_train / 255.0, x_test / 255.0
8
9 model = tf.keras.models.Sequential([
10     tf.keras.layers.Flatten(input_shape=(28, 28)),
11     tf.keras.layers.Dense(128, activation='relu'),
12     tf.keras.layers.Dropout(0.2),
13     tf.keras.layers.Dense(10, activation='softmax')
14 ])
15
16 model.compile(optimizer='adam',
17               loss='sparse_categorical_crossentropy',
18               metrics=['accuracy'])
19
20 model.fit(x_train, y_train, epochs=5)
21 model.evaluate(x_test, y_test)
22
```

문제 1 출력 디버그 콘솔 터미널 포트 1

```
1737/1875 [=====>...] - ETA: 0s - loss: 0.0734 - accuracy: 0
1772/1875 [=====>...] - ETA: 0s - loss: 0.0736 - accuracy: 0
1806/1875 [=====>...] - ETA: 0s - loss: 0.0736 - accuracy: 0
1841/1875 [=====>...] - ETA: 0s - loss: 0.0733 - accuracy: 0
1875/1875 [=====] - ETA: 0s - loss: 0.0732 - accuracy: 0
1875/1875 [=====] - 3s 2ms/step - loss: 0.0732 - accuracy: 0.9772
2023-07-29 04:40:02.438510: W tensorflow/core/framework/cpu_allocator_impl.cc:82]
Allocation of 3136000 exceeds 10% of free system memory.
1/313 [.....] - ETA: 39s - loss: 0.0380 - accuracy: 1.
43/313 [==>.....] - ETA: 0s - loss: 0.0919 - accuracy: 0.9
85/313 [=====>.....] - ETA: 0s - loss: 0.1071 - accuracy: 0.9
129/313 [=====>.....] - ETA: 0s - loss: 0.1087 - accuracy: 0.9
172/313 [=====>.....] - ETA: 0s - loss: 0.0995 - accuracy: 0.9
212/313 [=====>.....] - ETA: 0s - loss: 0.0937 - accuracy: 0.9
255/313 [=====>.....] - ETA: 0s - loss: 0.0822 - accuracy: 0.9
298/313 [=====>.....] - ETA: 0s - loss: 0.0760 - accuracy: 0.9
313/313 [=====] - 0s 1ms/step - loss: 0.0789 - accuracy: 0.9757
@chatkang →/workspaces/code-jupyter (main) $
```

Faker 모듈로 한글 데이터 생성 x VS Code의 유용한 Extension(확) x mydata/requirements.txt at mai x 점프 투 파이썬 - 라이브러리 예 x 120 테스트용 데이터를 생성하라 x +

wikidocs.net/105448

113 반드시 매서드를 구현하도록 하려면? -
114 프로그램 종료 시 특정 작업을 실행하
115 오류 위치와 그 원인을 알려면? — trac
116 데이터의 타입을 확인하려면? — typin
18장 외부 라이브러리 다루기
117 패키지를 설치하고 관리하려면? — pip
118 HTTP 메서드를 테스트하려면? — requ
119 문자열 중 바뀐 부분을 확인하려면? —
120 테스트용 데이터를 생성하려면? — fak
121 파이썬으로 방정식을 풀려면? — symp
122 실행 파일(exe)로 배포하려면? — pyins
A 부록 - 파이썬 라이브러리를 이해하기 위한
01 파이썬과 유니코드
02 클로저와 데코레이터
03 이터레이터와 제너레이터
04 파이썬 타입 어노테이션

점프 투 파이썬 - 라이브러리 예제 편 / 18장 외부 라이브러리 다루기 / 120 테스트용 데이터를 생성하려면? — f...

120 테스트용 데이터를 생성하려면? — faker

faker는 테스트용 가짜 데이터를 생성할 때 사용하는 라이브러리이다. 마찬가지로 pip을 이용하여 설치한다.

