

# Fairness and Explainable AI

- An End-to-end example on Tensorboard with Kubeflow

葉信和 / Hsin-Ho Yeh  
Founder @ 信誠金融科技  
[hsinho.yeh@footprint-ai.com](mailto:hsinho.yeh@footprint-ai.com)

# 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 Igloolnsure (16M+ in series A+ 2020)
  - Provide digital insurance for e-economic 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



# 課程綱要

- 課前知識
- 概念簡介
- 環境介紹
- 詞彙定義
- 範例練習
- 問與答

# 課前知識

- Be comfortable with UNIX command line
  - Navigating directories with `cd` or `tree`
  - Editing files, like `vim`, `nano`
  - Bash scripting, like env or looping
- Be an export with `Google`
  - <https://letmegoogletest.com/?q=you+can+google+it>
- It is totally OK if you don't know what is Container and Kubernetes

孩子，您多久沒唸中文了？

## 荀子《儒效篇》

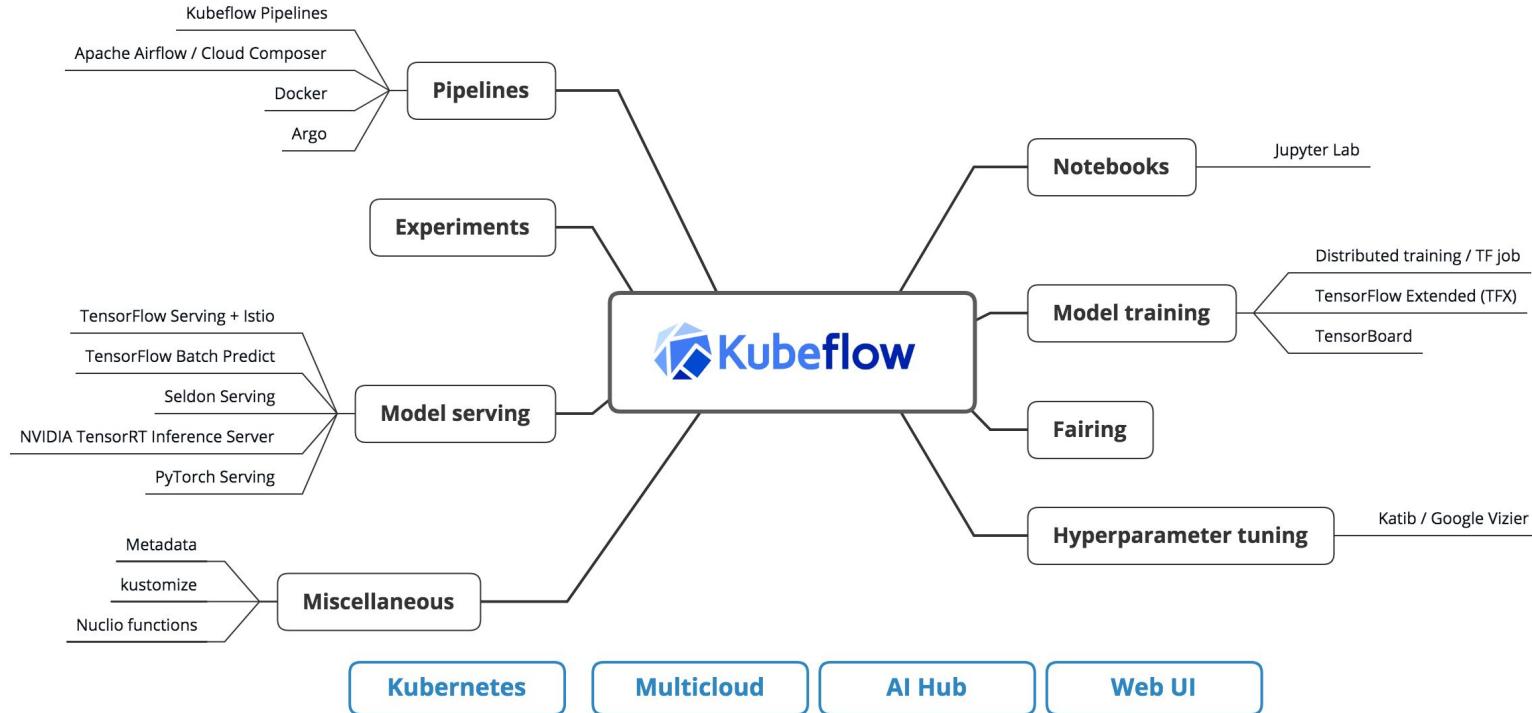
「不聞不若聞之，聞之不若見之，見之不若知之，知之不若行之；學至于行之而止矣。」

## 範例資源

```
git clone https://github.com/FootprintAI/kubeflow-workshop
```

Or [Click Me](#)

# Kubeflow架構



# Fairness and Explainable AI

# Why explainable AI? (1/2)

- Can we trust the model?
  - KDD 2008 Breast Cancer Identification Winner Report

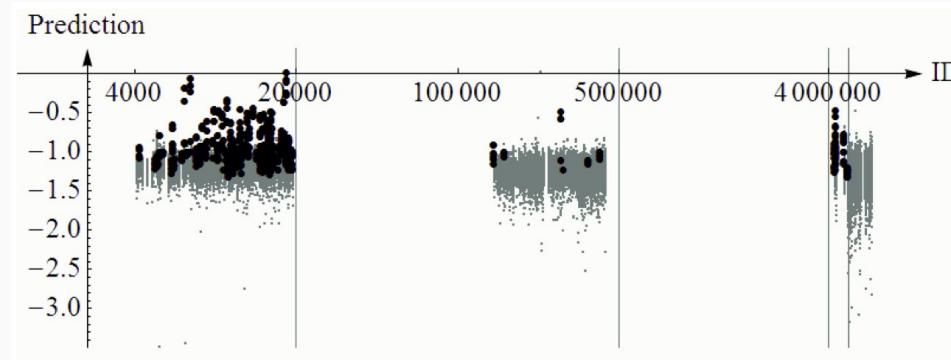


Figure 1: Distribution of malignant (black) and benign (gray) candidates depending on patient ID on the X-axis in log scale. The Y-axes is the score of a linear SVM model on the 117 features. Vertical lines show the boundaries of the identified ID bins.

## Why explainable AI? (2/2)

- 是「神之一手」還是「無法接受的錯誤」?
