End-to-end Python Practice on Industrial Data Engineering

葉信和 / Hsin-Ho Yeh
Software Engineer / Funder / CEO @ 信誠金融科技
hsinho.yeh@footprint-ai.com

2022/06/01
2022/06/08

Download Slides

https://reurl.cc/p1pg8Q

https://github.com/FootprintAI/talks/tree/main/slides

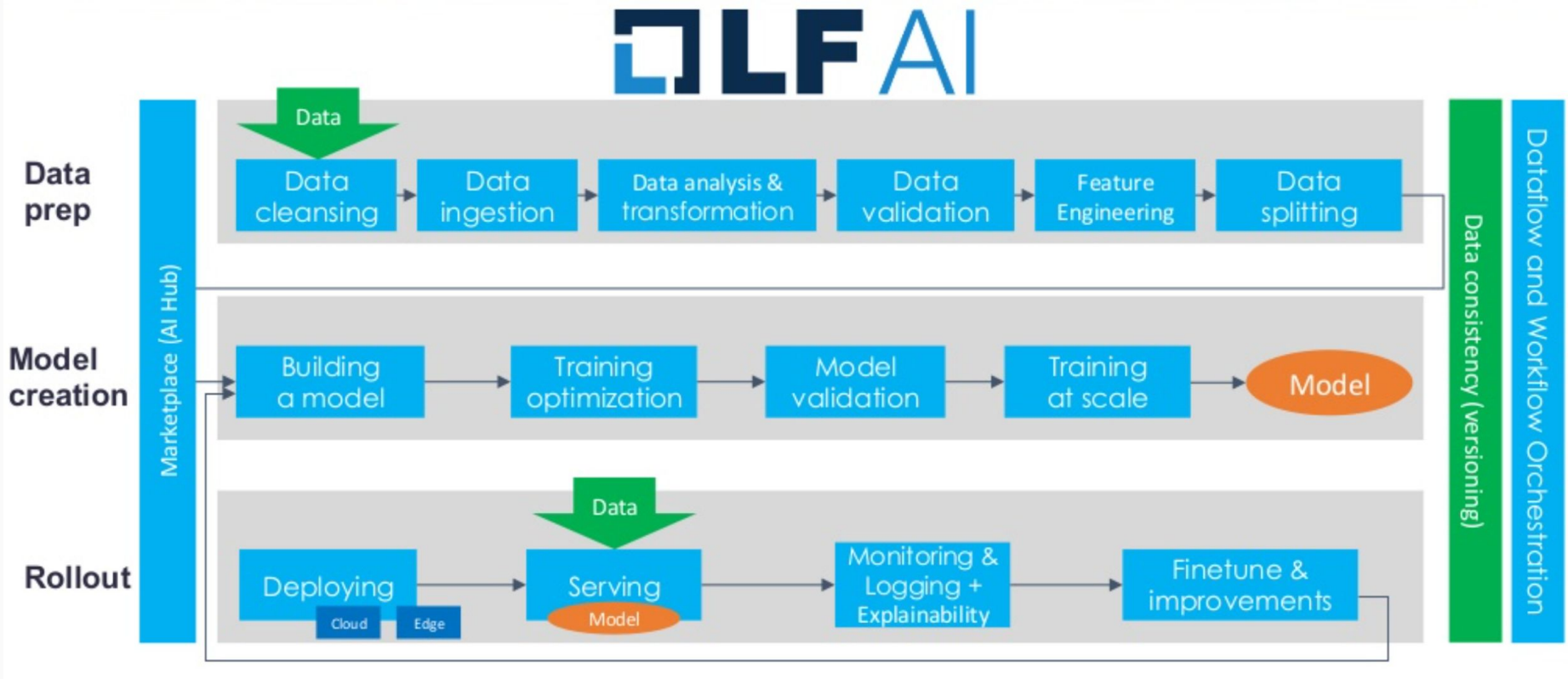
About me

- 2020 - Present at 信誠金融科技
  - Shrimping: A data-sharing platform
    - https://get-shrimping.footprint-ai.com
  - Tintin: a machine learning platform for everyone
    - https://get-tintin.footprint-ai.com
- 2016 - 2020 at IglooInsure (16M+ in series A+ 2020)
  - Provide digital insurance for e-conomic world
  - Funded in KUL, Headquartered in Singapore
  - First employee/ Engineering Lead / Regional Head/ Chief Engineer
- 2013 - 2016 at Studio Engineering @ hTC
  - Principal Engineer on Cloud Infrastructure Team
- 2009 - 2012 at IIS @ Academia Sinica
  - Computer vision, pattern recognition, and data mining
- CS@CCU, CS@NCKU alumni

Agenda

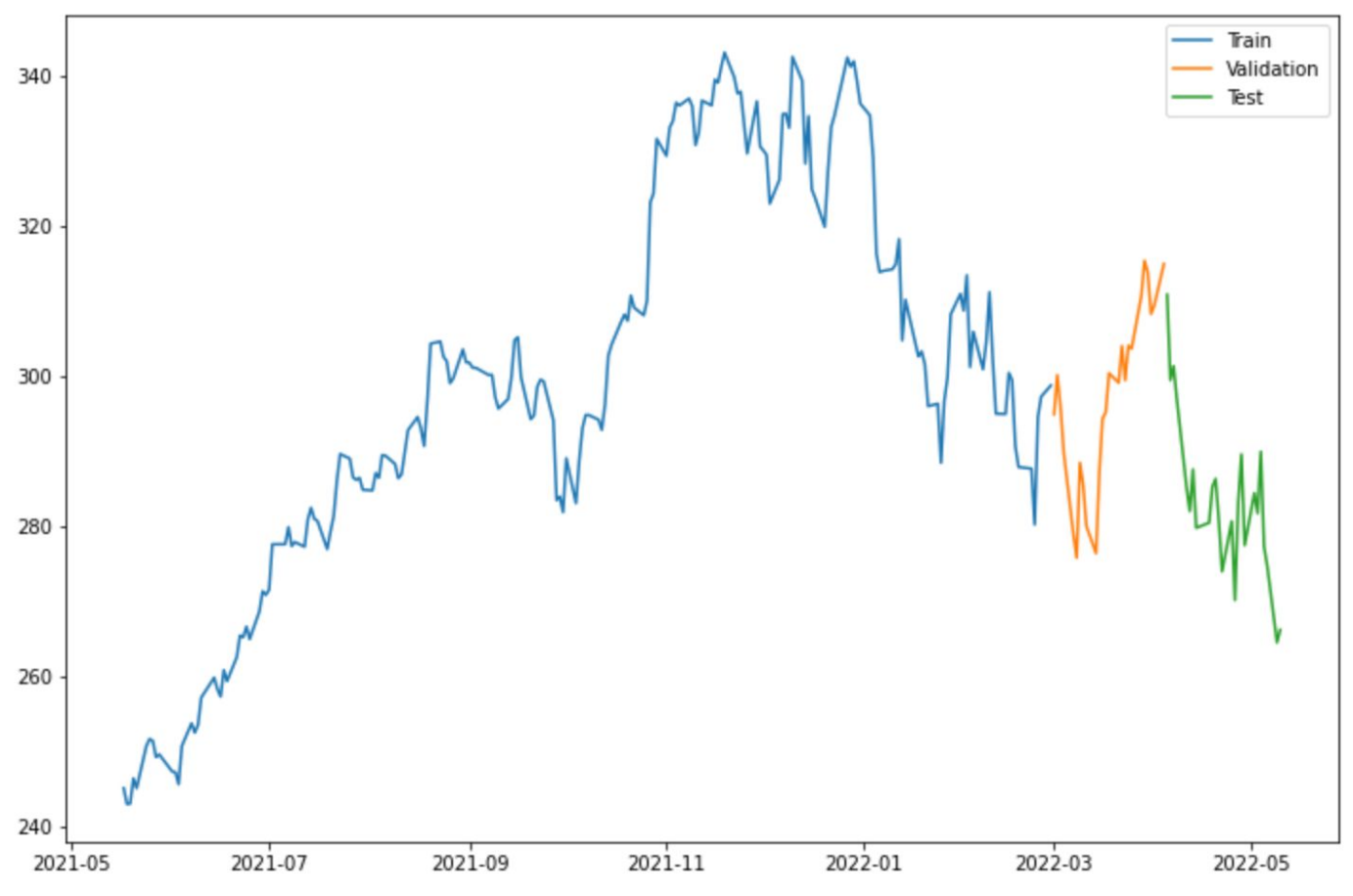
- End-to-end flow on industrial data analysis practice
- How and where do we collect data?
- Web scraping with selenium
- Data Analysis with Pandas
- Build a financial model with Kubeflow
- Q&A

Real-world Machine Learning Application - End-to-End ML LifeCycle



Source: https://www.slideshare.net/AnimeshSingh/advanced-model-inferencing-leveraging-kubeflow-serving-knative-and-istio-196096385

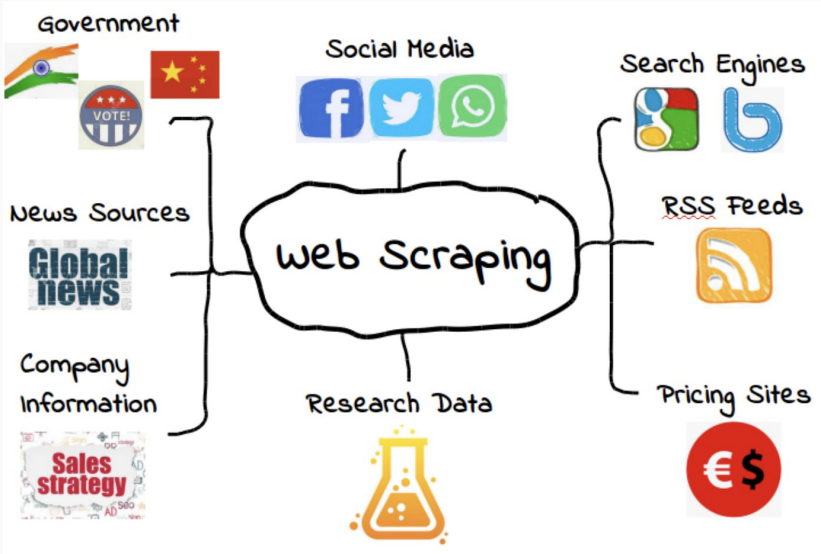
How to predict stock price for the next day?



What is Web Scraping?

- Web Scraping is an **automation** process of collecting **structure data** from public websites.

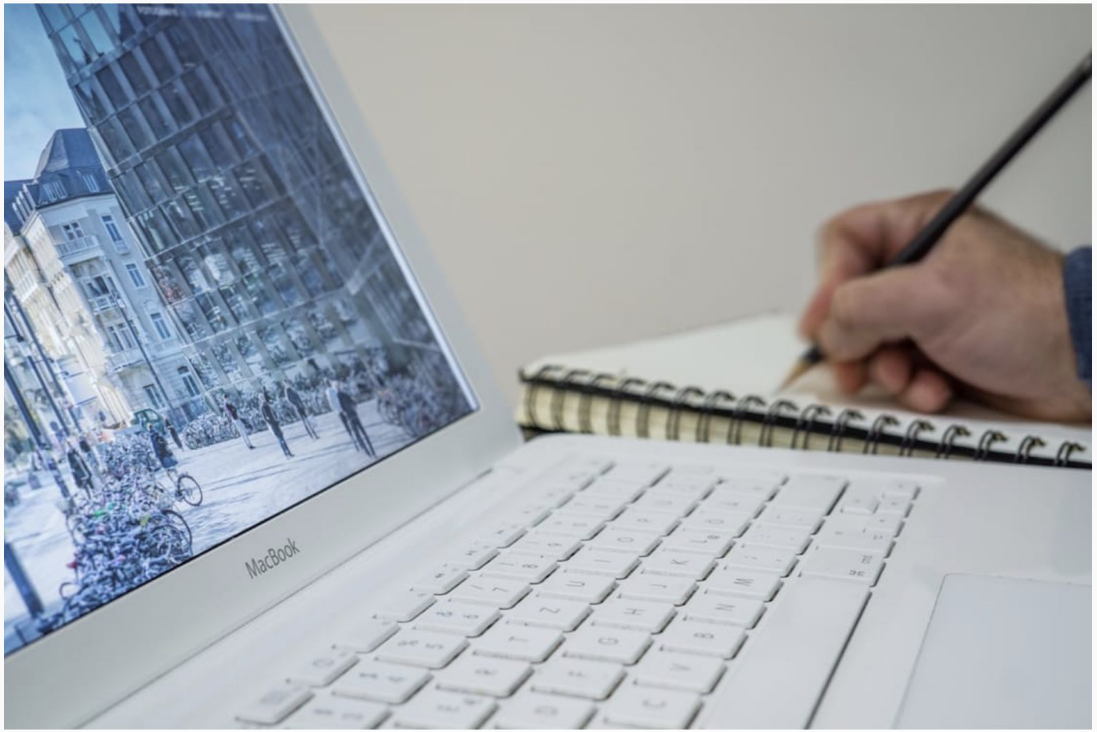
- Common Use Cases including
  - Competitor price monitoring on E-commerce
  - News monitoring from social network



Ref:
https://thestartupcto.org/web-scraping-techniques-5030fbf1fba

Different ways of collecting data from websites (1/3)

- Manually (slow & slow & slow)



Ref: https://scrapingexpert.com/web-scraping-vs-work-manually/

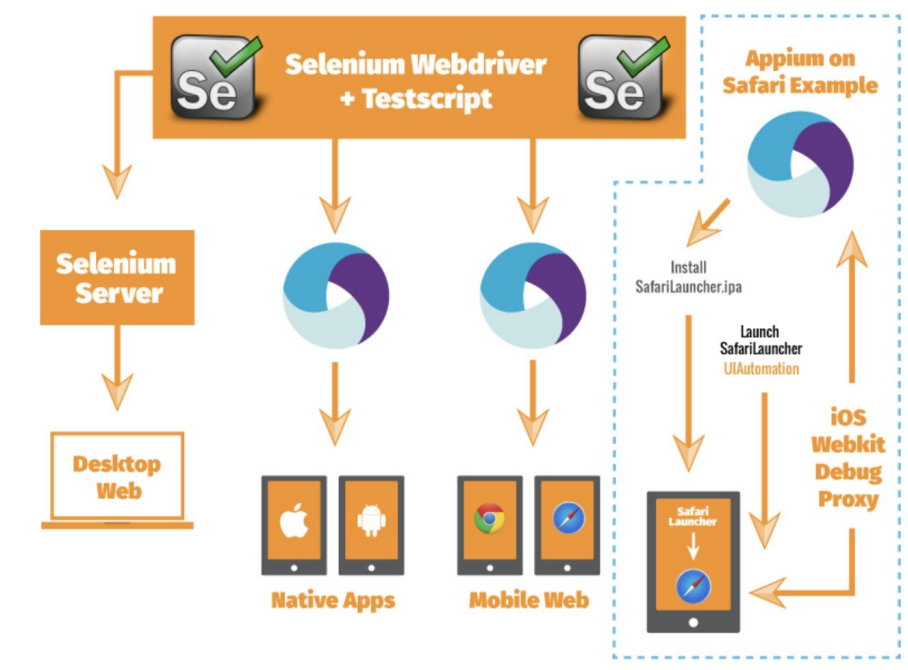
Different ways of collecting data from websites (2/3)

- API Requests with Python(Stable & Fast but not always works…

```
>>> import requests

>>> r = requests.get('https://www.google.com.tw/')

>>> print(r.status_code)
200

>>>print(r.text)
<!doctype html><html itemscope="" itemtype="http://schema.org/WebPage"
lang="zh-TW"><head><meta content="text/html; charset=UTF-8"
http-equiv="Content-Type"><meta
content="/images/branding/googleg/1x/googleg_standa….
```