```
pip install Faker
```

문제

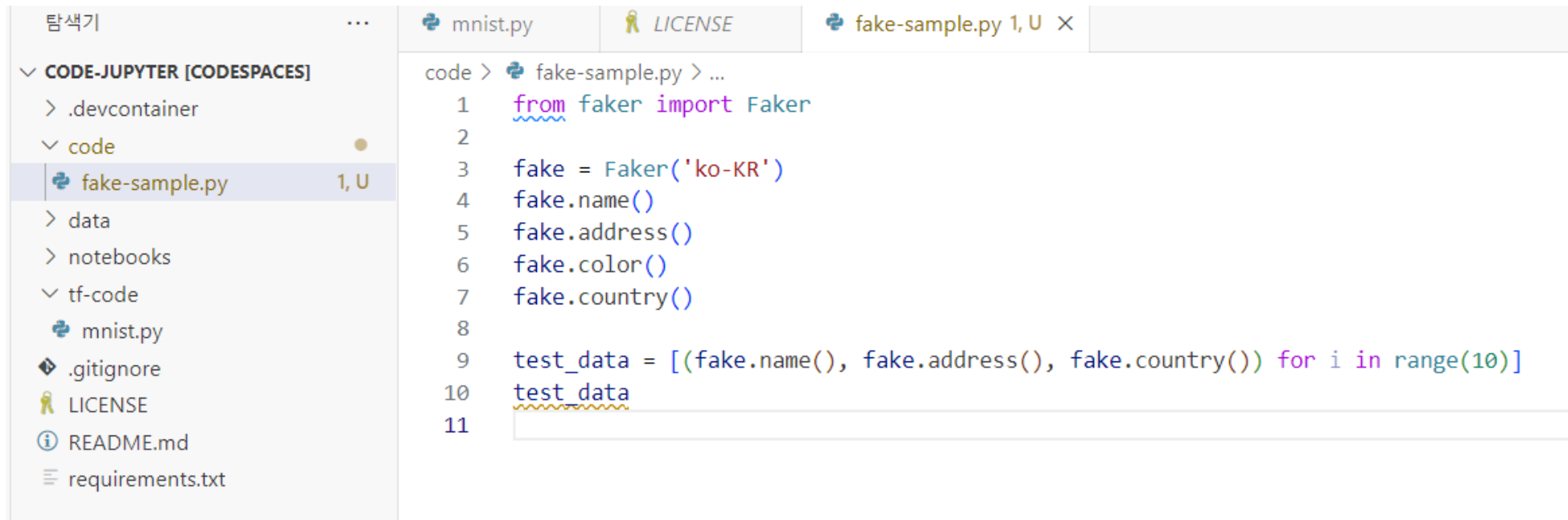
주소록 프로그램을 개발 중이다. 사용할 테스트 데이터가 필요하나 일일이 입력하는 것은 너무 비효율적이고 충분한 양을 준비하기도 쉽지 않다. 보안 문제로 수집한 실제 데이터를 사용할 수도 없다.

이럴 때 다음과 같은 형식의 테스트 데이터 30건이 필요하다고 하자. 직접 데이터를 작성하지 말고 좀 더 편리하게 테스트 데이터를 만들려면 어떻게 해야 할까?

```
[(이름1, 주소1), (이름2, 주소2), ..., (이름30, 주소30)]
```

풀이

테스트 데이터는 faker를 사용하면 아주 쉽게 만들 수 있다. 이름은 다음처럼 만들 수 있다.



탐색기 ...

- CODE-JUPYTER [CODESPACES]
- > .devcontainer
- code
 - fake-sample.py 1, U
- > data
- > notebooks
- tf-code
 - mnist.py
- .gitignore
- LICENSE
- README.md
- requirements.txt

code > fake-sample.py > ...

```
1 from faker import Faker
2
3 fake = Faker('ko-KR')
4 fake.name()
5 fake.address()
6 fake.color()
7 fake.country()
8
9 test_data = [(fake.name(), fake.address(), fake.country()) for i in range(10)]
10 test_data
11
```

```
from faker import Faker
```

```
fake = Faker('ko-KR')
fake.name()
fake.address()
fake.color()
fake.country()
```

```
test_data = [(fake.name(), fake.address(),
fake.country()) for i in range(10)]
test_data
```

줄 실행 화면

오픈소스 소프트웨어를 위한 깃과 깃허브 Python language

- Shift + enter

```
code > fake-sample.py > ...
1 from faker import Faker
2
3 fake = Faker('ko-KR')
4 fake.name()
5 fake.address()
6 fake.color()
7 fake.country()
8
9 test_data = [(fake.name(), fake.address(), fake.country()) for i in range(10)]
10 test_data
11
```