  - “Although during the match it was unclear why the system played this move, it was the deciding move for AlphaGo to win the game.” [1]

*“It’s not a human move. I’ve never seen a human play this move.”* (Fan Hui, 2016).

### Fan Hui

From Wikipedia, the free encyclopedia

**Fan Hui** (Chinese: 樊麾; pinyin: Fán Huī; born 27 December 1981) is a Chinese-born French Go player.<sup>[2]</sup> Becoming a professional Go player in 1996, Fan moved to France in 2000 and became the coach of the French national Go team in 2005.<sup>[3]</sup> He was the champion of the European Go Championship in 2013, 2014<sup>[4]</sup> and 2015.<sup>[5]</sup> As of 2015, he is ranked as a 2 dan professional.<sup>[5]</sup> He additionally won the 2016 European Professional Go Championship.<sup>[6]</sup>

[1] <https://arxiv.org/abs/1708.08296>

# What is explainable AI?

1. Verification of the system
  - A black box system is must not trust by default. For example, the use of models in health care should be interpreted and verified by medical experts in an absolute necessity.
2. Improvement of the system
  - If the model is explainable, it is easy to perform weakness analysis for better model improvement.
3. Learning from the system
  - AIs are trained with millions of examples while experts are trained with a limited number of examples. With explainable AI, we are able to acquire new insights from the distilled knowledge.
4. Compliance to legislation