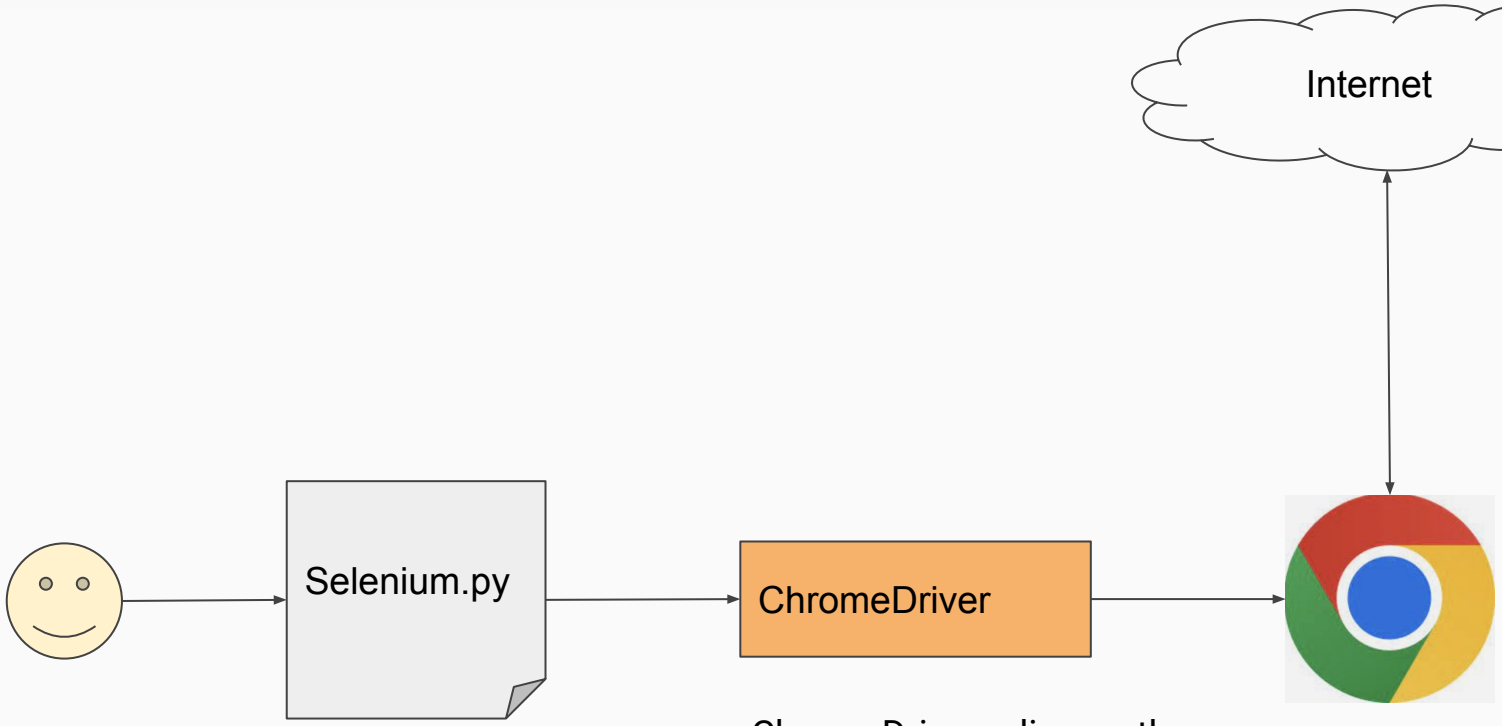
Different ways of collecting data from websites (3/3)

- UI Automation with Selenium
  - Selenium is an web automation tool which allows you to automate UI testing for different browsers via its webdriver.
  - Webdriver is used to control each Browser behaviors making it more like real-human interaction.
- Because of its nature of automation and mimic human behavior, making it a good human-like web-scraping



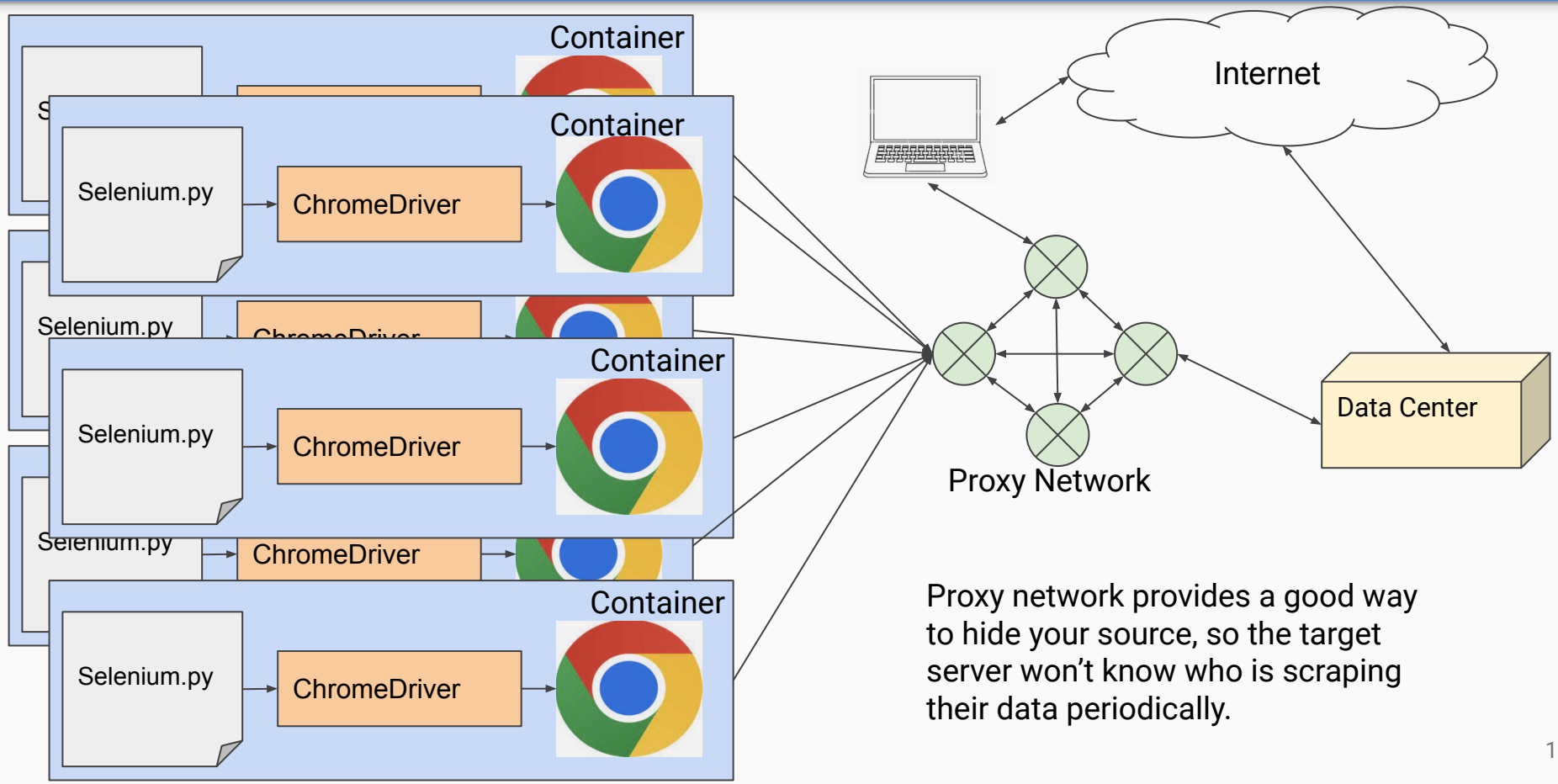
Ref: https://smartbear.com/blog/selenium-cross-browser-testing-on-mobile-devices/

How Selenium WebDriver Works?



ChromeDriver relies on the pre-installed chrome package to send the requests and render the HTML tags

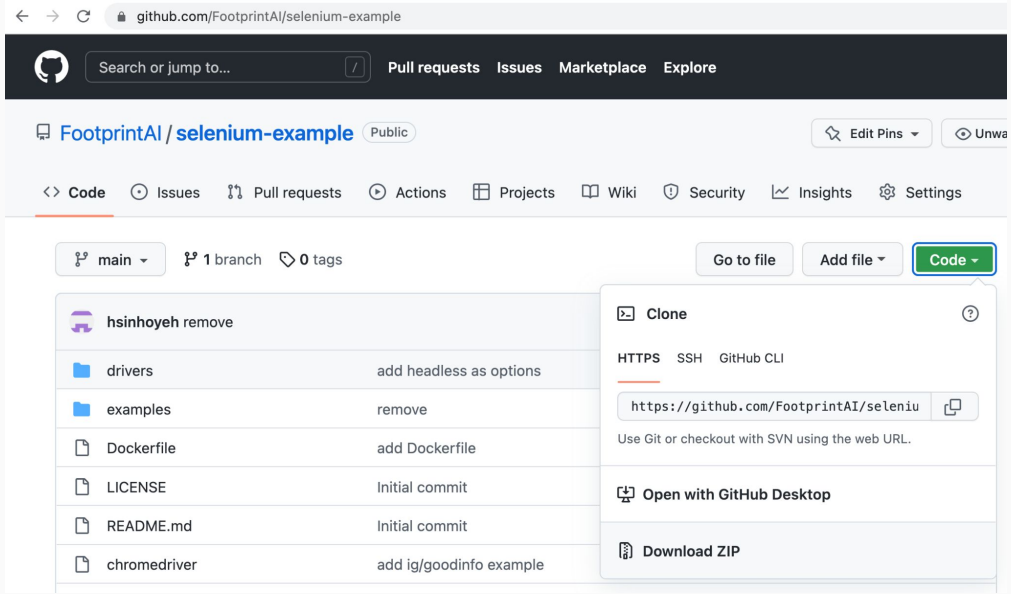
How Selenium WebDriver Works In Scale?



Proxy network provides a good way to hide your source, so the target server won't know who is scraping their data periodically.

Selenium Examples

https://github.com/FootprintAI/selenium-example



Selenium: Basic Example

```
1  from drivers.webdriver import WebDriver
2
3  from selenium.webdriver.common.by import By
4  from selenium.webdriver.common.keys import Keys
5
6  def do(driver: WebDriver, q: str):
7      driver.get("http://www.python.org")
8      assert "Python" in driver.title
9      elem = driver.find_element(By.NAME, "q")
10     elem.clear()
11     elem.send_keys(q)
12     elem.send_keys(Keys.RETURN)
13     assert "No results found." not in driver.page_source
```

By is for element selection
Key is for mapping keyboard

Ref: https://github.com/FootprintAI/selenium-example/blob/main/examples/pyconf.py

Selenium: Webdriver Inheritance & encapsulation

```python
import os
import platform
from selenium.webdriver import Chrome, ChromeOptions

class WebDriver(Chrome):
    def __init__(self, headless:bool=False):
        exec_path = None
        if platform.system() == "Darwin":
            exec_path = self.__wrap_abspath('chromedriver.darwin')
        elif platform.system() == "Windows":
            exec_path = self.__wrap_abspath('chromedriver.exe')
        elif platform.system() == "Linux":
            exec_path = self.__wrap_abspath('chromedriver.linux')

        chrome_options = ChromeOptions()
        # comment this to enable guest browser to show up (for debugging
        if headless:
            chrome_options.add_argument('--headless')
        chrome_options.add_argument('--no-sandbox')

        Chrome.__init__(self,chrome_options=chrome_options, executable_path=exec_path)

    def __wrap_abspath(self, binary)-> str:
        return os.path.join(os.path.dirname(os.path.abspath(__file__)), binary)
```

inheritance

encapsulation

Ref: https://github.com/FootprintAI/selenium-example/blob/main/drivers/webdriver.py

Selenium: Cookies & Navigation

```python
from drivers.webdriver import WebDriver

from time import sleep

def navigation(driver: WebDriver):
    driver.get("https://www.google.com")
    sleep(1)
    driver.get("https://www.python.org")
    sleep(1)
    driver.back()
    sleep(1)
    driver.forward()

def get_cookies(driver: WebDriver):
    driver.get("https://www.google.com")
    print(driver.get_cookies())
```

Ref: https://github.com/FootprintAI/selenium-example/blob/main/examples/browser.py

Selenium: Exceptions

```python
from drivers.webdriver import WebDriver

def exception_handling(driver: WebDriver):
    from selenium.common.exceptions import TimeoutException, NoSuchElementException
    try:
        driver.get("https://www.google.com")
    except TimeoutException:
        print("time out")
    try:
        driver.find_element_by_id("you-should-not-see-me")
    except NoSuchElementException:
        print("no element found")
```

Ref: https://github.com/FootprintAI/selenium-example/blob/main/examples/exceptions.py

Selenium: query methods

Tag

Attribute

```
<input type="text" name="passwd" id="passwd-id" />
```

```
// we have four different ways to locate an individual UI element


element = driver.find_element(By.ID, "passwd-id")
element = driver.find_element(By.NAME, "passwd")
element = driver.find_element(By.XPATH, "//input[@id='passwd-id']")
element = driver.find_element(By.CSS_SELECTOR, "input#passwd-id")
```

Ref: https://selenium-python.readthedocs.io/navigating.html

DOM(Document Object Model) Tree

An example of the DOM

Let's start with the following simple document:

```
<!DOCTYPE HTML>
<html>
<head>
  <title>About elk</title>
</head>
<body>
  The truth about elk.
</body>
</html>
```

The DOM represents HTML as a tree structure of tags. Here's how it looks:

Ref: https://javascript.info/dom-nodes

Example: Instagram Login

```
    driver.get("https://www.instagram.com")
    sleep(5) # wait for the page loading is finished
    # find username fields
    username_field=driver.find_element_by_css_selector("input[name='username']")
    # find password fields
    password_field=driver.find_element_by_css_selector("input[name='password']")
    username_field.clear()
    password_field.clear()
    # send email in username field
    username_field.send_keys(email)
    # send password in password field
    password_field.send_keys(password)
    sleep(5)

    # click login button
    driver.find_element_by_css_selector("button[type='submit']").click()
    __post_login(driver)

def __post_login(driver: WebDriver):

```

Ref: https://github.com/FootprintAI/selenium-example/blob/main/examples/instagram.py

Example: Find Stock Price

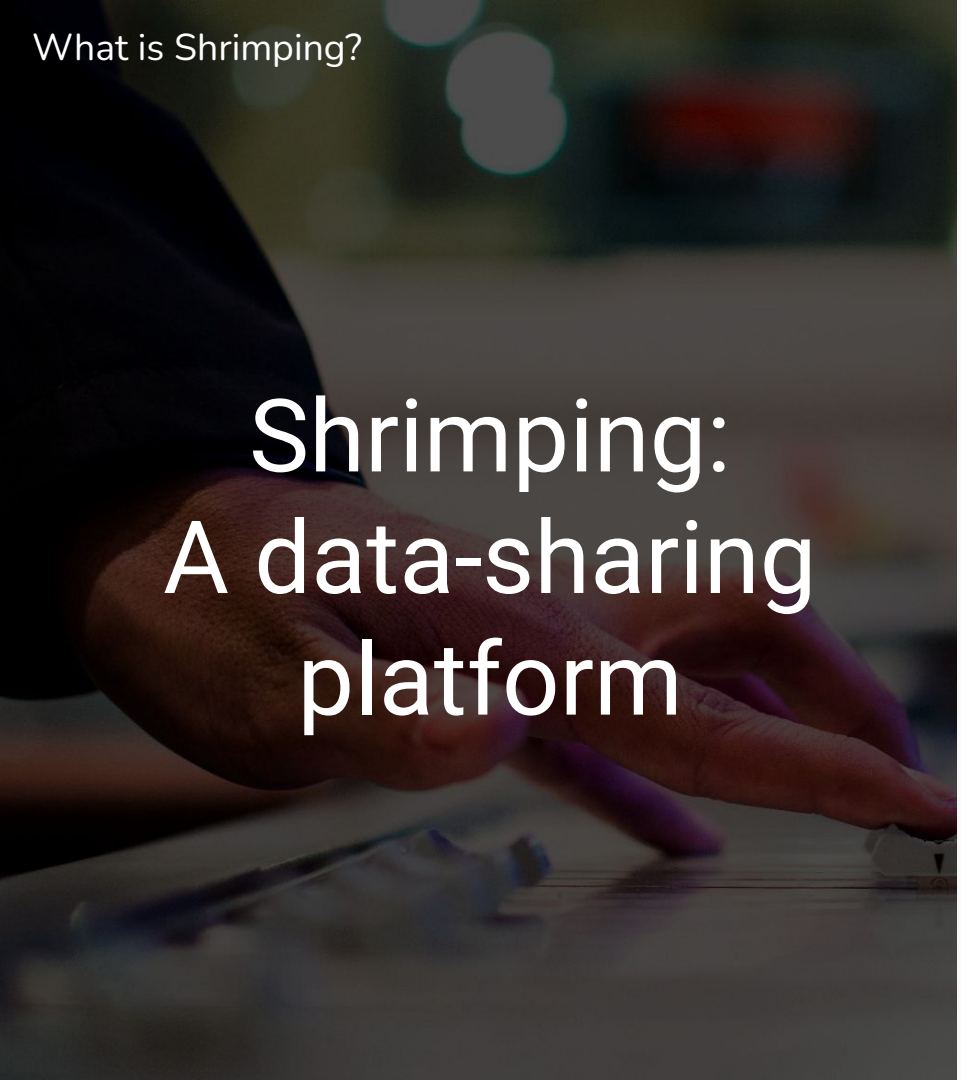
```python
from drivers.webdriver import WebDriver

from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC

def do_find(driver: WebDriver, symbol: str):
    driver.get("https://goodinfo.tw/StockInfo/index.asp")
    ele = driver.find_element_by_xpath(u"//*[@id='txtStockCode']")
    ele.clear()
    ele.send_keys(symbol)
    driver.find_element_by_xpath(u"//input[@value='股票查詢']").click()

    # the following code will wait until the element is visible
    try:
        wait = WebDriverWait(driver, 30)
        wait.until(EC.visibility_of_element_located((By.XPATH, u"//*[@id='imgKC']")))
    except:
        raise Exception("target element is not show up")
```

Ref: https://github.com/FootprintAI/selenium-example/blob/main/examples/goodinfo.py

You have a scraping, we have lots of anti-Scraping tools.

- How to bypass Anti-Scraping Tool
  - Keep Rotate Your IP Address
    - By a well-paid proxy service provider
    - By restricting a certain area of your location
  - Use Real User Agent and valid Referrer
    - UA(User-Agent) are http header which is used identify what browser type you used to visit this website.
    - Keep a list of valid UI and use them randomly.
  - Avoid Periodically Requests
    - Keep random intervals between requests, random delay are helpful.
  - Update your code frequently
    - Anti-scraping tool would change DOM structure frequently, making your script failed to find target elements.
- However, Implementing these guidelines could costly if your application is still below a certain scale.