```
'수단'
>>> test_data = [(fake.name(), fake.address(), fake.country()) for i in range(10)]
>>> test_data
[('장수빈', '세종특별자치시 동대문구 개포588길 (정자권리)', '가뽕'), ('박준서', '충청북도 음성군 삼성가', '폴란드'), ('이영환', '충청남도 고양시 덕양구 가락9가 (명숙중면)', '몽골'), ('오민재', '서울특별시 도봉구 서초중앙76거리 (지민심동)', '세인트빈센트 그레나딘'), ('박경희', '서울특별시 송파구 서초대길 (수빈장동)', '가나'), ('최은서', '대전광역시 영동포구 도산대거리 (은서백읍)', '루마니아'), ('김민수', '경상북도 성남시 연주길', '보스니아 헤르체고비나'), ('이현지', '울산광역시 영동포구 서초중앙11로', '도미니카 연방'), ('이민석', '경기도 원주시 서초중앙가', '파푸아 뉴기니'), ('노건우', '부산광역시 영동포구 백제고분0길 (성현김송리)', '부르키나파소')]
>>> dir(fake)
['_annotations_', '_class_', '_deepcopy_', '_delattr_', '_dict_', '_dir_', '_doc_', '_eq_', '_format_', '_ge_', '_getattr_', '_getattribute_', '_getitem_', '_gt_', '_hash_', '_init_', '_init_subclass_', '_le_', '_lt_', '_module_', '_ne_', '_new_', '_reduce_', '_reduce_ex_', '_repr_', '_setattr_', '_setstate_', '_sizeof_', '_str_', '_subclasshook_', '_weakref_', '_factories_', '_factory_map_', '_locales_', '_map_provider_method_', '_select_factory_', '_select_factory_choice_', '_select_factory_dist_ribution_', '_unique_proxy_', '_weights_', '_aba_', '_add_provider_', '_address_', '_address_detail_', '_administrative_unit_', '_am_pm_', '_android_platform_token_', '_ascii_company_email_', '_ascii_email_', '_ascii_free_email_', '_ascii_safe_email_', '_bank_country_', '_bban_', '_binary_', '_boolean_', '_borou_gh_', '_bothify_', '_bs_', '_building_dong_', '_building_name_', '_building_number_', '_building_suffix_', '_cache_pattern_', '_catch_phrase_', '_century_', '_chrome_', '_city_', '_city_suffix_', '_color_', '_color_name_', '_company_', '_company_email_', '_company_suffix_', '_coordinate_', '_country_', '_country_calli_ ng_code_', '_country_code_', '_credit_card_expire_', '_credit_card_full_', '_credit_card_number_', '_cr edit_card_provider_', '_credit_card_security_code_', '_cryptocurrency_', '_cryptocurrency_code_', '_c ryptocurrency_name_', '_csv_', '_currency_', '_currency_code_', '_currency_name_', '_currency_symbol_', '_current_country_', '_current_country_code_', '_date_', '_date_between_', '_date_between_dates_', '_dat e_object_', '_date_of_birth_', '_date_this_century_', '_date_this_decade_', '_date_this_month_', '_date _this_year_', '_date_time_', '_date_time_ad_', '_date_time_between_', '_date_time_between_dates_', '_da te_time_this_century_', '_date_time_this_decade_', '_date_time_this_month_', '_date_time_this_year_']
```

AI Experts
Who Lead
The Future

04

주피터 노트북 실행

- 파일
 - notebooks/mnist-digit.ipynb
- 화면 분할해 사용

The screenshot shows a JupyterLab interface with the following components:

- Left Sidebar (File Explorer):** Shows the file structure. The 'notebooks' directory is expanded, showing 'mnist-digit.ipynb' (highlighted with a red dashed box).
- Top Toolbar:** Shows the active notebook 'mnist-digit.ipynb' and a new file 'mnist.py' (also highlighted with a red dashed box).
- Main Area (Notebook):** Displays the content of 'mnist-digit.ipynb'. It includes a code cell with the following Python code:

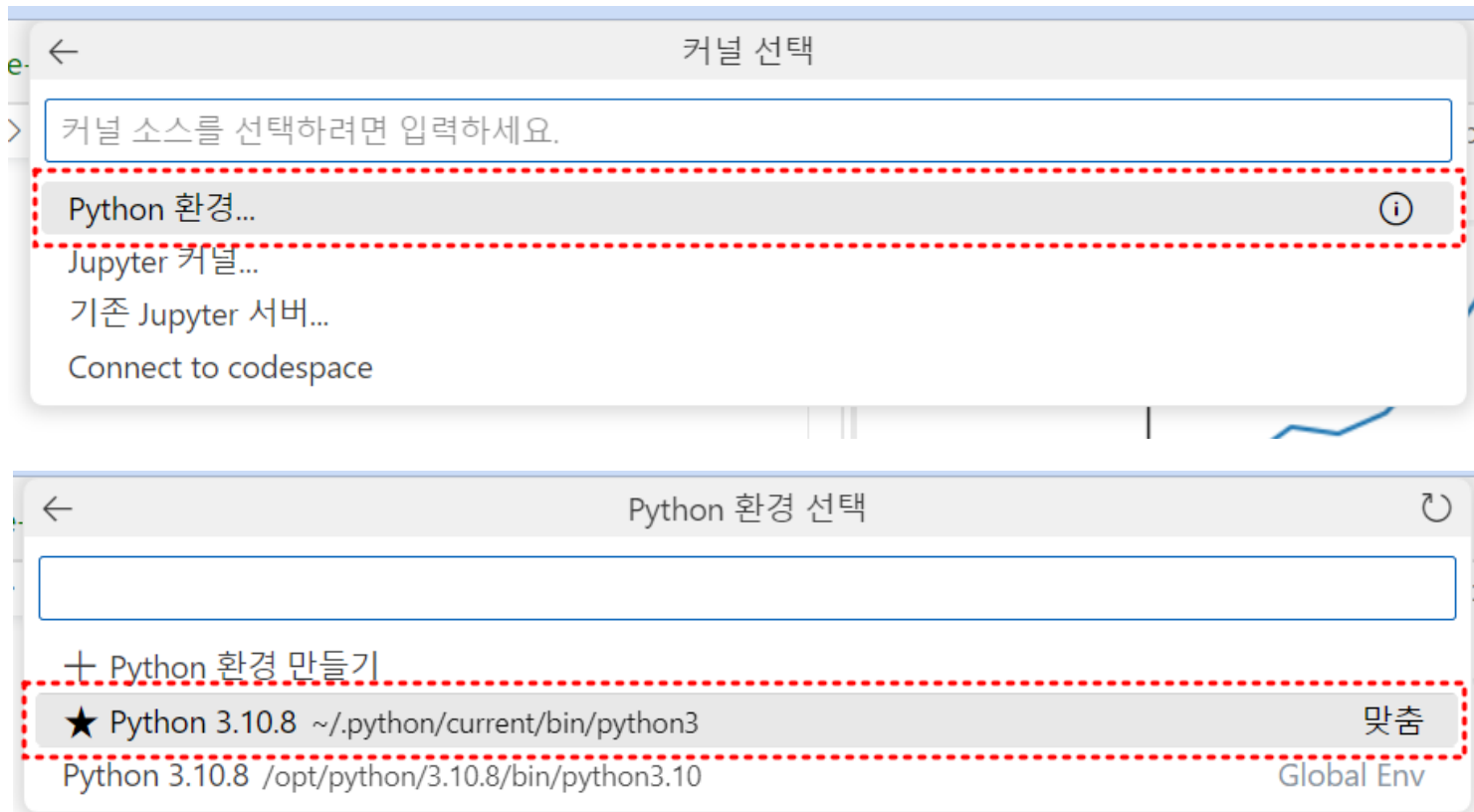
```
import tensorflow as tf
```

The output of this cell shows TensorFlow logs and the version '2.10.0'.

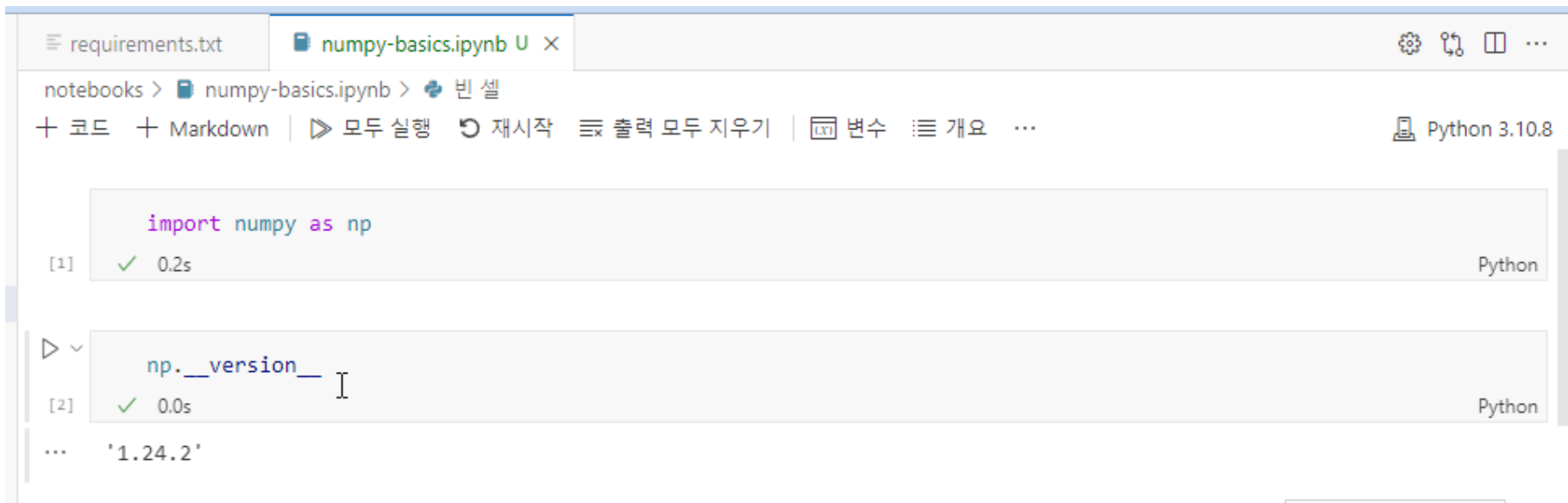
```
tf.__version__
```

The output of this cell is '2.10.0'.

- 셀에서
 - Shift + enter



- 셀 복사 붙이기
 - C
 - V
- 셀 나누기
 - Ctrl + shift + -
- 셀 실행
 - Ctrl + shift



requirements.txt numpy-basics.ipynb U x

notebooks > numpy-basics.ipynb > 빈 셀

+ 코드 + Markdown | ▶ 모두 실행 ↺ 재시작 ≡ 출력 모두 지우기 | [x] 변수 ≡ 개요 ... Python 3.10.8