# From GDPR perspective

## Art. 22 GDPR

### Automated individual decision-making, including profiling

1. The data subject shall have the right not to be subject to a decision based solely on automated processing, including profiling, which produces legal effects concerning him or her or similarly significantly affects him or her.
2. Paragraph 1 shall not apply if the decision:
  - (a) is necessary for entering into, or performance of, a contract between the data subject and a data controller;
  - (b) is authorised by Union or Member State law to which the controller is subject and which also lays down suitable measures to safeguard the data subject's rights and freedoms and legitimate interests; or
  - (c) is based on the data subject's explicit consent.
3. In the cases referred to in points (a) and (c) of paragraph 2, the data controller shall implement suitable measures to safeguard the data subject's rights and freedoms and legitimate interests, at least the right to obtain human intervention on the part of the controller, to express his or her point of view and to contest the decision.
4. Decisions referred to in paragraph 2 shall not be based on special categories of personal data referred to in Article 9(1), unless point (a) or (g) of Article 9(2) applies and suitable measures to safeguard the data subject's rights and freedoms and legitimate interests are in place.

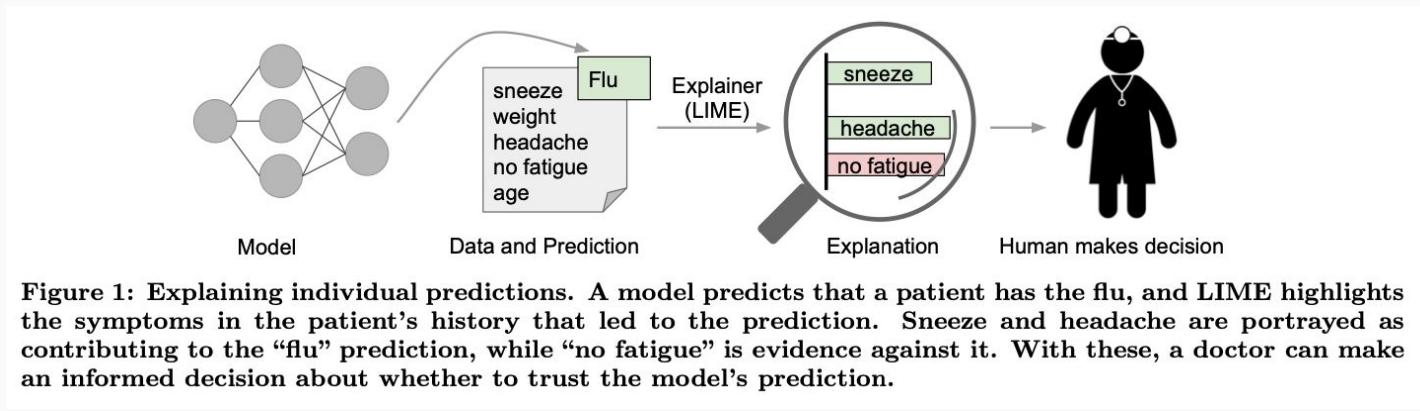
對個人有法律或重大影響的決定，不得基於(Article 9 規定的)個人種族、政治立場、宗教與哲學信仰、商業關係這些個人資料[2]

[1] <https://gdpr-info.eu/art-22-gdpr/>

[2] [https://medium.com/trustableai/%E5%8D%80%E5%85%A8-gdpr-%E6%97%89%E5%80%BC%E5%AF%86%E5%88%86%E5%8A%A0%E5%8A%9B%E5%95%86d153de909e4f](https://medium.com/trustableai/%E5%8D%80%E5%85%A8-gdpr-%E6%97%89%E5%80%BC%E5%AF%86%E5%88%86%E5%8A%A8%E5%8A%A0%E5%8A%9B%E5%95%86d153de909e4f)