Ref: https://www.blog.datahut.co/post/web-scraping-how-to-bypass-anti-scraping-tools-on-websites



Shrimping: A data-sharing platform

Shrimping provides a unified way for clients to get human-centric information in a simple , easy, and low cost fashion.

https://get-shrimping.footprint-ai.com/

Source: https://www.kubeflow.org/

Web scraping scenario for Shrimping.

- UI Validation
  - When working with business partners, it is extremely important that your partner has interpreted your product correctly.
- Collecting sell history on an ecommerce platform
  - Track the sell volume of all products on an eCommerce platform.
  - As the number of products could be big (approximately 100M active products), how to get each product's sell records on daily basis is extremely challenging.
- Scraping and analyzing KOL's feeds on social network platforms
  - Find out a KOL and his/her fans preference for retargeting, reselling, or other marketing strategies.
  - Social network platform always implemented anti-bot mechanism, making it hard to collect in a large scale fashion.

Data Analysis with Panda and Numpy

https://pandas.pydata.org/docs/
https://numpy.org/doc/stable/user/index.html#user

```
import numpy as np

import pandas as pd
```

Pandas Series: One-dimensional array of holding **any** data type

```
>>> import numpy as np
>>> a = pd.Series(np.random.randn(5))
>>> a
0    0.427981
1    0.998562
2    0.649786
3    1.341638
4    0.743008
dtype: float64
>>> a[a > a.median()]
1    0.998562
3    1.341638
dtype: float64
>>> a.size
5
>>> a.array
<PandasArray>
[0.4279808645556574, 0.9985615198039794, 0.6497859868724282,
 1.3416378899984038, 0.7430084840615195]
Length: 5, dtype: float64
>>> a.dtype
dtype('float64')
>>> a.to_numpy()
array([0.42798086, 0.99856152, 0.64978599, 1.34163789, 0.74300848])
>>> a2 = a * 2
>>> a2
0    0.855962
1    1.997123
2    1.299572
3    2.683276
4    1.486017
dtype: float64
>>> a3 = a[:-1] + a2[1:]
>>> a3
0         NaN
1    2.995685
2    1.949358
3    4.024914
4         NaN
dtype: float64
```

DataFrame: Think like table, work like table…

- Pandas is a Python library and the de-facto standard for working with **structured** tabular data on Python
- Rich Format Supported including CSV, JSON, Parquet, MS EXCEL, …

asset	max_training	max_validation	mean_training	mean_validation	med_training	med_validation	min_training	min_validation	std_training	std_validation
AAN	2.685913	5.132591	0.046193	0.029751	0.120043	0.237710	-4.324665	-7.955100	1.356045	2.851172
ACC	1.546164	3.473945	0.024481	-0.191317	0.067734	0.000000	-1.493781	-8.192803	0.662752	2.181609
ACGL	1.225804	3.676009	0.035951	-0.350968	0.072673	0.000000	-1.788876	-6.347304	0.607838	1.949640
ADC	1.343051	1.567481	0.118847	-0.090484	0.152312	0.001203	-1.302874	-4.310633	0.603213	1.131439
AEL	3.002969	5.464704	0.026192	-0.094789	0.076017	0.000000	-3.330113	-7.611560	1.290956	2.659290
AFG	1.095037	3.558017	0.005339	-0.387737	0.072613	-0.060772	-2.050200	-7.330084	0.614609	2.013369
AFL	1.159893	3.050265	0.051159	-0.295960	0.072861	-0.038611	-1.414419	-5.840748	0.514807	1.709449
AGM	2.189914	2.572398	0.119186	-0.179405	0.130402	-0.031308	-2.258110	-5.378916	0.944030	1.627170
AGNC	0.934225	2.371371	-0.016914	-0.196734	0.026154	0.159754	-1.373769	-6.580712	0.481505	1.653974

Example of a simple Pandas table with “assets” as index and various numerical columns

Ref: https://towardsdatascience.com/spark-vs-pandas-part-1-pandas-10d768b979f5

Dataframe Indexing

- iloc (position-based indexing) vs loc (label-based indexing)
  - Label-based indexing returns with “match” labels
  - Position-based indexing returns elements with its numeric position index

```
>>> f =  pd.DataFrame({'a':[1,2,3,4,5], 'b':[10,20,30,40,50], 'c': [9,8,7,6,5]})
>>> f.head()
   a   b  c
0  1  10  9
1  2  20  8
2  3  30  7
3  4  40  6
4  5  50  5
>>> f.loc[2:3, 'b']
2    30
3    40
Name: b, dtype: int64
>>> f.iloc[2:3, 1]
2    30
Name: b, dtype: int64
```

Dataframe Join Operation

- Join provides a way to intersect / union two Dataframes
  - Use ‘On’ to specify a specific join key
  - Use ‘How’ to specify we want to have inner / outer join

```
>>> f =  pd.DataFrame({'a':[1,2,3,4,5], 'b':[10,20,30,40,50], 'c': [9,8,7,6,5]})
>>> g = pd.DataFrame({'e':['I','II','III']})
>>> f.join(g, how='outer')
   a   b  c    e
0  1  10  9    I
1  2  20  8   II
2  3  30  7  III
3  4  40  6  NaN
4  5  50  5  NaN
>>> f.join(g, how='inner')
   a   b  c    e
0  1  10  9    I
1  2  20  8   II
2  3  30  7  III
>>> f.join(g, how='inner', on="a")
   a   b  c    e
0  1  10  9   II
1  2  20  8  III
```

Dataframe Multiindex

- Indexing allows you projecting Dataframe, and Multiindex simplify this process

```
>>> adult[adult.columns[:8]].head()
                                    workclass  fnlwgt  marital_status      occupation     relationship    race  capital_gain  capital_loss
education education_num sex     age
 10th     6             Female 17          ?  304873   Never-married               ?        Own-child   White         34095             0
                                17    Private  169658   Never-married   Other-service        Own-child   White             0             0
                                17          ?  158762   Never-married               ?        Own-child   White             0             0
                                17    Private  329783   Never-married           Sales    Other-relative   White             0             0
                                17    Private  139183   Never-married           Sales        Own-child   White             0             0
>>> adult.loc[[' Bachelors'], ['label']]
                                     label
education  education_num sex     age
 Bachelors 13            Female 19    <=50K
                                19    <=50K
                                20    <=50K
                                21    <=50K
                                21    <=50K
...                                     ...
                         Male   90    <=50K
                                90     >50K
                                90    <=50K
                                90    <=50K
                                90    <=50K

[5355 rows x 1 columns]
>>> adult.loc[(' Bachelors', 13, ' Female', 19), ['label']]
                                     label
education  education_num sex     age
 Bachelors 13            Female 19    <=50K
                                19    <=50K
```

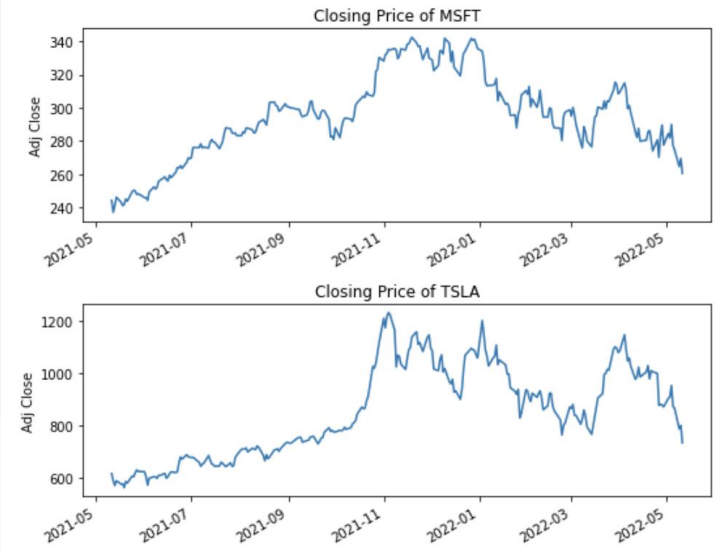
Dataframe Query

- Query is the most used filter to filter rows by its column's value.

```
>>> adult
                                          workclass  fnlwgt       marital_status        occupation  ... capital_loss hours_per_week  native_country   label
education     education_num sex     age                                                             ...
 10th         6             Female 17              ?  304873        Never-married                 ?  ...            0             32   United-States   <=50K
                                   17        Private  169658        Never-married     Other-service  ...            0             21   United-States   <=50K
                                   17              ?  158762        Never-married                 ?  ...            0             20   United-States   <=50K
                                   17        Private  329783        Never-married             Sales  ...            0             10   United-States   <=50K
                                   17        Private  139183        Never-married             Sales  ...            0             15   United-States   <=50K
...                                             ...     ...                  ...               ...  ...          ...            ...             ...     ...
 Some-college 10             Male   84   Self-emp-inc  172907   Married-civ-spouse             Sales  ...            0             35   United-States    >50K
                                   90        Private   52386        Never-married     Other-service  ...            0             35   United-States   <=50K
                                   90        Private   52386        Never-married     Other-service  ...            0             35   United-States   <=50K
                                   90   Self-emp-not-inc  282095   Married-civ-spouse   Farming-fishing  ...            0             40   United-States   <=50K
                                   90              ?  225063        Never-married                 ?  ...            0             10           South   <=50K

[32561 rows x 11 columns]
>>> adult.query("workclass == ' Private'")
                                    workclass  fnlwgt       marital_status       occupation  ... capital_loss hours_per_week  native_country   label
education     education_num sex     age                                                       ...
 10th         6             Female 17   Private  169658        Never-married    Other-service  ...            0             21   United-States   <=50K
                                   17   Private  329783        Never-married            Sales  ...            0             10   United-States   <=50K
                                   17   Private  139183        Never-married            Sales  ...            0             15   United-States   <=50K
                                   17   Private  130125        Never-married    Other-service  ...            0             20   United-States   <=50K
                                   17   Private   27032        Never-married            Sales  ...            0             12   United-States   <=50K
...                                        ...     ...                  ...              ...  ...          ...            ...             ...     ...
 Some-college 10             Male   79   Private  121318   Married-civ-spouse     Adm-clerical  ...            0             20   United-States   <=50K
                                   80   Private   87518        Never-married   Prof-specialty  ...         1816             60   United-States   <=50K
                                   81   Private  122651   Married-civ-spouse            Sales  ...            0             15   United-States   <=50K
                                   90   Private   52386        Never-married    Other-service  ...            0             35   United-States   <=50K
                                   90   Private   52386        Never-married    Other-service  ...            0             35   United-States   <=50K

[22696 rows x 11 columns]
```

Dataframe Visualization

```
plt.figure(figsize=(15, 6))
plt.subplots_adjust(top=1.25, bottom=1.2)

company_list = [MSFT, AAPL, TSLA, VOO]

for i, company in enumerate(company_list, 1):
    plt.subplot(2, 2, i)
    company['Adj Close'].plot()
    plt.ylabel('Adj Close')
    plt.xlabel(None)
    plt.title(f"Closing Price of {tech_list[i - 1]}")

plt.tight_layout()
```

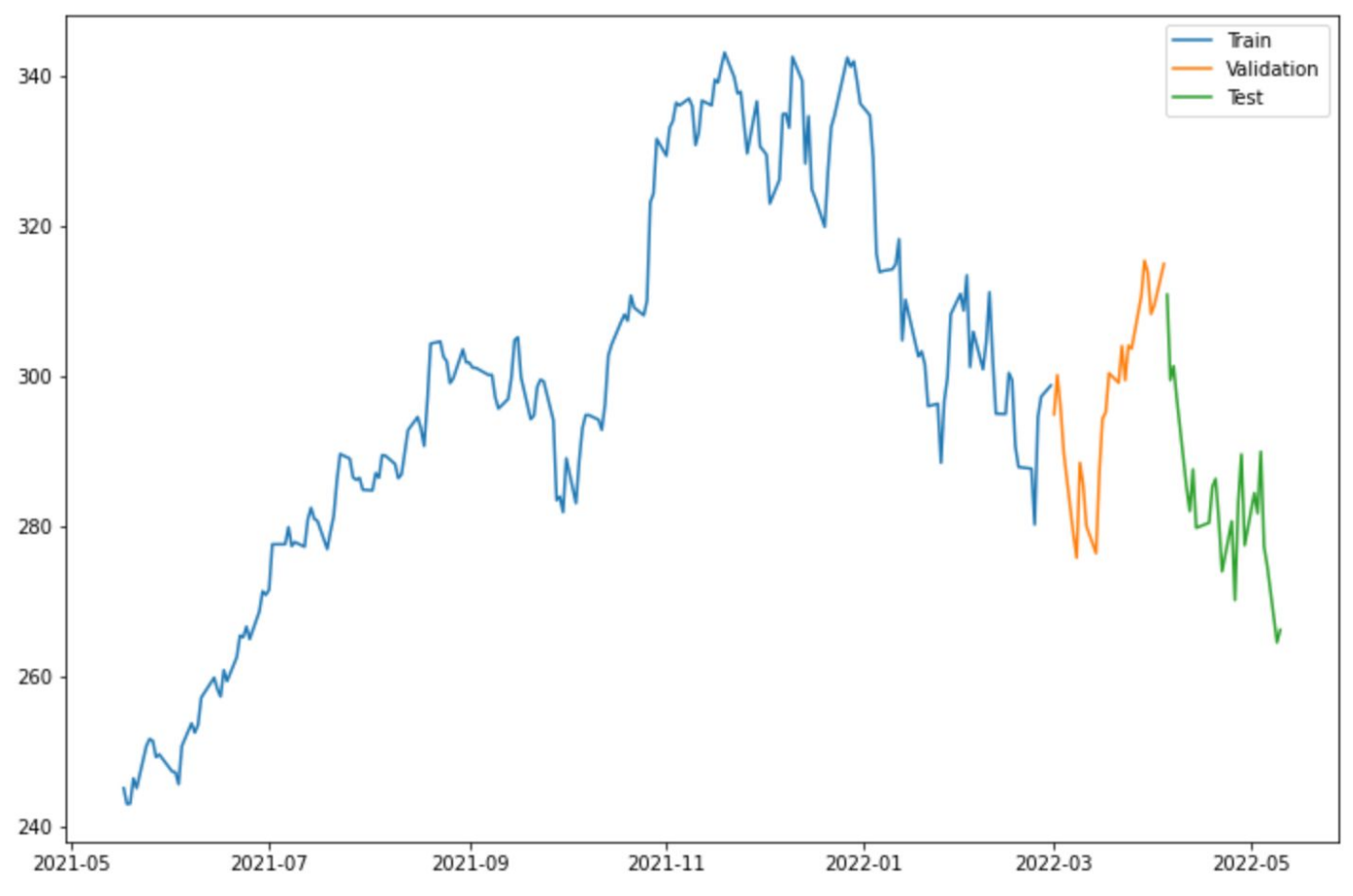


Ref: https://github.com/FootprintAI/kubeflow-workshop/blob/main/tutorials/stockprice-with-lstm/1.visualize-and-build-stockprice-model.ipynb

Financial Model Analysis with Kubeflow

Can LSTM model be used to predict the stock price?

How to predict stock price for the next day?



What sequences means in a series?

Input: [300, 320, 310, 350, 390]

t0	300
t1	320
t2	310
t3	350
t4	390
t5	?

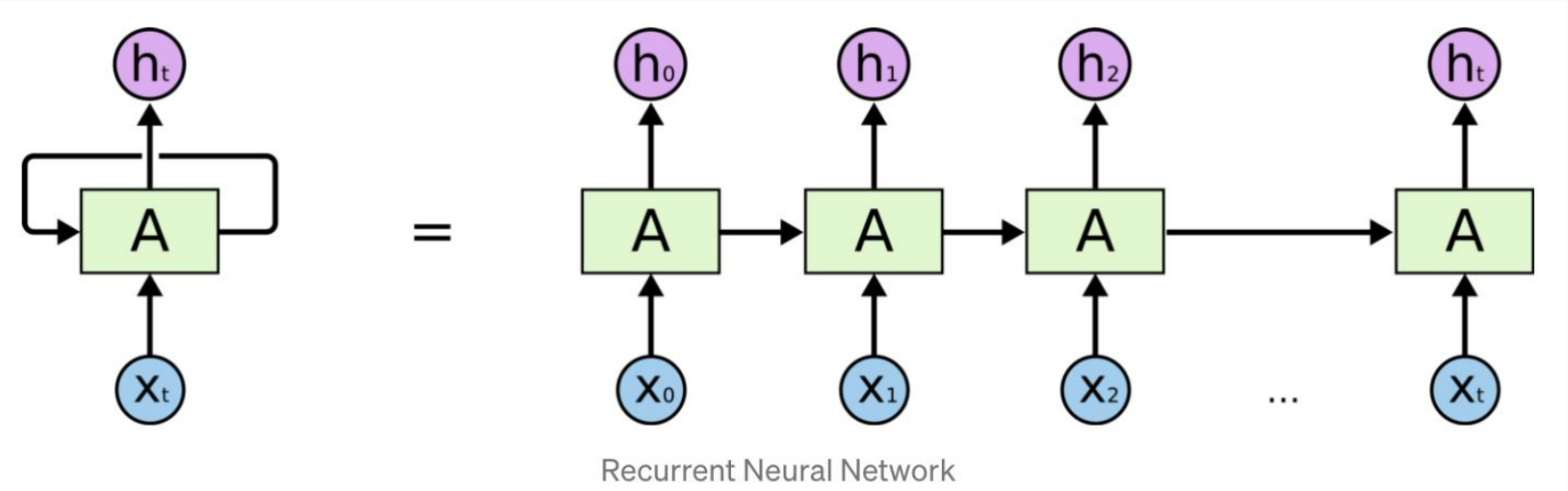
What sequences means in a sentence?