```
[1] ✓ 0.2s import numpy as np
```

```
[2] ✓ 0.0s np.__version__
```

```
... '1.24.2'
```

• 행 2개

The screenshot shows a JupyterLab environment with the following components:

- File Browser (Left):** Displays the file structure. The 'notebooks' folder is expanded, showing 'mnist-digit.ipynb' (highlighted with a red dashed box).
- Code Editor (Center):** Displays the code for 'mnist-digit.ipynb'. The code includes:

```
import tensorflow as tf
```

Execution output for [1]:

```
2023-07-29 05:02:32.643564: I tensorflow/core/platform/cpu_fi
2023-07-29 05:02:32.779321: W tensorflow/stream_executor/pla
2023-07-29 05:02:32.779348: I tensorflow/stream_executor/cud
2023-07-29 05:02:32.811626: E tensorflow/stream_executor/cud
2023-07-29 05:02:33.684790: W tensorflow/stream_executor/pla
2023-07-29 05:02:33.684875: W tensorflow/stream_executor/pla
2023-07-29 05:02:33.684886: W tensorflow/compiler/tf2tensorr
```

Execution output for [2]:

```
tf.__version__
'2.10.0'
```
- Code Editor (Right):** Displays the code for 'mnist.py' (highlighted with a red dashed box). The code includes:

```
# tensorflow.org 초보자용 코드
import tensorflow as tf
mnist = tf.keras.datasets.mnist

(x_train, y_train), (x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0

model = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(10, activation='softmax')
])

model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

model.fit(x_train, y_train, epochs=5)
model.evaluate(x_test, y_test)
```

코딩이나 복사 & 붙이기

The screenshot displays a Jupyter Notebook environment with the following components:

- File Explorer (Left):** Shows a project structure with folders like `CODE-JUPYTER [CODESPACES]`, `data`, `notebooks`, and `tf-code`. Files include `fake-sample.py`, `image-classifier.ipynb`, `matplotlib.ipynb`, `mnist-digit.ipynb` (selected), `population.ipynb`, `mnist.py`, `.gitignore`, `LICENSE`, `README.md`, and `requirements.txt`.
- Code Editor (Center):** Contains the following Python code:


```
import tensorflow as tf
mnist = tf.keras.datasets.mnist

(x_train, y_train), (x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0

model = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(input_shape=(28, 28)),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(10, activation='softmax')
])

model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

model.fit(x_train, y_train, epochs=5)

model.evaluate(x_test, y_test)
```
- Output Console (Bottom):** Shows execution results and logs:
 - Execution [3]: `mnist.load_data()` completed in 0.7s.
 - Execution [4]: Model compilation completed in 0.3s.
 - Execution [5]: Model fitting (`model.fit`) completed in 18.1s. The console output shows training progress for 5 epochs:


```
Epoch 1/5
1875/1875 [=====] - 5s 2ms/step - loss: 0.3013 - accuracy: 0.9127
Epoch 2/5
1875/1875 [=====] - 3s 2ms/step - loss: 0.1482 - accuracy: 0.9560
Epoch 3/5
1875/1875 [=====] - 3s 2ms/step - loss: 0.1115 - accuracy: 0.9664
Epoch 4/5
1875/1875 [=====] - 3s 2ms/step - loss: 0.0900 - accuracy: 0.9729
Epoch 5/5
1875/1875 [=====] - 3s 2ms/step - loss: 0.0771 - accuracy: 0.9759

<keras.callbacks.History at 0x7fe37059ba00>
```
 - Execution [6]: Model evaluation (`model.evaluate`) completed in 0.6s. The console output shows:


```
1/313 [.....] - ETA: 46s - loss: 0.0063 - accuracy: 1.0000
2023-07-29 05:07:58.470240: W tensorflow/core/framework/cpu_allocator_impl.cc:82] Allocation of 31360000 exceeds 10% of free system memory.
313/313 [=====] - 1s 1ms/step - loss: 0.0703 - accuracy: 0.9776

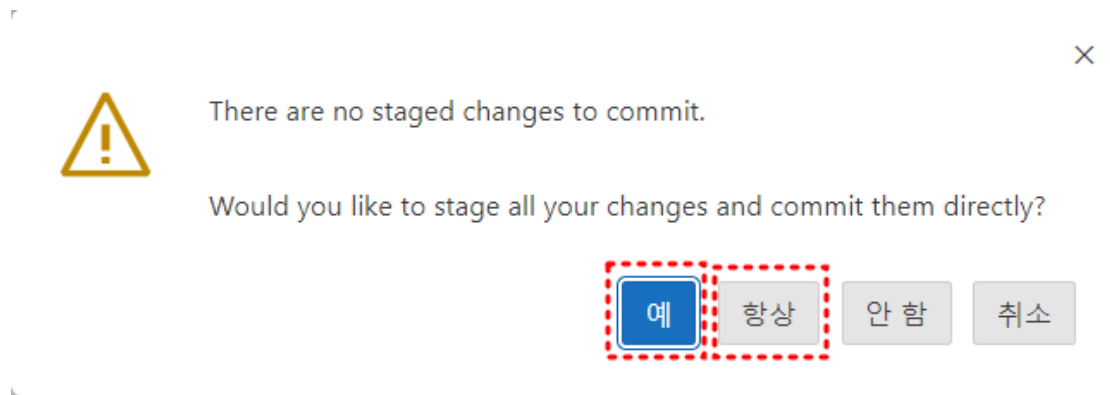
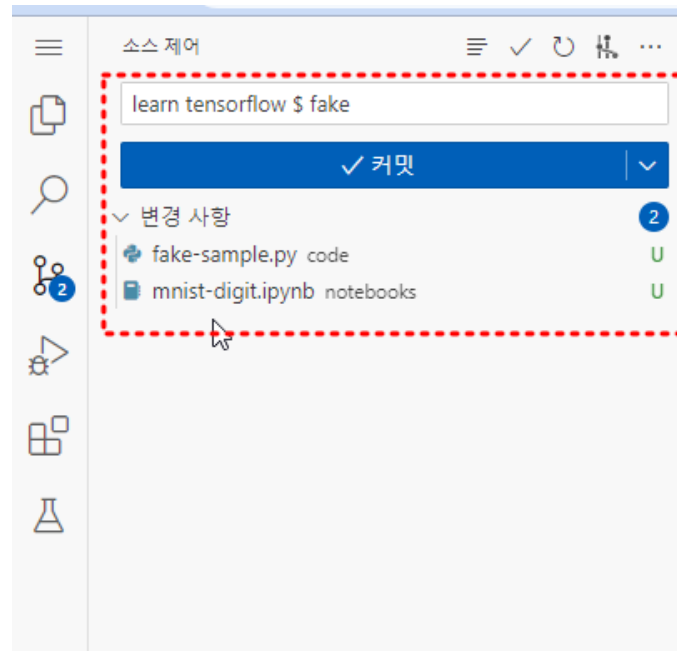
[0.07028835266828537, 0.9775999784469604]
```

AI Experts
Who Lead
The Future

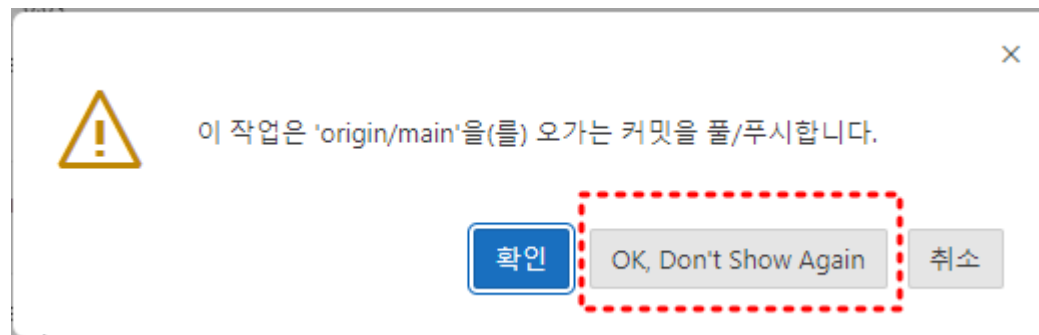
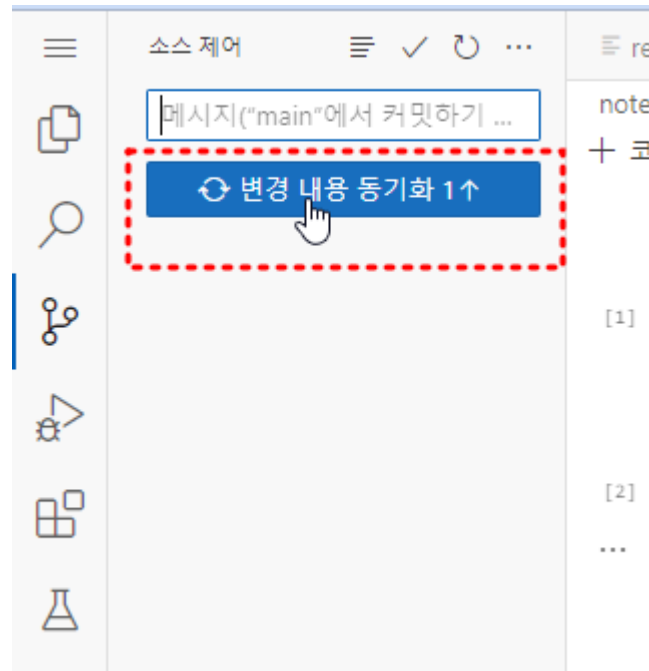
05

코드스페이스 저장소를 깃허브에 push

- 메시지 작성



- push



자신의 깃허브 저장소 확인

오픈소스 소프트웨어를 위한 깃과 깃허브 Python language

The screenshot shows a GitHub repository page for 'chatkang / code-jupyter'. The file 'mnist-digit.ipynb' is selected, showing its code and execution output. The code is a Jupyter notebook cell with the following content:

```
In [1]: import tensorflow as tf
```

The output shows several TensorFlow warnings and error messages, including:

```
2023-07-29 05:02:32.643564: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical operations: AVX2 AVX512F FMA
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
2023-07-29 05:02:32.779321: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'libcudart.so.11.0'; dlderror: libcudart.so.11.0: cannot open shared object file: No such file or directory
2023-07-29 05:02:32.779348: I tensorflow/stream_executor/cuda/cudart_stub.cc:29] Ignore above cudart dlerror if you do not have a GPU set up on your machine.
2023-07-29 05:02:32.811626: E tensorflow/stream_executor/cuda/cuda_blas.cc:2981] Unable to register cuBLAS factory: Attempting to register factory for plugin cuBLAS when one has already been registered
2023-07-29 05:02:33.684790: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'libnvinfer.so.7'; dlderror: libnvinfer.so.7: cannot open shared object file: No such file or directory
2023-07-29 05:02:33.684875: W tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load dynamic library 'libnvinfer_plugin.so.7'; dlderror: libnvinfer_plugin.so.7: cannot open shared object file: No such file or directory
2023-07-29 05:02:33.684886: W tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Cannot dlopen some TensorRT libraries. If you would like to use Nvidia GPU with TensorRT, please make sure the missing libraries mentioned above are installed properly.
```

The code continues with:

```
In [2]: tf.__version__
```

The output is:

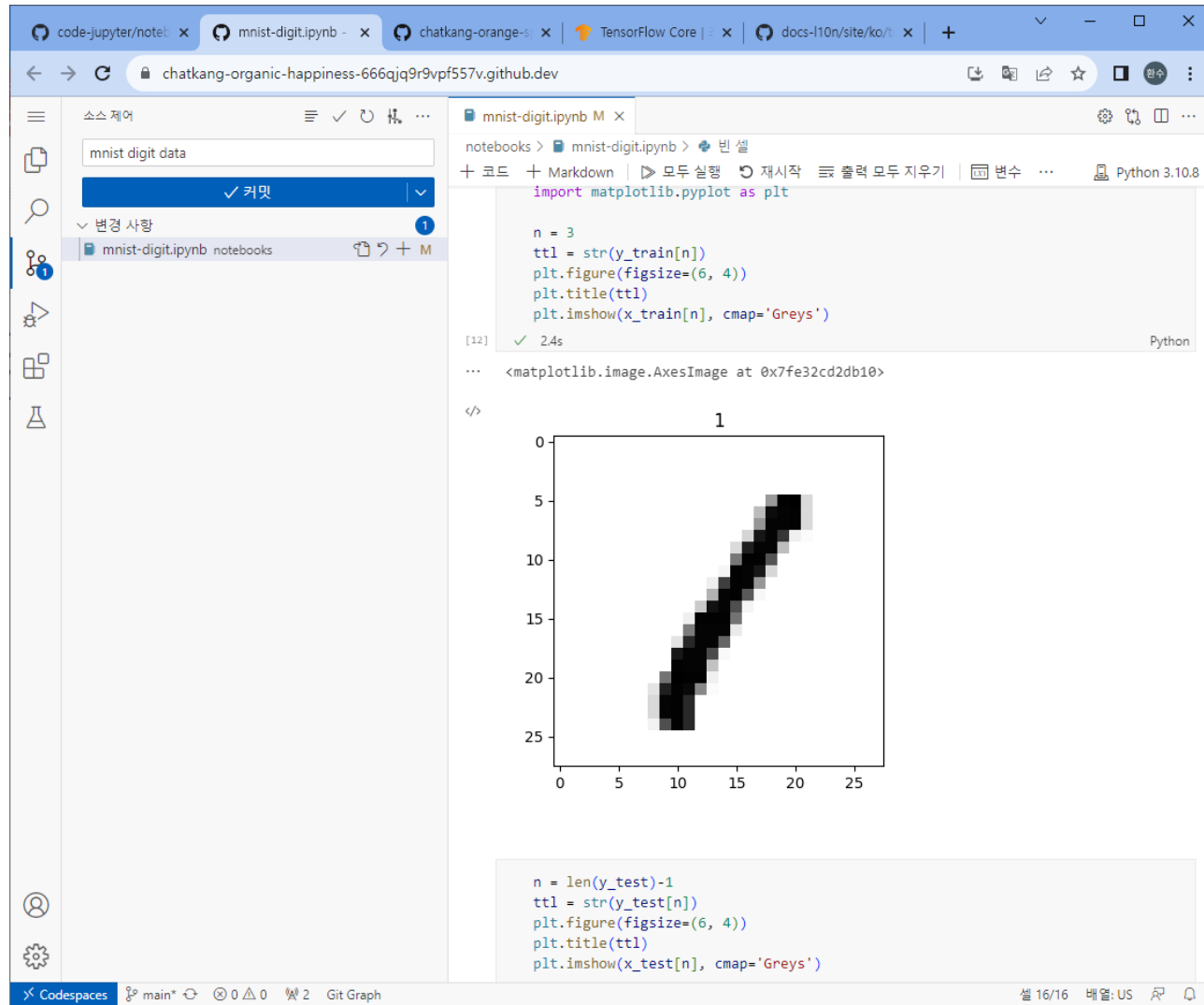
```
Out[2]: '2.10.0'
```

The code continues with:

```
In [3]: import tensorflow as tf
mnist = tf.keras.datasets.mnist

(x_train, y_train), (x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0
```


• 커밋



The screenshot shows a web browser interface for a Jupyter Notebook titled 'mnist-digit.ipynb' in a Code Spaces environment. The left sidebar displays the file explorer with 'mnist digit data' and a '커밋' (Commit) button. The main area shows the notebook content, which includes a Python code cell and its output. The code cell contains the following Python code:

```
import matplotlib.pyplot as plt

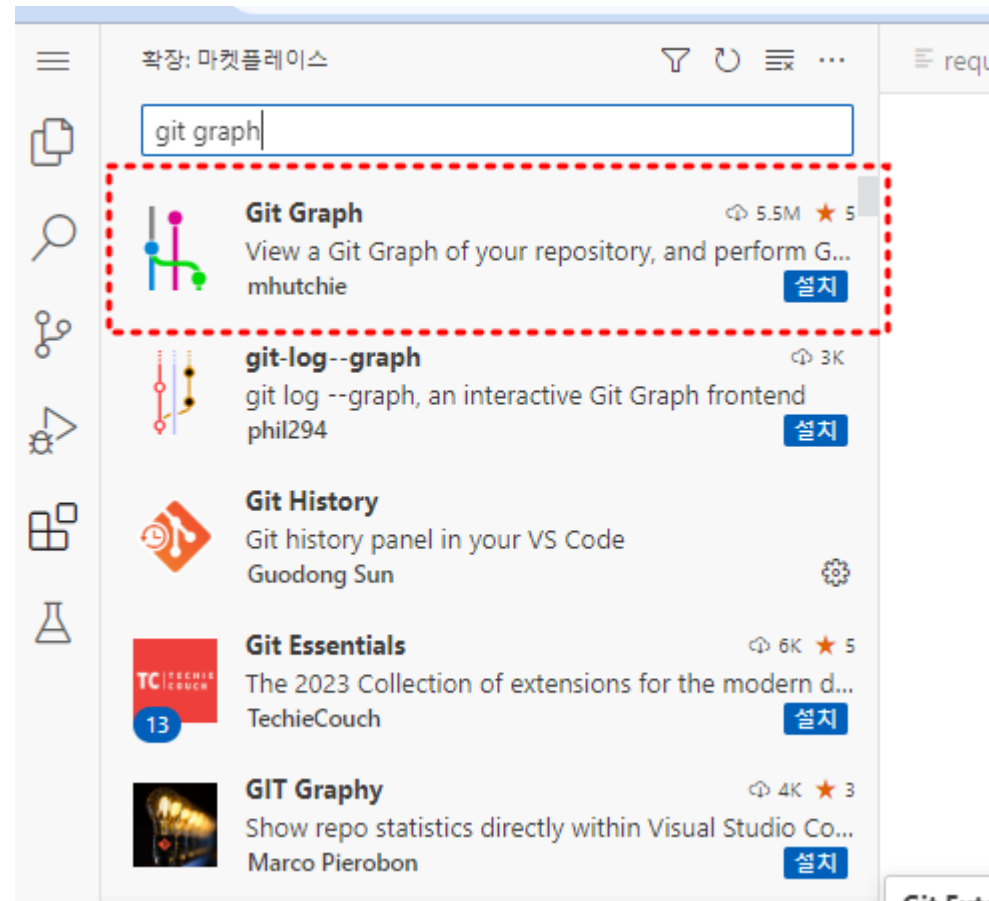
n = 3
ttl = str(y_train[n])
plt.figure(figsize=(6, 4))
plt.title(ttl)
plt.imshow(x_train[n], cmap='Greys')
```

The output of the code cell shows the execution result: `<matplotlib.image.AxesImage at 0x7fe32cd2db10>`. Below the code, a plot is displayed showing a handwritten digit '1' on a grayscale background. The plot has a title '1' and axes ranging from 0 to 25. Below the plot, another code cell is visible, containing the following Python code:

```
n = len(y_test)-1
ttl = str(y_test[n])
plt.figure(figsize=(6, 4))
plt.title(ttl)
plt.imshow(x_test[n], cmap='Greys')
```

The bottom status bar indicates the current branch is 'main' and shows the Git Graph icon.

- Git graph



버전 확인

오픈소스 소프트웨어를 위한 깃과 깃허브 Python language

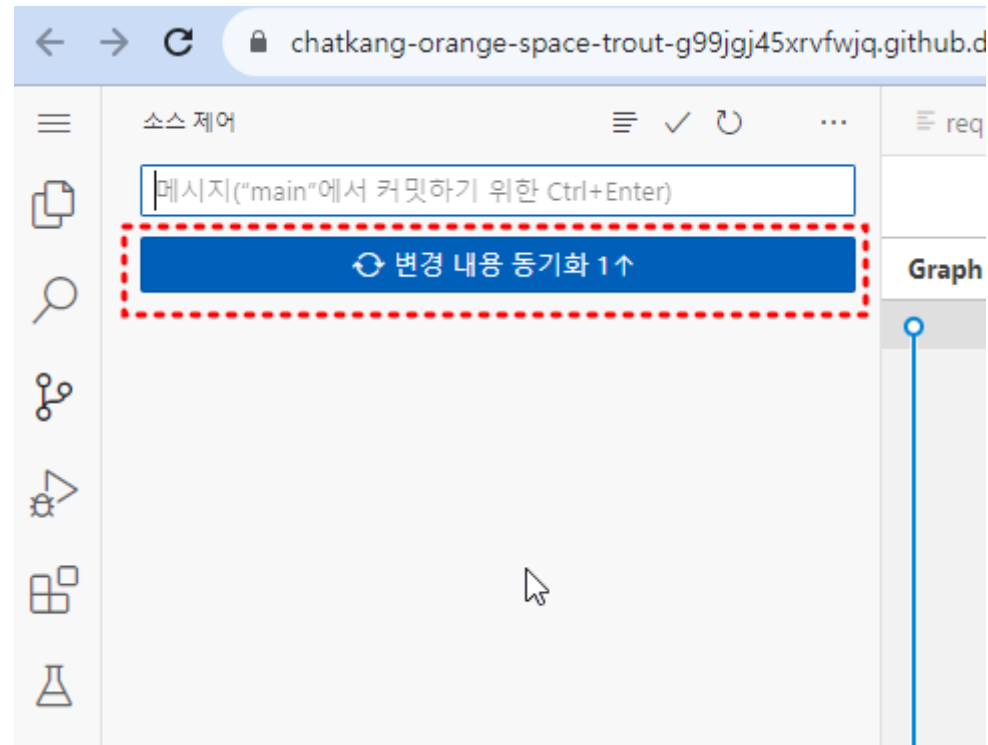
- 아래로

The screenshot shows a web browser with multiple tabs open, including 'code-jupyter/notebooks', 'Git Graph - code-jupyter', 'chatkang-orange-space', 'TensorFlow Core | 초보', and 'docs-l10n/site/ko/tutorial'. The main browser window displays the GitHub repository page for 'chatkang-organic-happiness-6666jq9r9vpf557v.github.dev'. The repository is named 'mnist-digit.ipynb'. The page shows the commit history and the content of the notebook. The commit history table is as follows:

Graph	Description	Date	Author	Commit
main	mnist digit data	29 Jul 2023 14:25	chatkang	ff5d707b
origin/HEAD	learn tensorflow \$ fake	29 Jul 2023 14:10	chatkang	93b6a066
origin/main	Create mnist.py	29 Jul 2023 13:21	chatkang	76e8dddc
	Update requirements.txt	29 Jul 2023 13:09	chatkang	c60092fa

The notebook content shows a Jupyter Notebook cell with a JSON output. The output is a dictionary with the following structure:

```
{  "data": {    "text/plain": [      "[0.07028835266828537, 0.9775999784469604]"    ]  },  "execution_count": 7,  "metadata": {},  "output_type": "execute_result",  "source": [    "model.evaluate(x_test, y_test)"  ]}
```



- 깃허브에서 파일 변화 확인