# Explainable AI Examples (1/3)

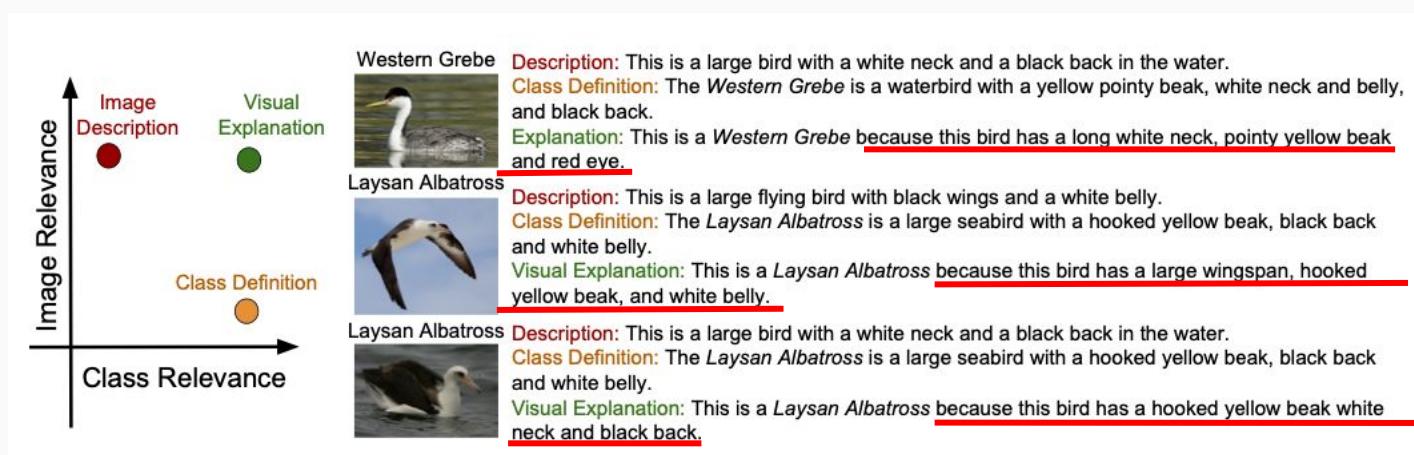
- Highlight the symptoms that led to the prediction.



**Figure 1: Explaining individual predictions.** A model predicts that a patient has the flu, and LIME highlights the symptoms in the patient's history that led to the prediction. Sneeze and headache are portrayed as contributing to the "flu" prediction, while "no fatigue" is evidence against it. With these, a doctor can make an informed decision about whether to trust the model's prediction.

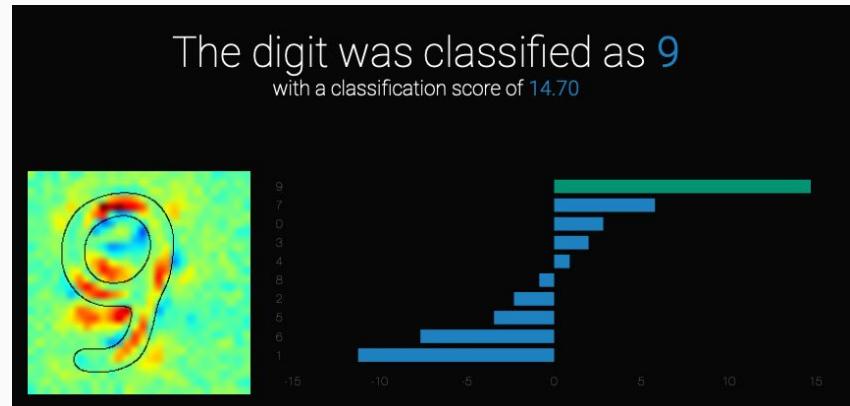
# Explainable AI Examples (2/3)

- Generating Visual Explanations



## Explainable AI Examples (3/3)

- Heatmap to support the decision in digital recognition.



Ref: <http://heatmapping.org>

# What is Fairness?

Fairness has different definitions across disciplines too. This is captured in a paper by Deirdre Mulligan, Joshua Kroll, Nitin Kohli and Richmond Wong.<sup>2</sup> For example:



**Law:** fairness includes protecting individuals and groups from discrimination or mistreatment with a focus on prohibiting behaviors, biases and basing decisions on certain protected factors or social group categories.



**Social science:** "often considers fairness in light of social relationships, power dynamics, institutions and markets."<sup>3</sup> Members of certain groups (or identities) that tend to experience advantages.