Input: I am a good guy

t0	I
t1	am
t2	a
t3	good
t4	guy
t5	?

What Is RNN (Recurrent Neural Network)?

Recurrent Neural Network (RNN) takes decisions on CURRENT (Xt) and PREVIOUS (Xt-1) inputs. Especially useful in topics including machine translation, speech recognition.



Recurrent Neural Network

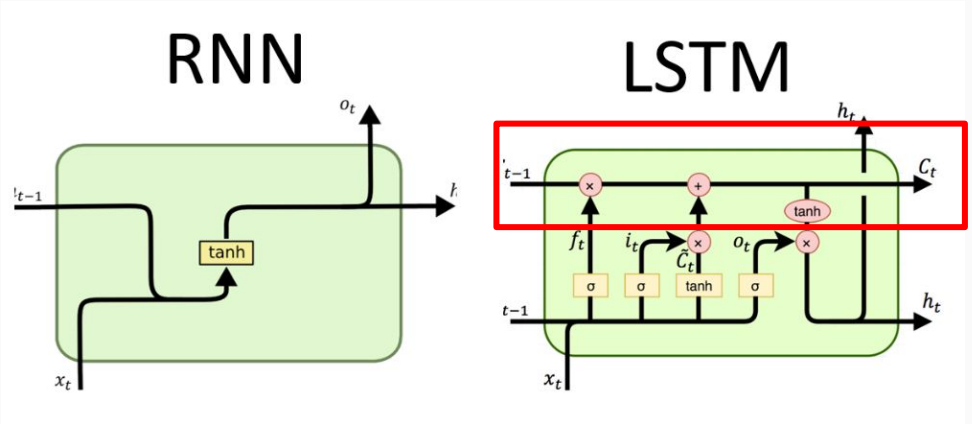
https://towardsdatascience.com/introduction-to-recurrent-neural-network-27202c3945f3

What Is LSTM (Long-short term memory)?

RNN only remember the latest things from X and it didn't remember(no memory) anything before at the beginning.

LSTM provides an information highway to let the neuron to selectively choose

1. forget from its memory (focus on the current inputs)
2. Listens to what information it added into memory (though information highway)



https://towardsdatascience.com/introduction-to-recurrent-neural-network-27202c3945f3

https://www.quora.com/How-is-LSTM-different-from-RNN-In-a-layman-explanation

How to formulate a sequence into a trainable dataset?

t0	t1	t2	t3	t4
300	320	310	350	390

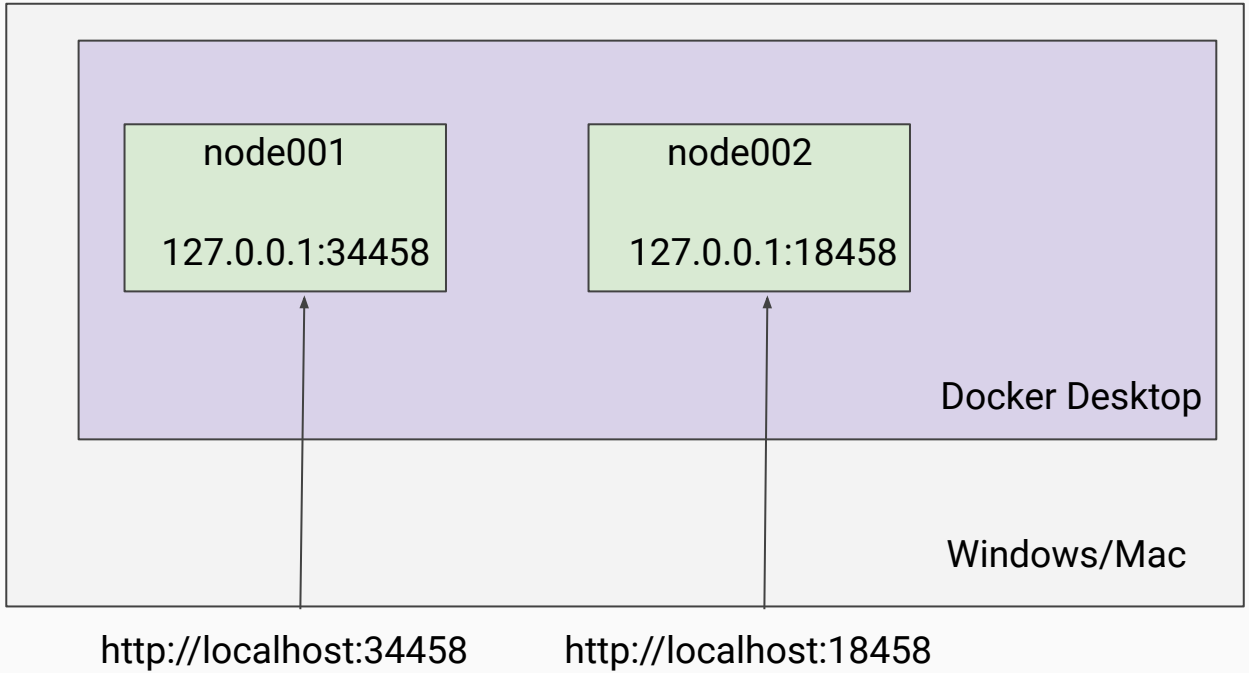
Windows size 3, we looks last three days data

Feature 1	Feature 2	Feature 3	Target
300	320	310	350
320	310	350	390

Environment Setup

Deployment Overview

Provisioning with multikf https://github.com/FootprintAI/multikf/tree/main/docs

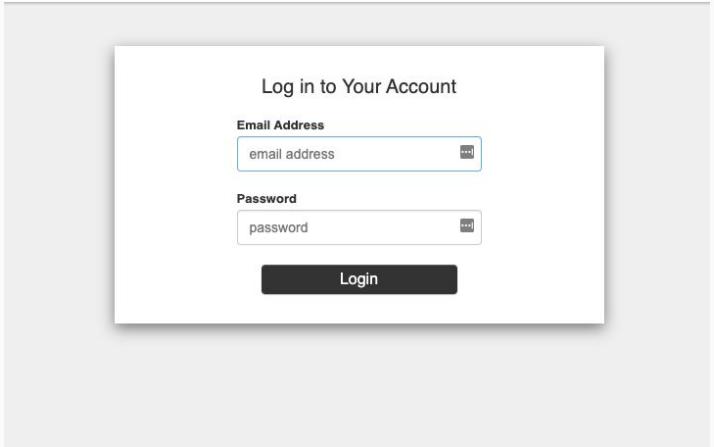


Deployment Overview: few steps setup

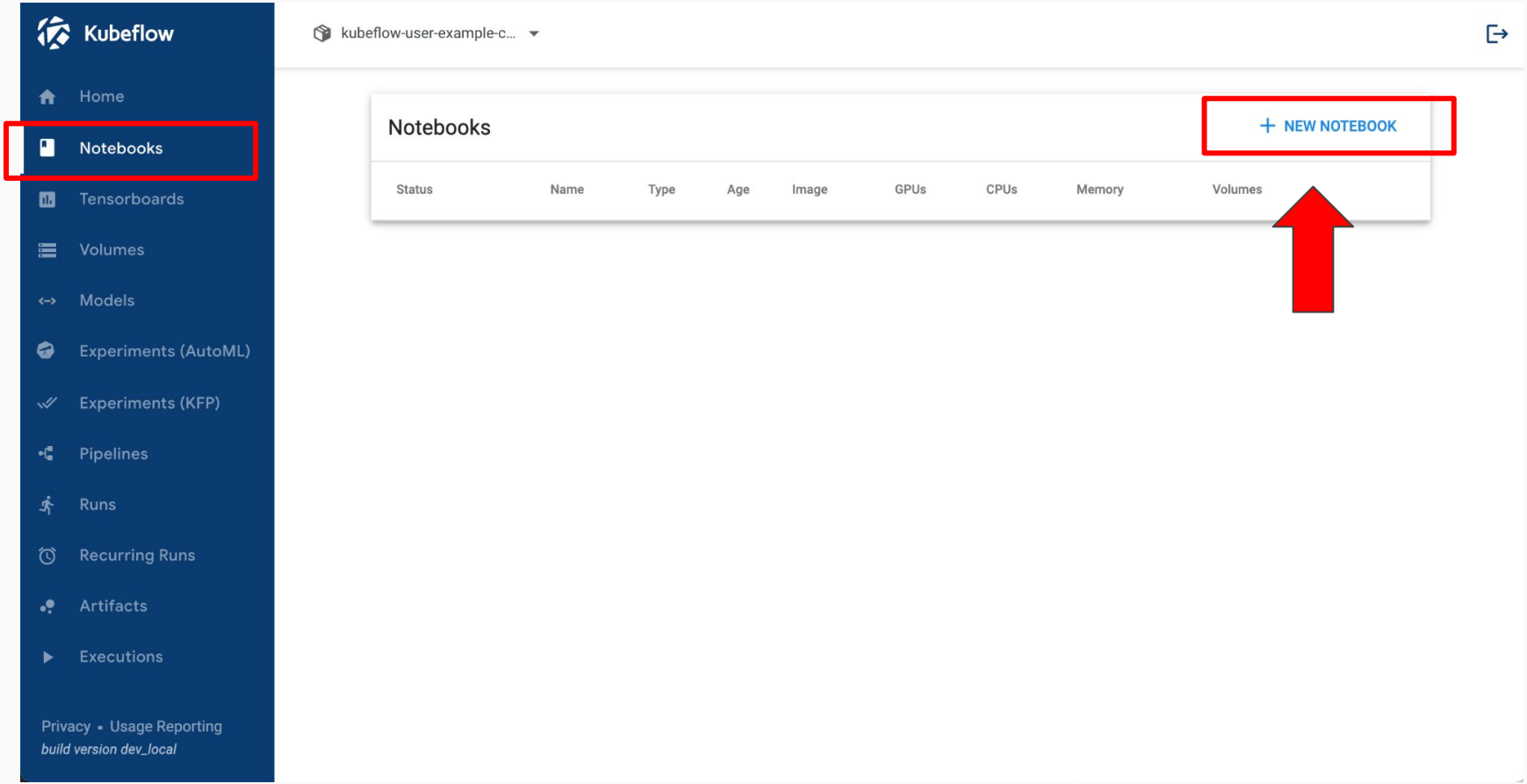
```
// install dockerd (windows)
https://github.com/FootprintAI/kubeflow-workshop/blob/main/install/windows/dockerd.bat.md

// install multikf (windows)
wget https://github.com/FootprintAI/multikf/raw/main/build/multikf.windows.exe
chmod +x multikf.windows.exe

// add an instances
./multikf.windows.exe add node002

// connect kubeflow
./multikf.windows.exe connect kubeflow node002
```

Wait! 所以我說那個帳號密碼呢?

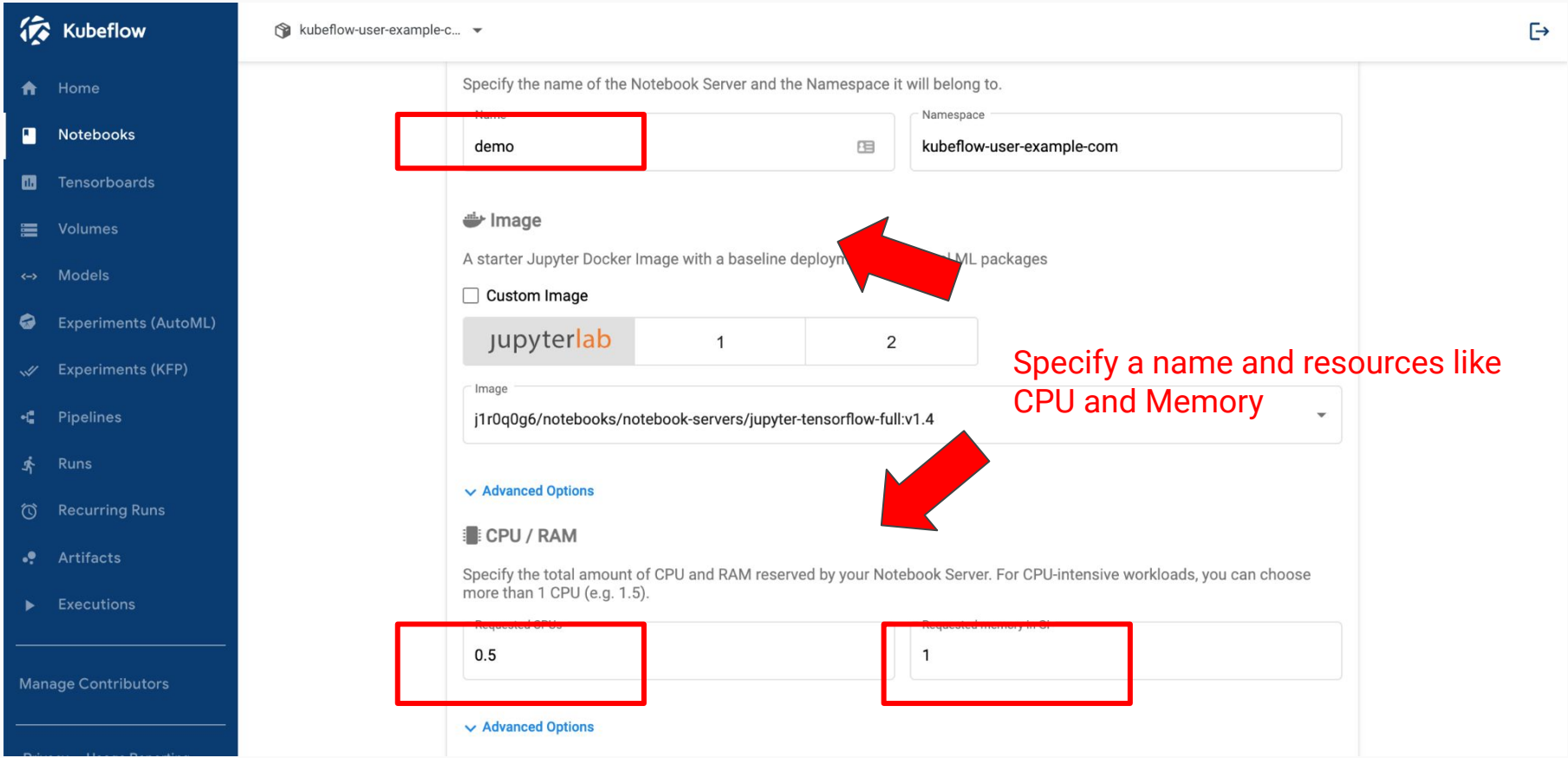
Account: user@example.com
Password: 12341234



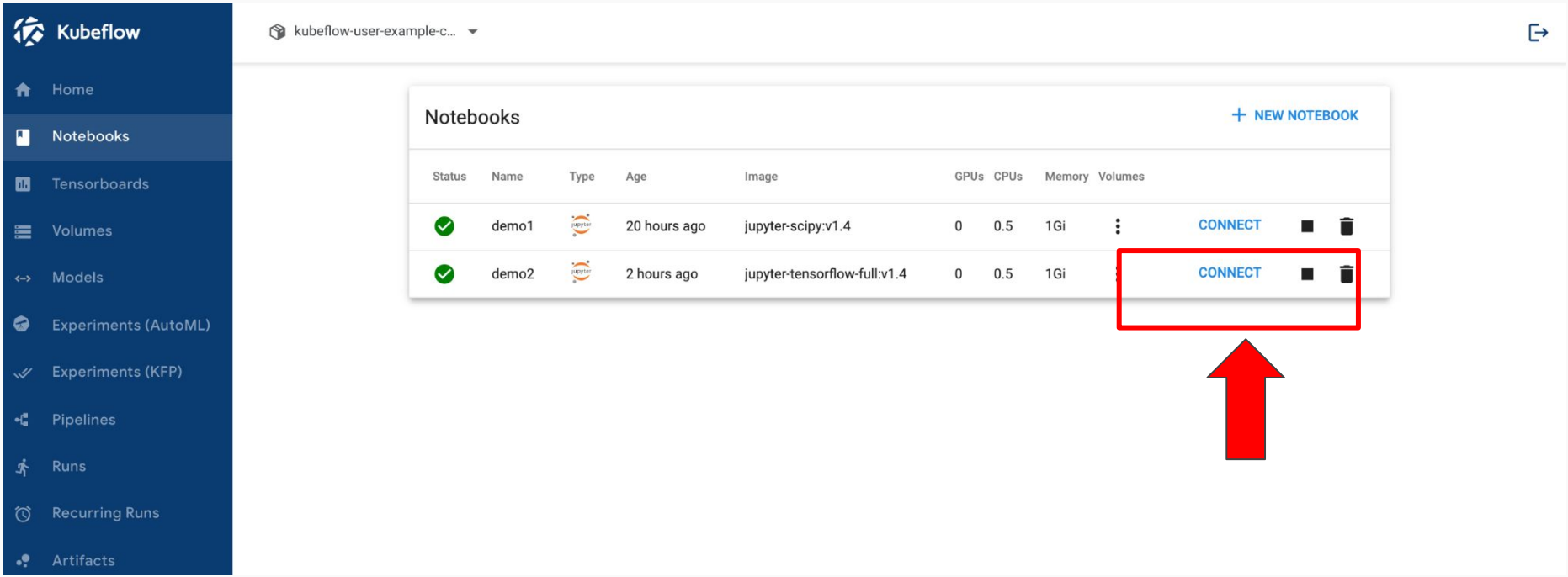
Step1: Use Notebook as Online IDE (1/3)



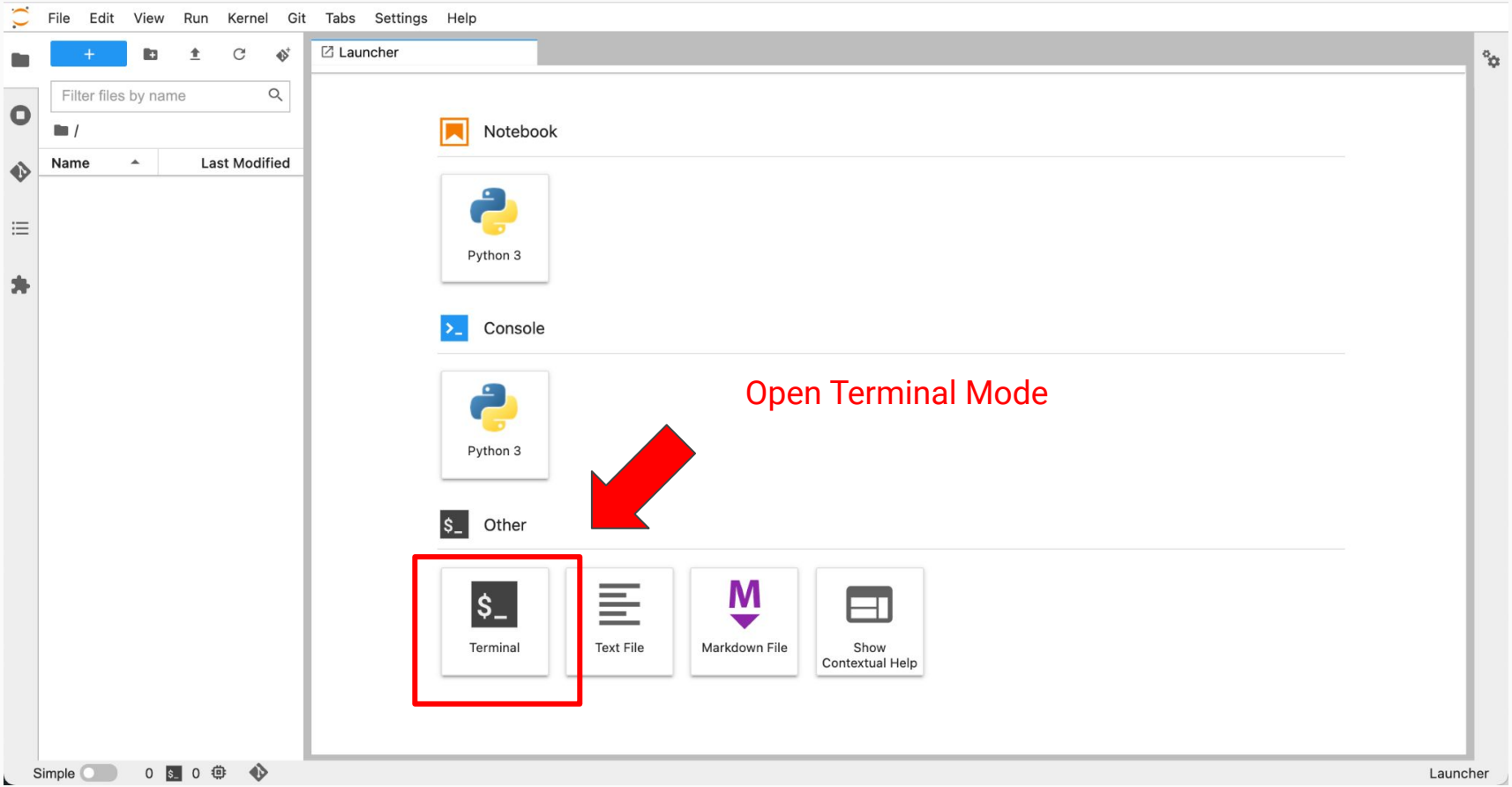
Step1: Use Notebook as Online IDE (2/3)



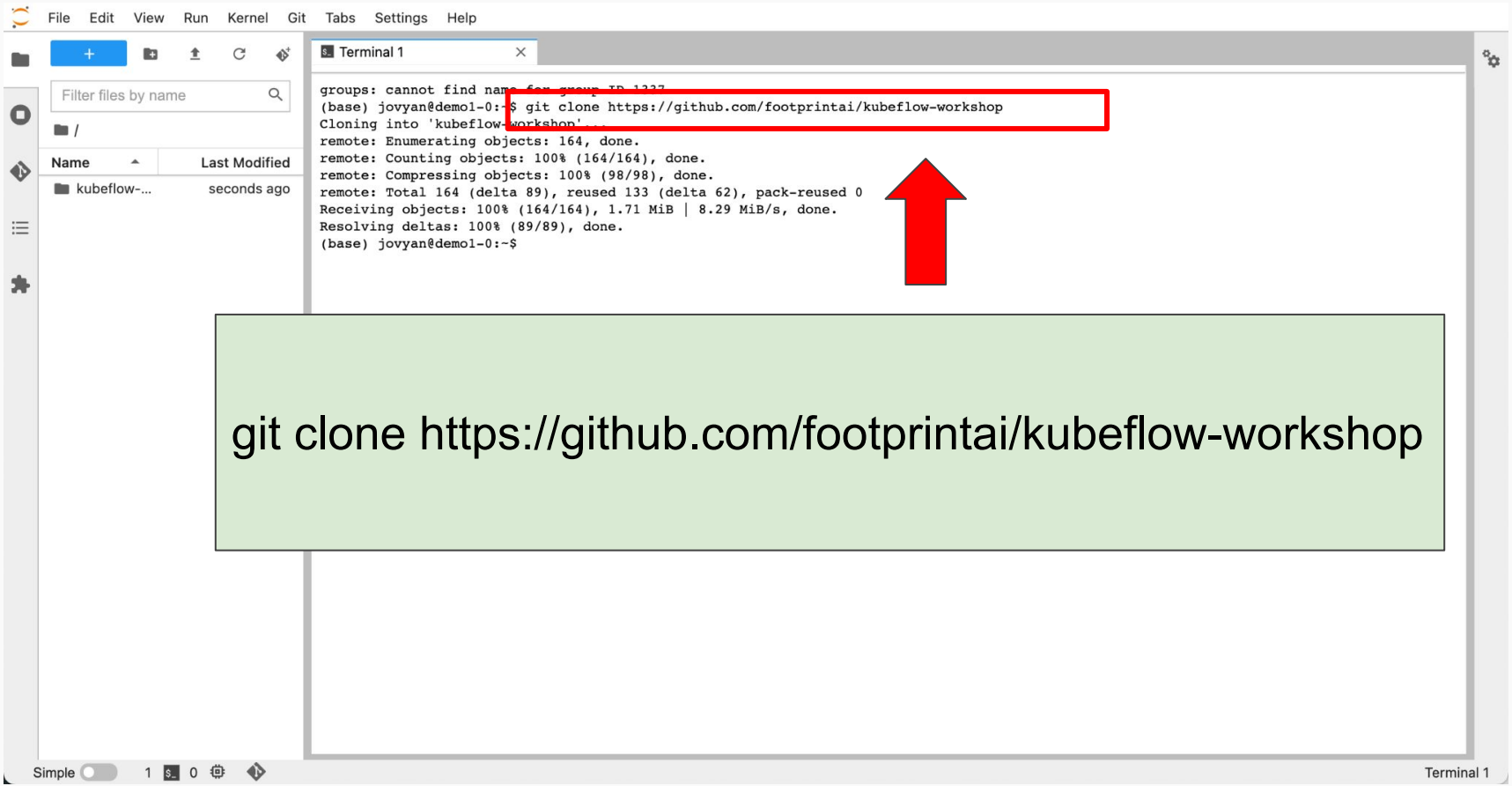
Step1: Use Notebook as Online IDE (3/3)



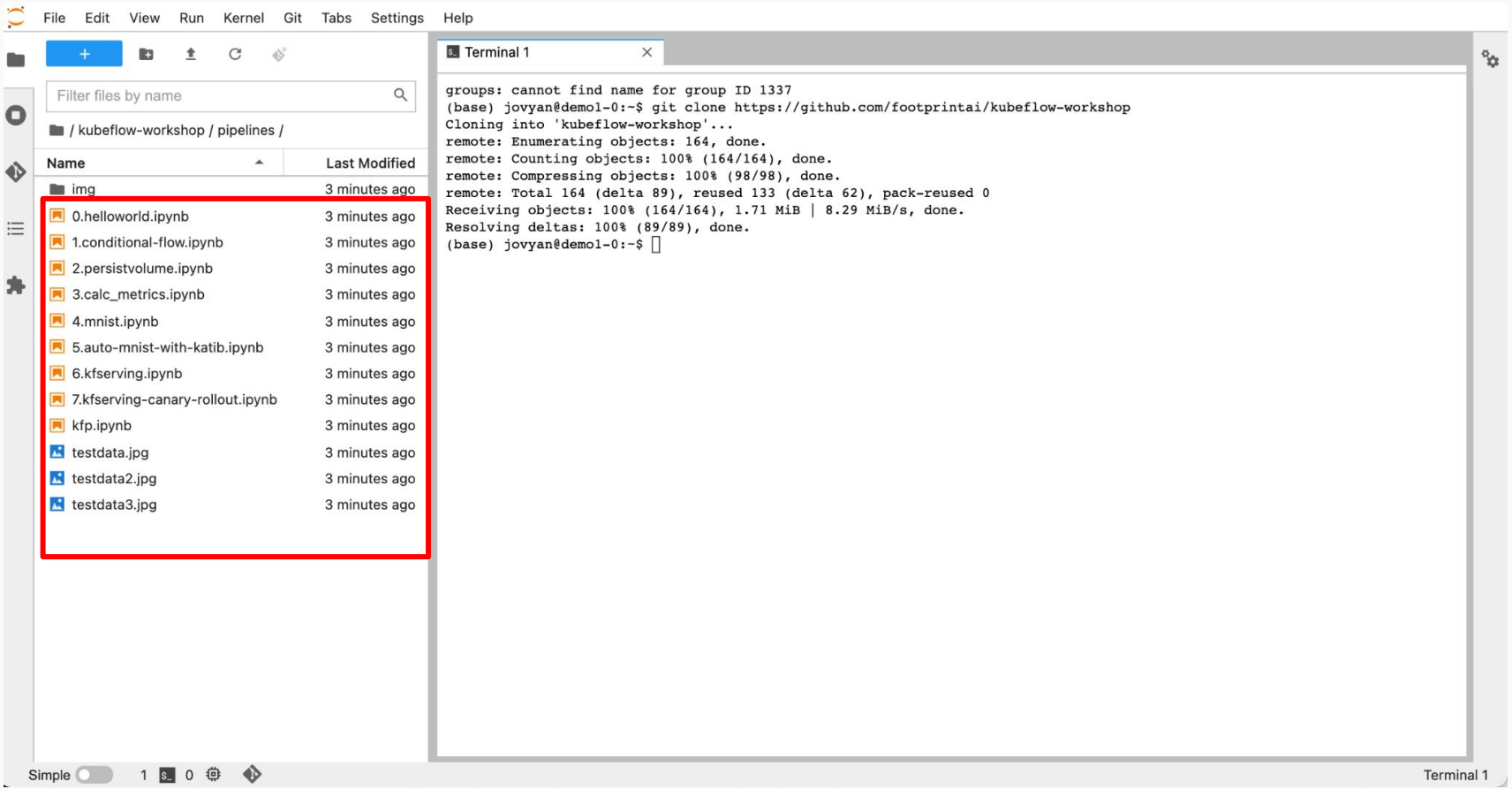
Step2: Use terminal to download the materials (1/3)



Step2: Use terminal to download the materials (2/3)

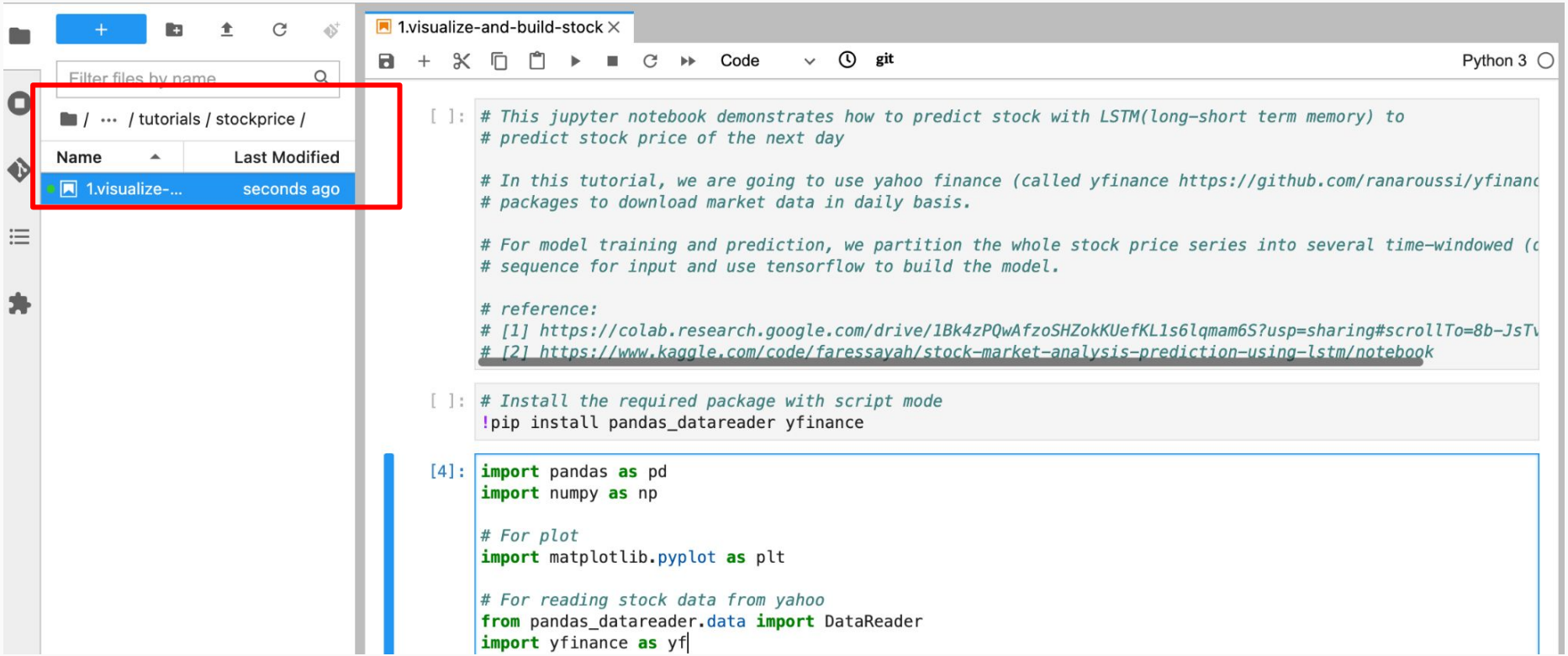


Step2: Use terminal to download the materials (3/3)