**Quantitative fields** (i.e. math, computer science, statistics, economics): questions of fairness are seen as mathematical problems. Fairness tends to match to some sort of criteria, such as equal or equitable allocation, representation, or error rates, for a particular task or problem.



**Philosophy:** ideas of fairness "rest on a sense that what is fair is also what is morally right."<sup>4</sup> Political philosophy connects fairness to notions of justice and equity.

# What is fairness AI?

## 1. Fairness of a system

- A particular problem is fit to be approximated or decided upon by an ML system

## 2. Fairness of a result

- do the outputs of the algorithm actually correspond to what would constitute a fair response by a human.

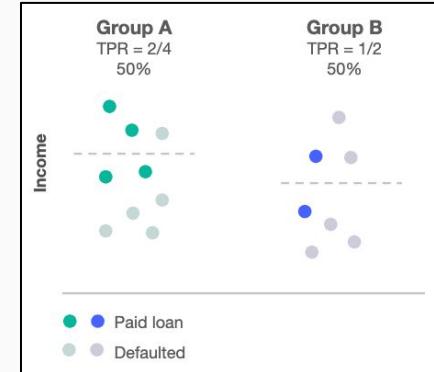
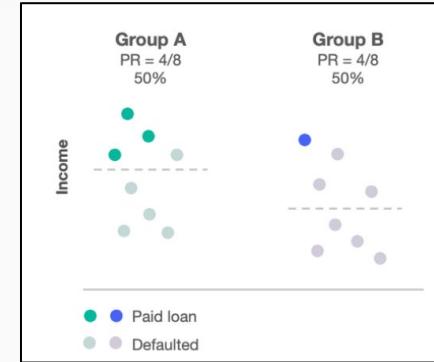
## 3. Fairness of an approach

- White box models
  - i. Approaching fairness by developing training methods that aim to produce interpretable ML models.
  - ii. Limitations:
    1. Not feasible for complex models (e.g. deep learning models)
    2. Fairness is bound by the ability for a subject to meaningfully understand.
- Mathematical Fairness

# What is fairness AI?

- Mathematical Fairness

- Demographic parity (top)
  - The proportion of each segment of a protected class (e.g. gender) should receive the positive outcome at equal rates.
  - Best for university admission.
- Individual fairness / Group fairness
  - Similar individual (or groups) are treated similarly
- Equal opportunity (bottom)
  - Each group should get the positive outcome at equal rates.
  - Best for fraud detection

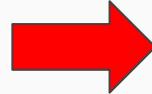


- Interactive website

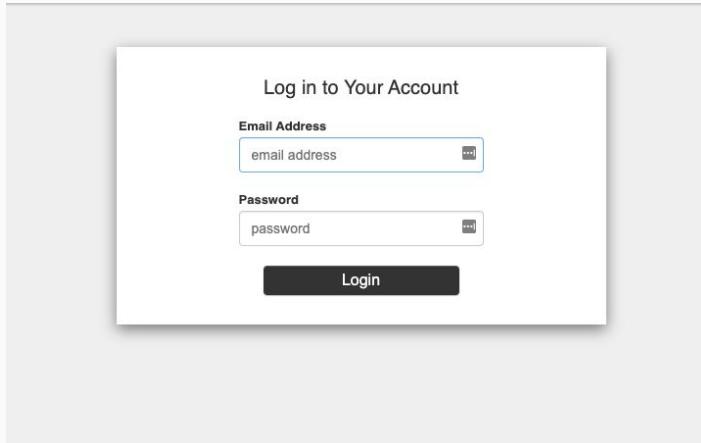
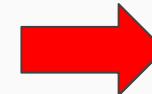
# 環境介紹

# 虛擬機環境

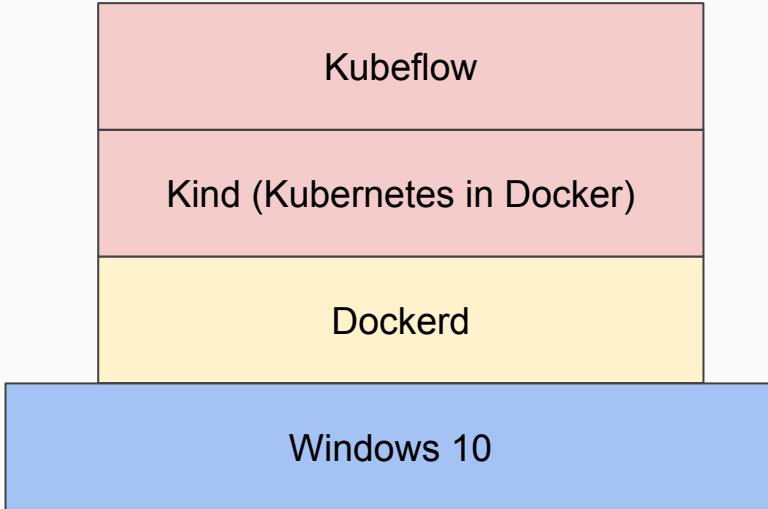
1.kubectl port-forward  
svc/istio-ingressgateway -n istio-system  
8080:80 --address 0.0.0.0



2.Open <http://localhost:8080>



A screenshot of a web browser showing a login form. The title bar says "Log in to Your Account". The form has two input fields: "Email Address" with placeholder "email address" and "Password" with placeholder "password". Below the fields is a black "Login" button.



Wait! 所以我說那個帳號密碼呢?

Account: user@example.com

Password: 12341234

# Kubectl Cli

```
// 查看所有namespace  
Kubectl get namespaces
```

```
// 查看kubeflow中所有運行的Pod  
kubectl get pods -n kubeflow
```

```
// 查看目前使用者運行的Pod  
kubectl get pods -n kubeflow-user-example-com
```

# Step1: 開啟Notebook作為開發環境 (1/3)

The screenshot shows the Kubeflow web interface. On the left is a dark sidebar with various navigation options: Home, Notebooks (which is selected and highlighted with a red box), Tensorboards, Volumes, Models, Experiments (AutoML), Experiments (KFP), Pipelines, Runs, Recurring Runs, Artifacts, and Executions. At the bottom of the sidebar are links for Privacy and Usage Reporting, and a note about the build version being dev\_local.

The main content area is titled "Notebooks". It features a table header with columns: Status, Name, Type, Age, Image, GPUs, CPUs, Memory, and Volumes. To the right of the table, there is a large red box highlighting a blue button labeled "+ NEW NOTEBOOK". A large red arrow points upwards from the bottom of the screen towards this button. Below the table, the text "建立Notebook" is displayed in red.

# Step1: 開啟Notebook作為開發環境 (2/3)

Kubeflow

kubeflow-user-example-c...

Specify the name of the Notebook Server and the Namespace it will belong to.

Name: demo

Namespace: kubeflow-user-example-com

Image

A starter Jupyter Docker Image with a baseline deployment of ML packages

Custom Image

jupyterlab 1 2

Image: j1r0q0g6/notebooks/notebook-servers/jupyter-tensorflow-full:v1.4

Advanced Options

CPU / RAM

Specify the total amount of CPU and RAM reserved by your Notebook Server. For CPU-intensive workloads, you can choose more than 1 CPU (e.g. 1.5).

Requested CPU: 0.5

Requested Memory: 1

Advanced Options

給定名稱以及指定其CPU/Memory

# Step1: 開啟Notebook作為開發環境 (3/3)

Kubeflow

kubeflow-user-example-c...