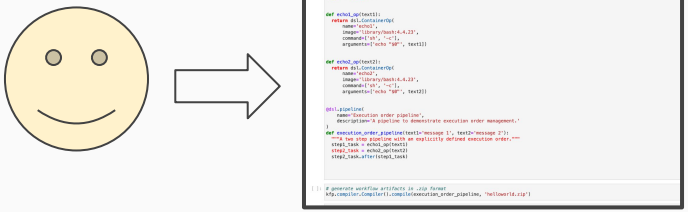
Financial Model Builder

Step3: Open financial model ipynb under tutorials/stockprice



Kubeflow Terms

Kubeflow Terms

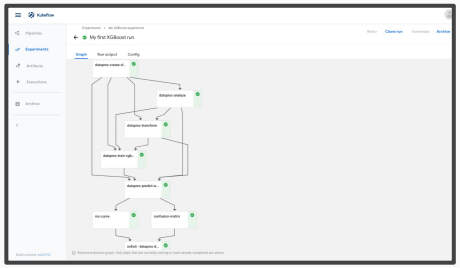


Pipeline Code

Compiled

Workflow Resource

Create a Pipeline



Create Run

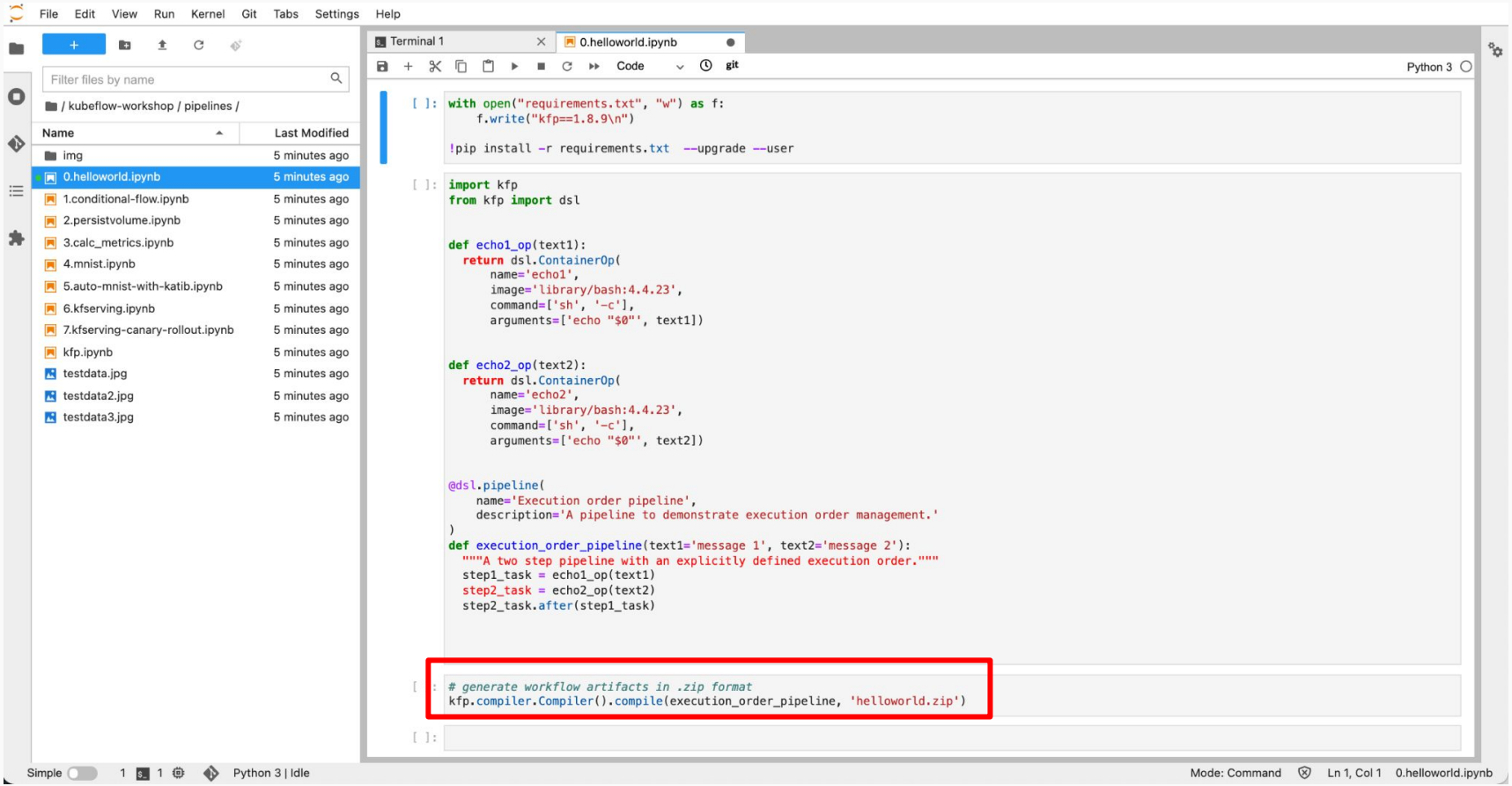
Experiment1

Experiment2

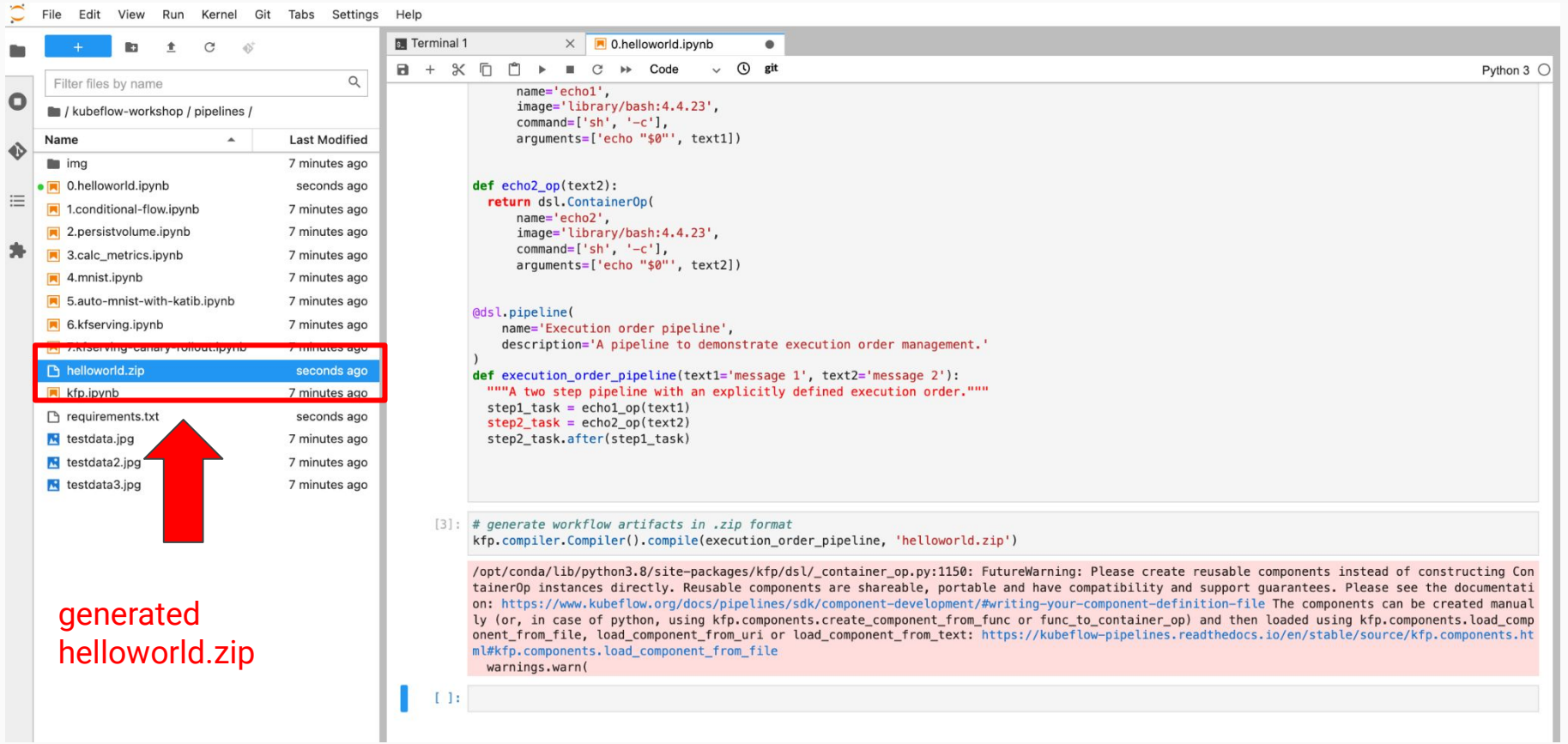
Runs

Hello World Example

Step4: Compile helloworld.ipynb (1/2)

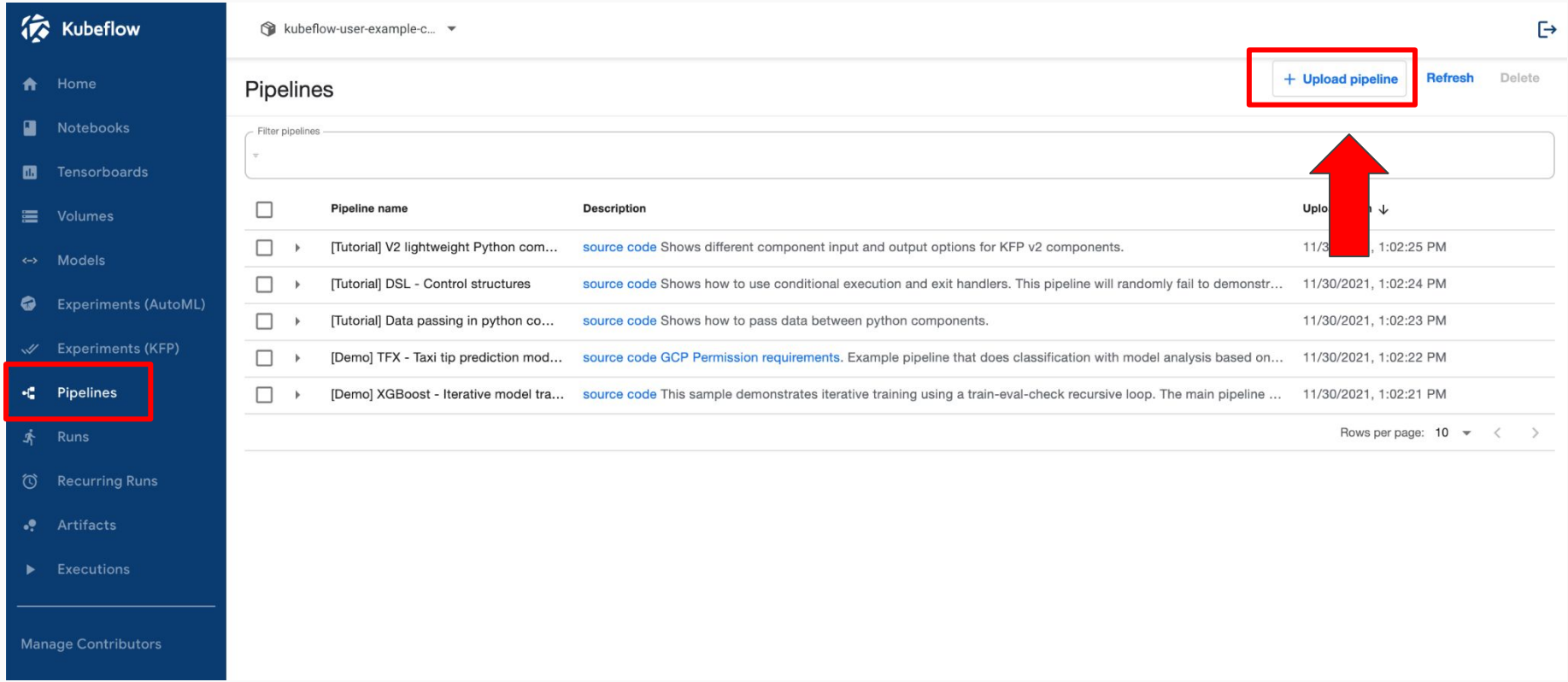


Step4: Compile helloworld.ipynb (2/2)

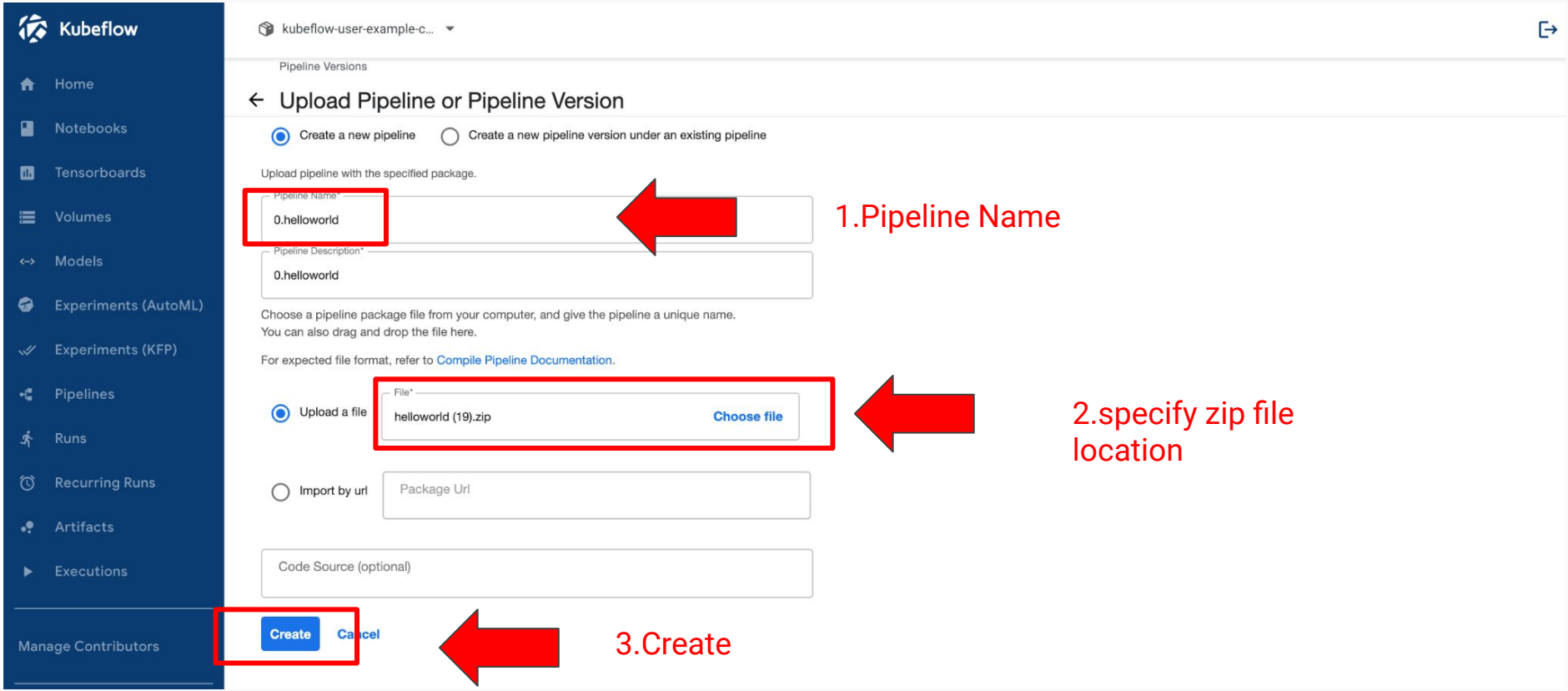


generated helloworld.zip

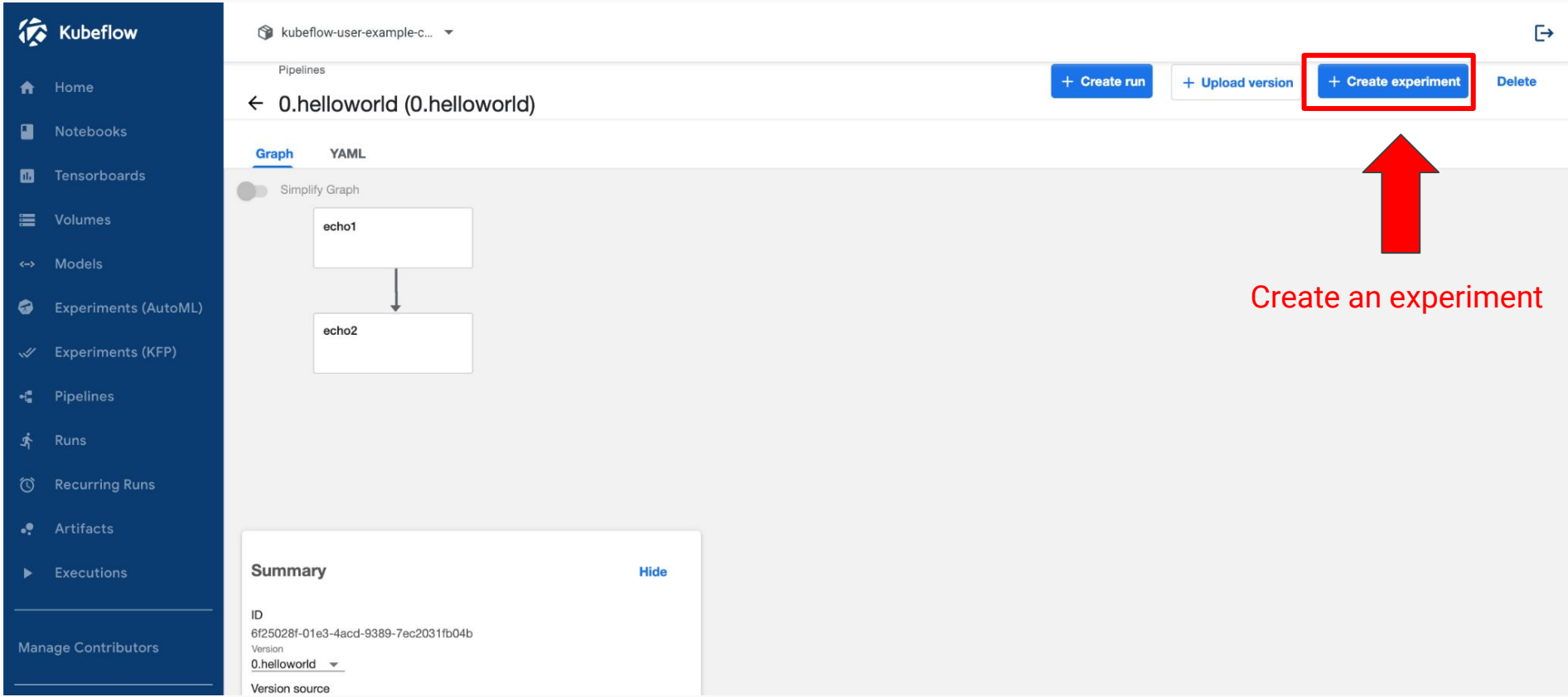
Step4: Create a Pipeline (1/7)



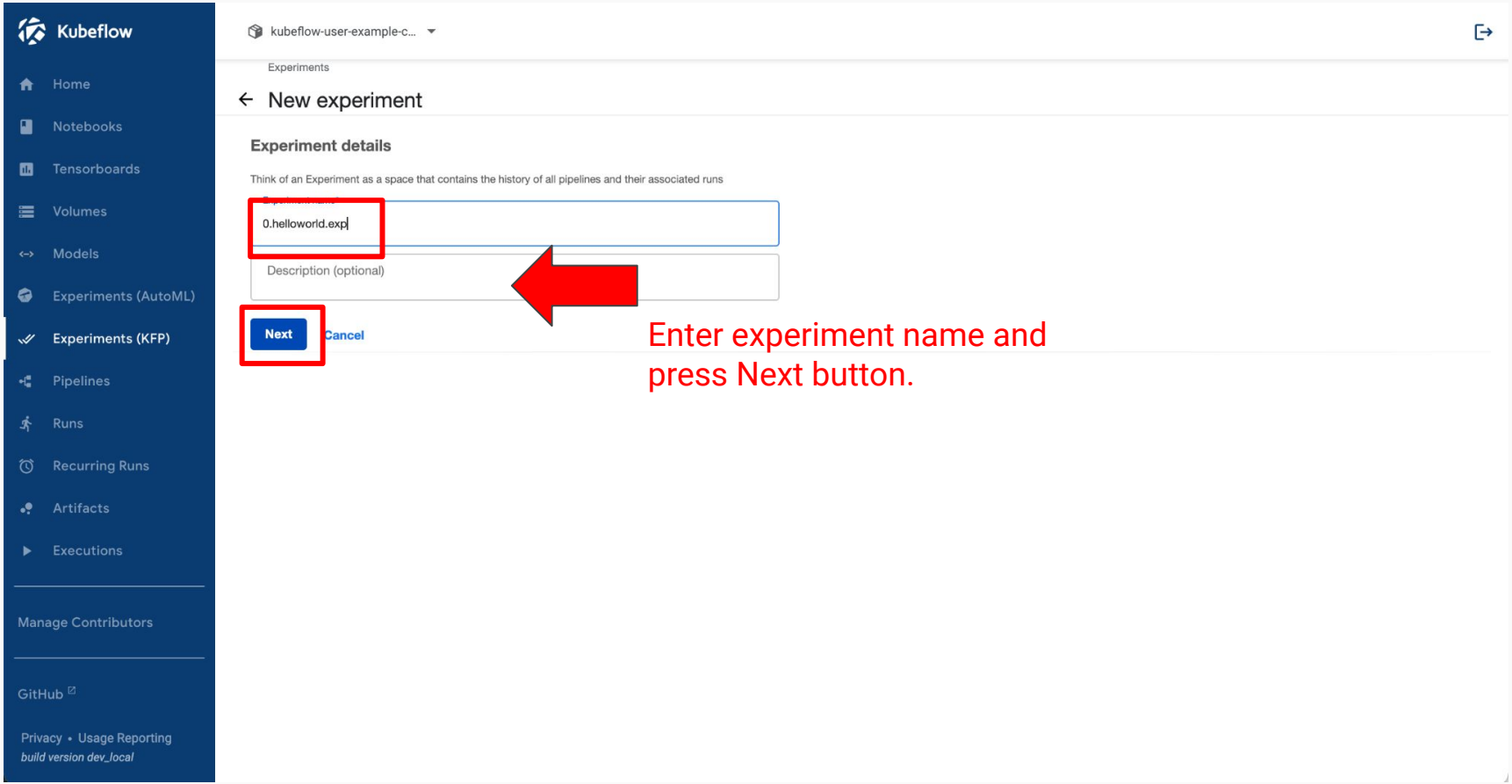
Step4: Create a Pipeline (2/7)



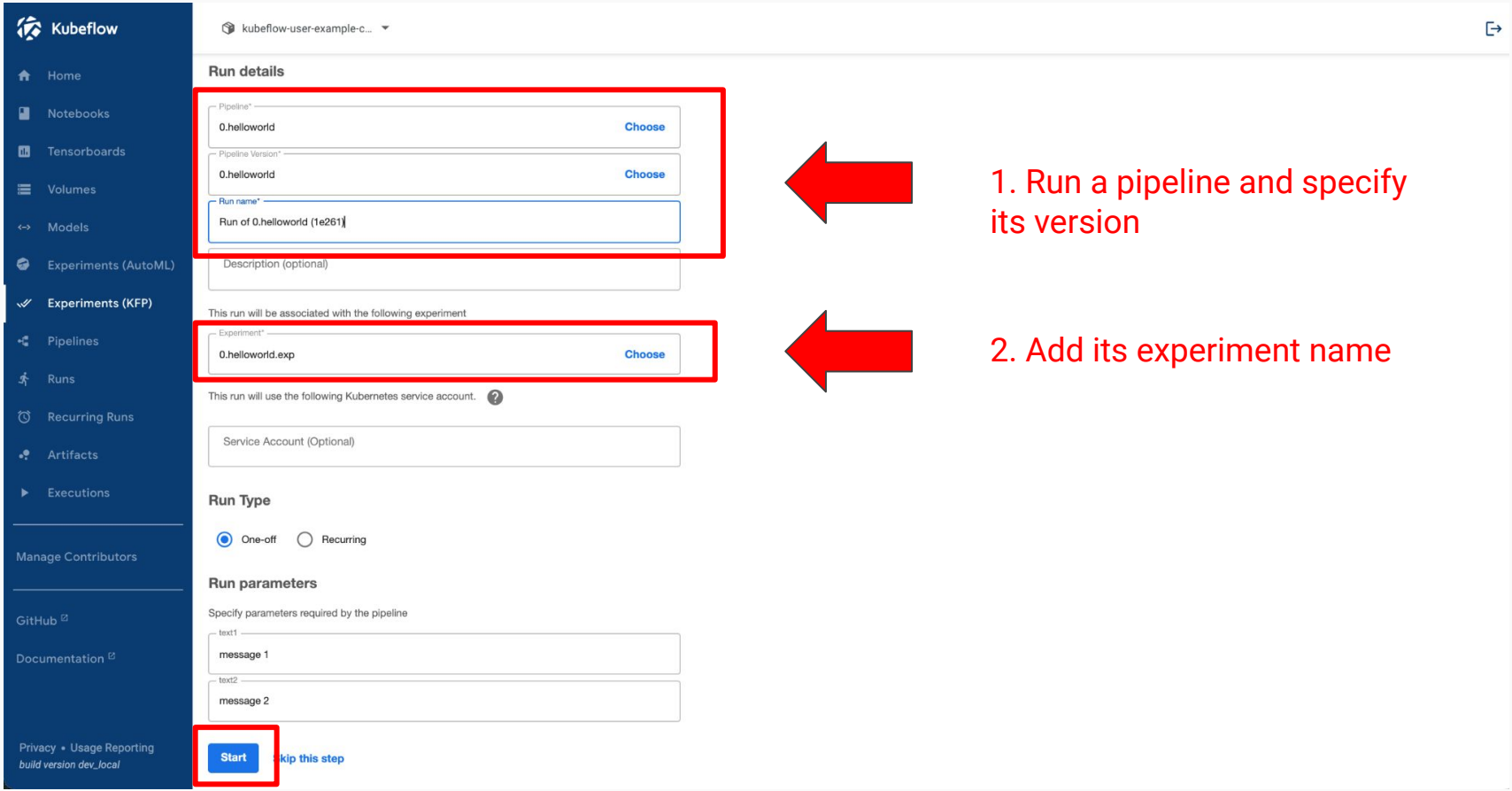
Step4: Create a Pipeline (3/7)



Step4: Create a Pipeline (4/7)



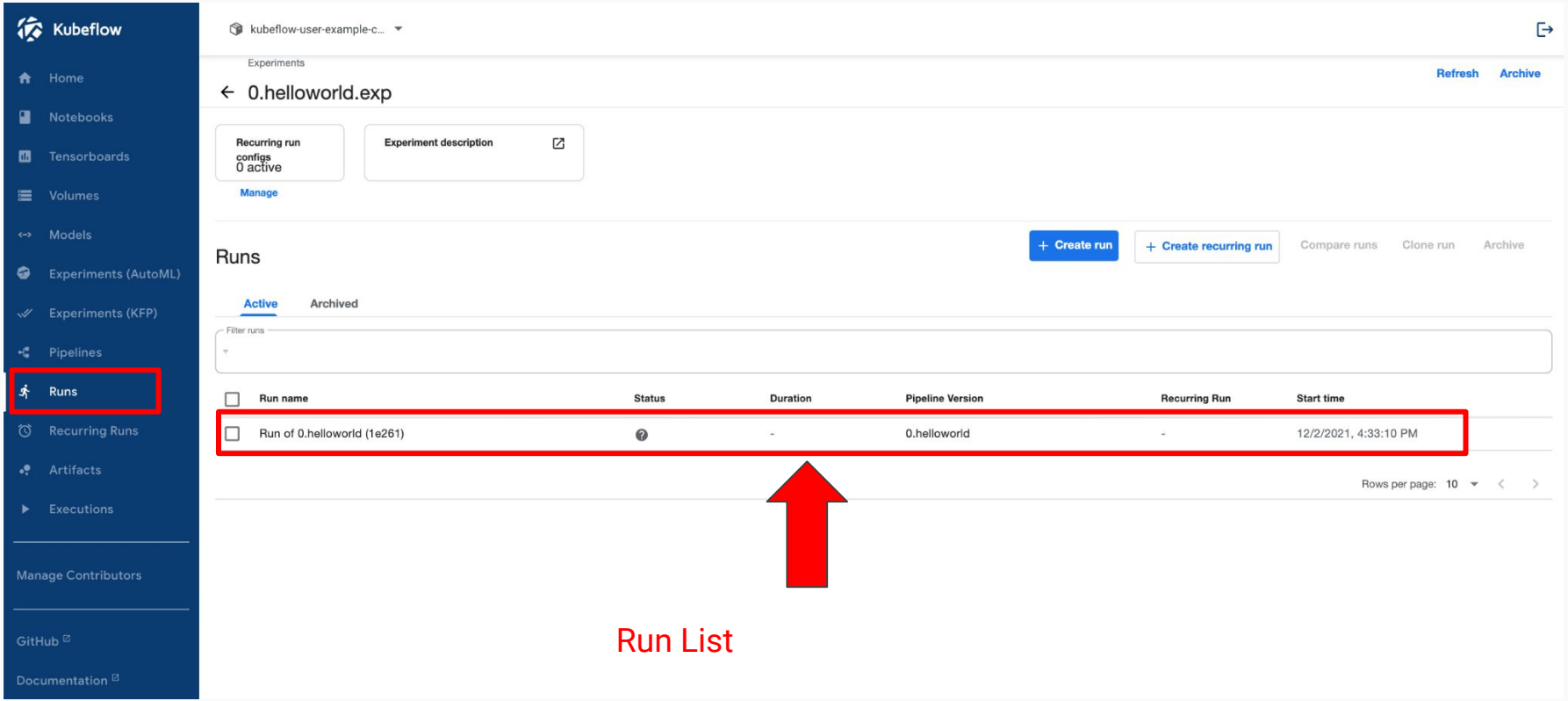
Step4: Create a Pipeline (5/7)



1. Run a pipeline and specify its version

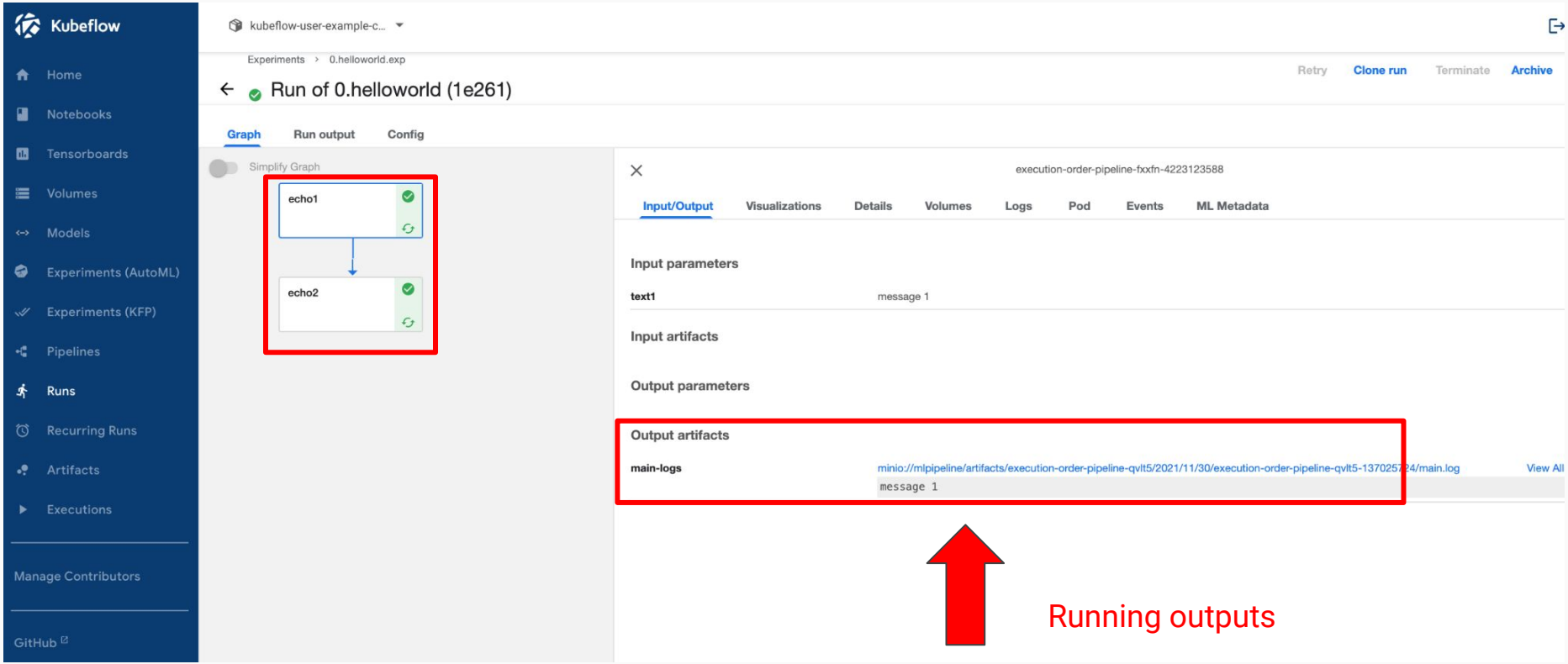
2. Add its experiment name

Step4: Create a Pipeline (6/7)



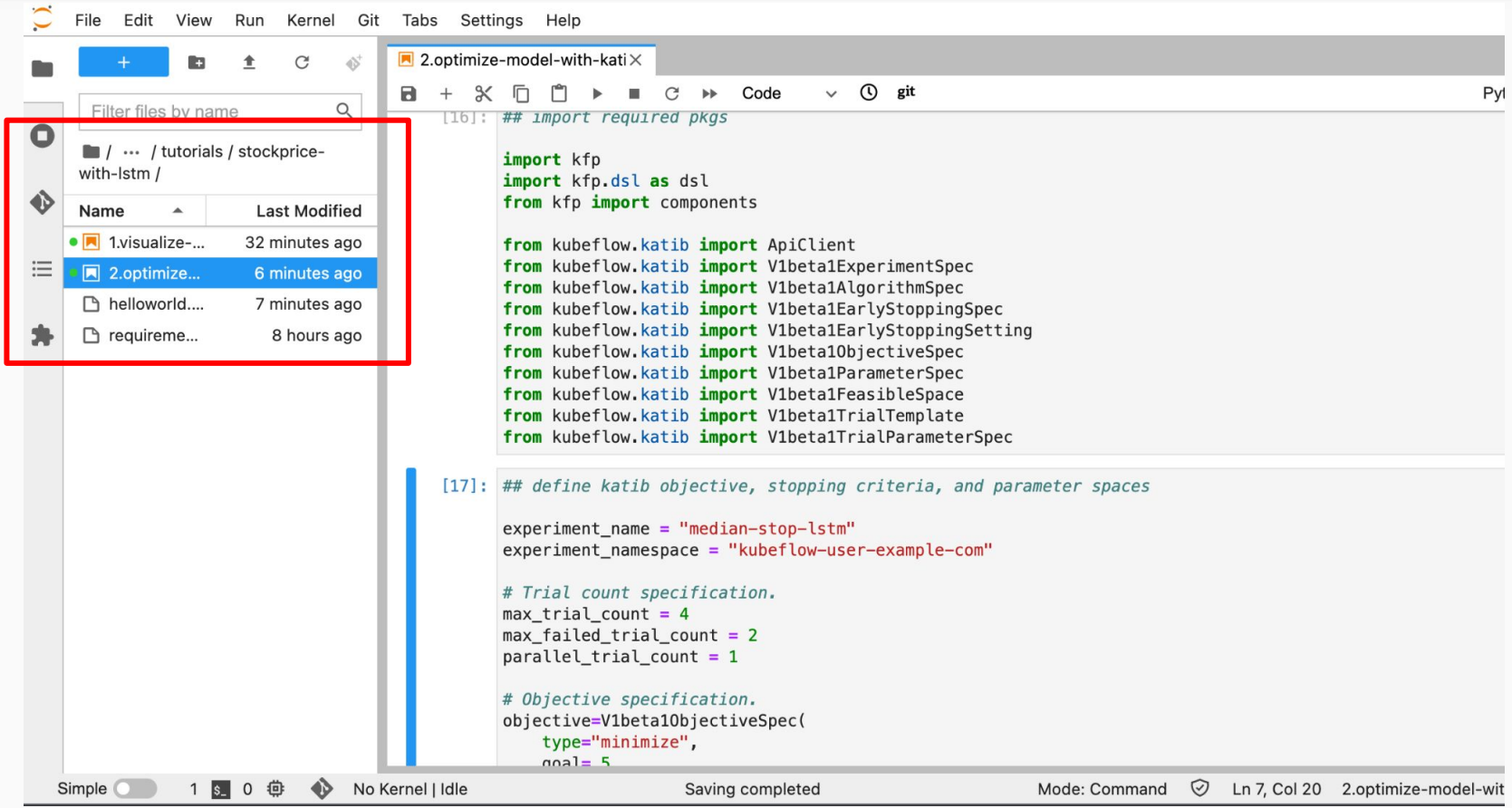
Run List

Step4: Create a Pipeline (7/7)

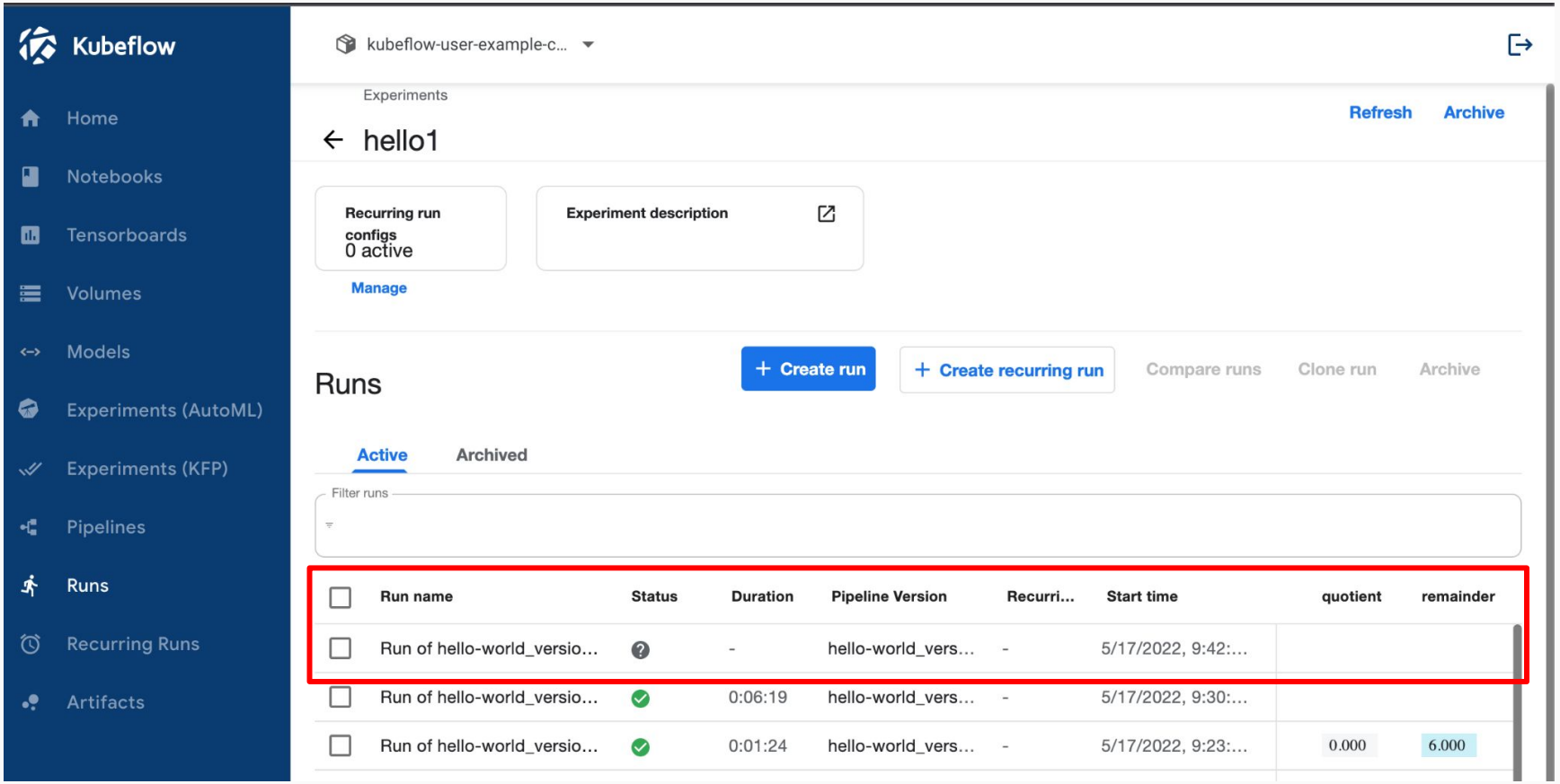


Hyperparameter Example

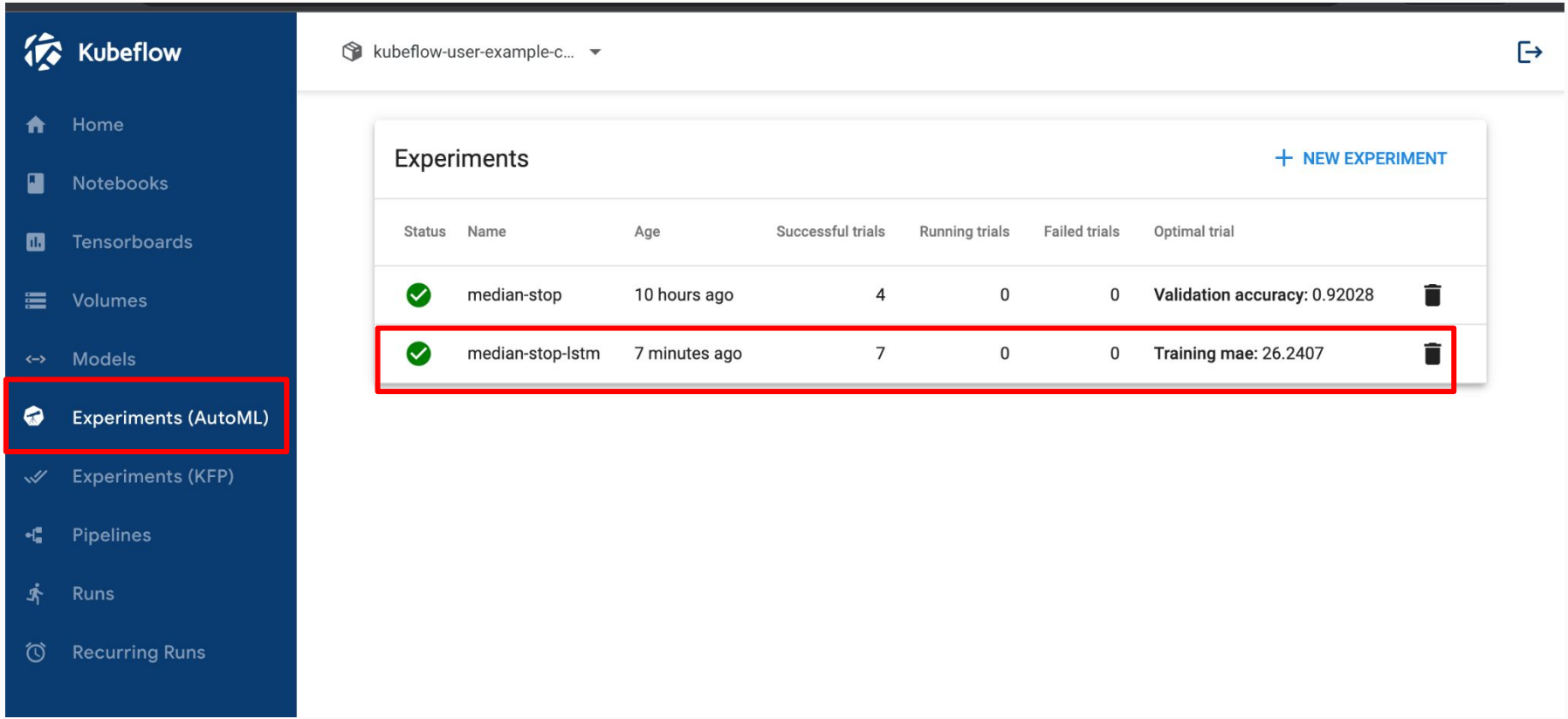
Step5: Hyperparameter tuning with katib (1/4)



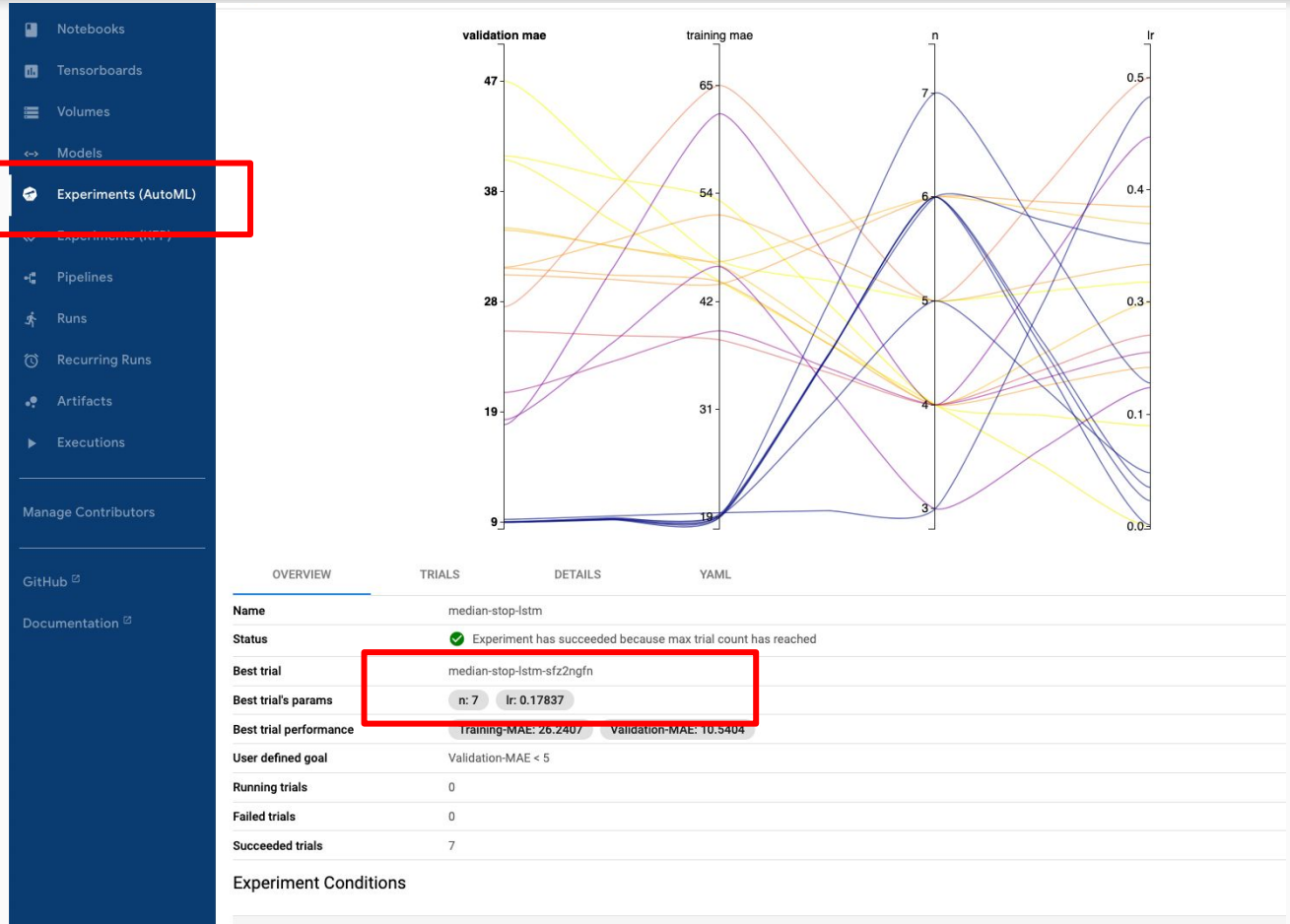
Step5: Hyperparameter tuning with katib (2/4)

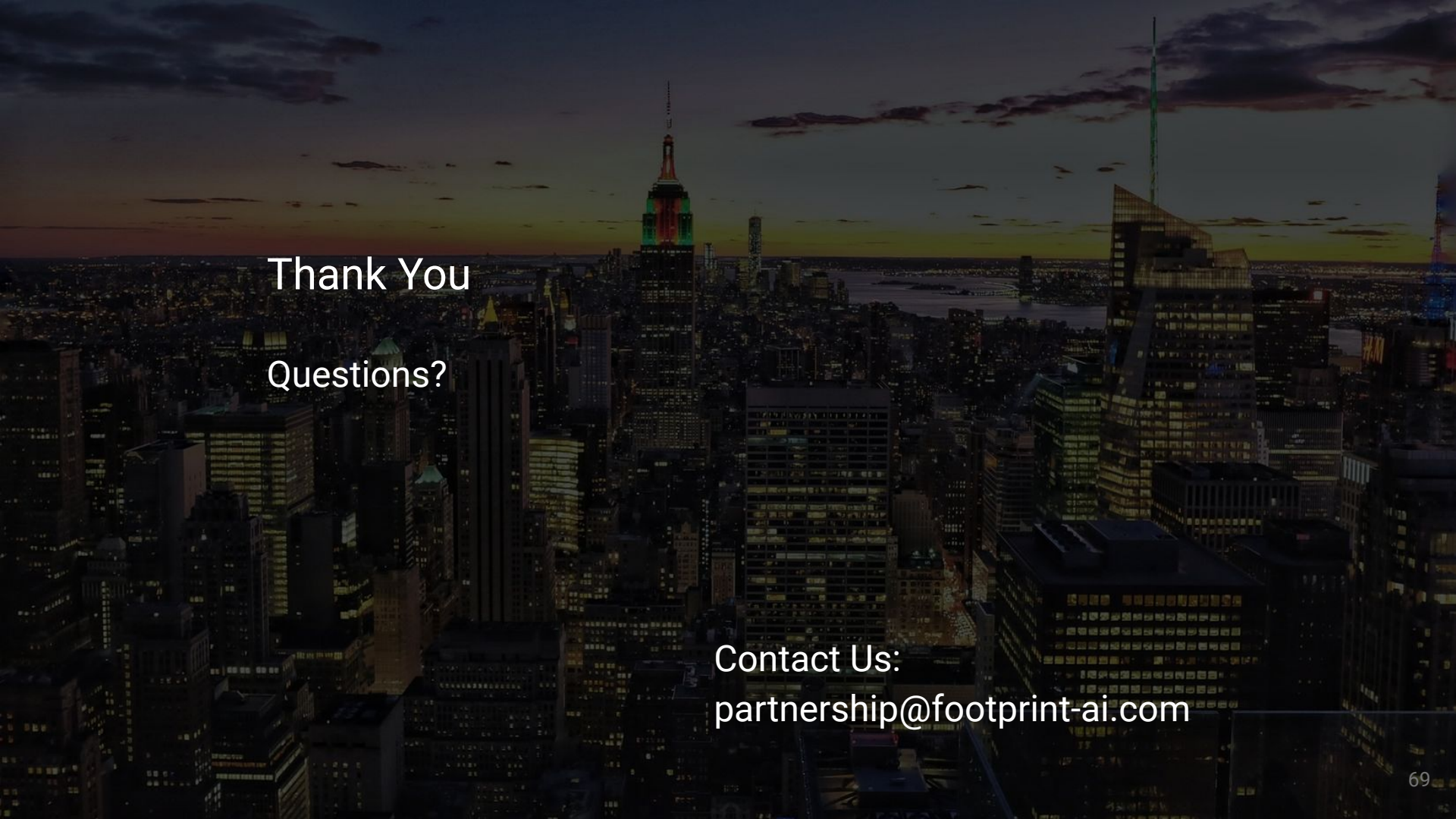


Step5: Hyperparameter tuning with katib (3/4)



Step5: Hyperparameter tuning with katib (4/4)




Thank You
Questions?
Contact Us:
partnership@footprint-ai.com

Q&A

Materials

- Slides:
  - https://github.com/FootprintAI/talks/tree/main/slides
- Multikf
  - https://github.com/FootprintAI/multikf
- Kubeflow Workshop
  - https://github.com/footprintai/kubeflow-workshop