Home

Notebooks

Tensorboards

Volumes

Models

Experiments (AutoML)

Experiments (KFP)

Pipelines

Runs

Recurring Runs

Artifacts

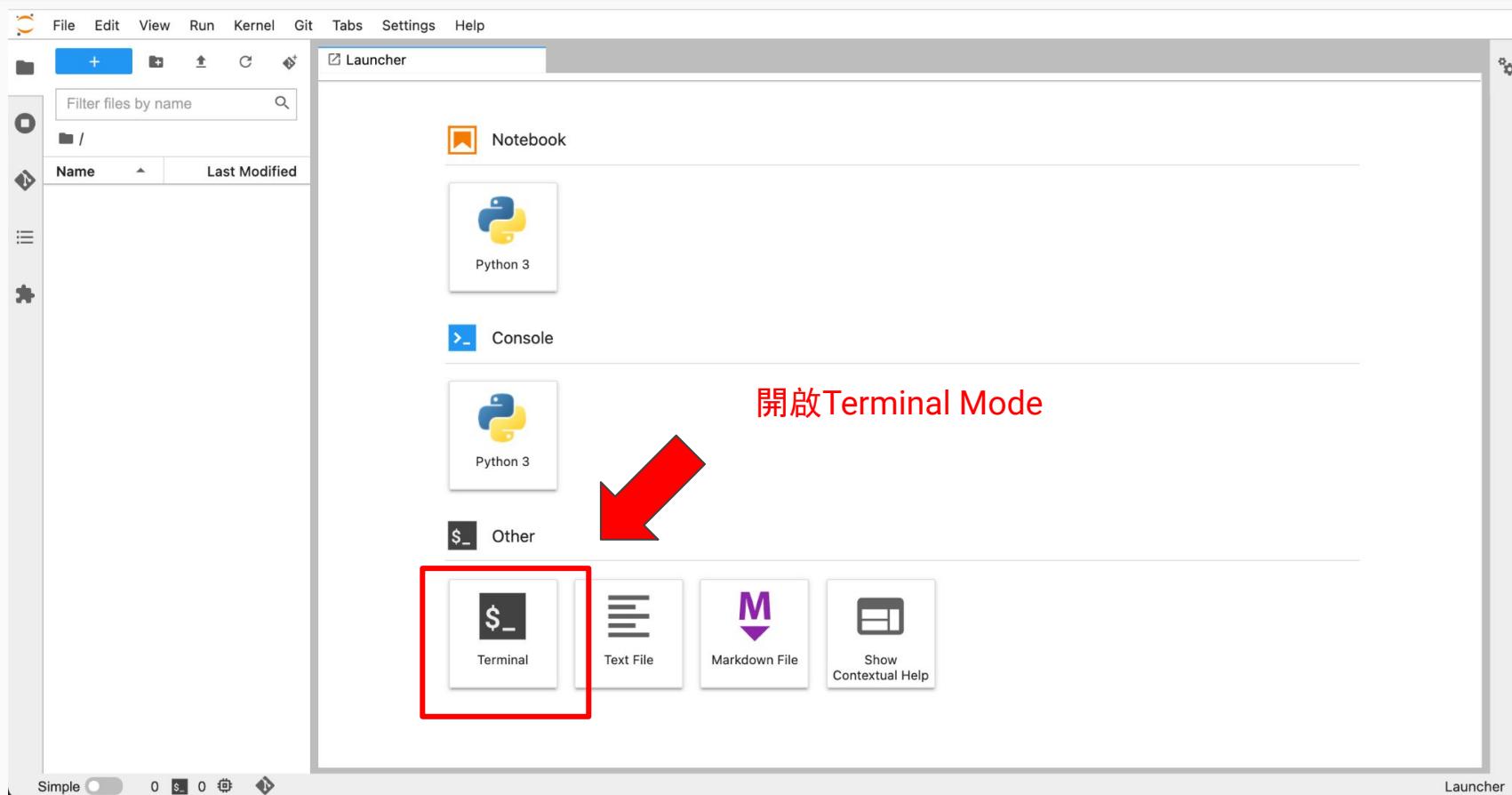
Notebooks

+ NEW NOTEBOOK

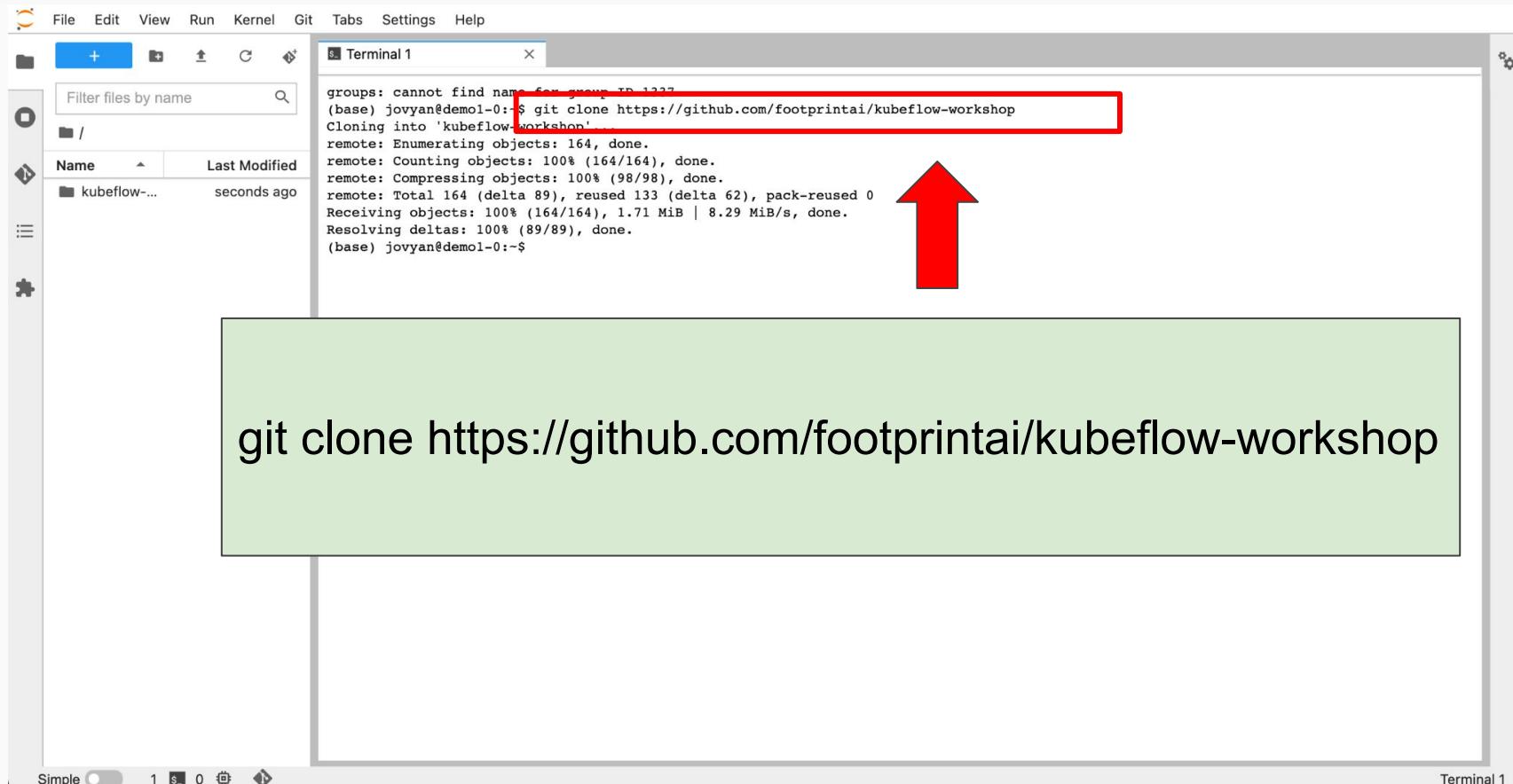
Status	Name	Type	Age	Image	GPUs	CPUs	Memory	Volumes	⋮	CONNECT	⋮	⋮
✓	demo1	jupyter	20 hours ago	jupyter-scipy:v1.4	0	0.5	1Gi		⋮	CONNECT	⋮	⋮
✓	demo2	jupyter	2 hours ago	jupyter-tensorflow-full:v1.4	0	0.5	1Gi		⋮	CONNECT	⋮	⋮

連線Notebook

## Step2: 開啟Terminal下載範例程式(1/3)



## Step2: 開啟Terminal下載範例程式(2/3)

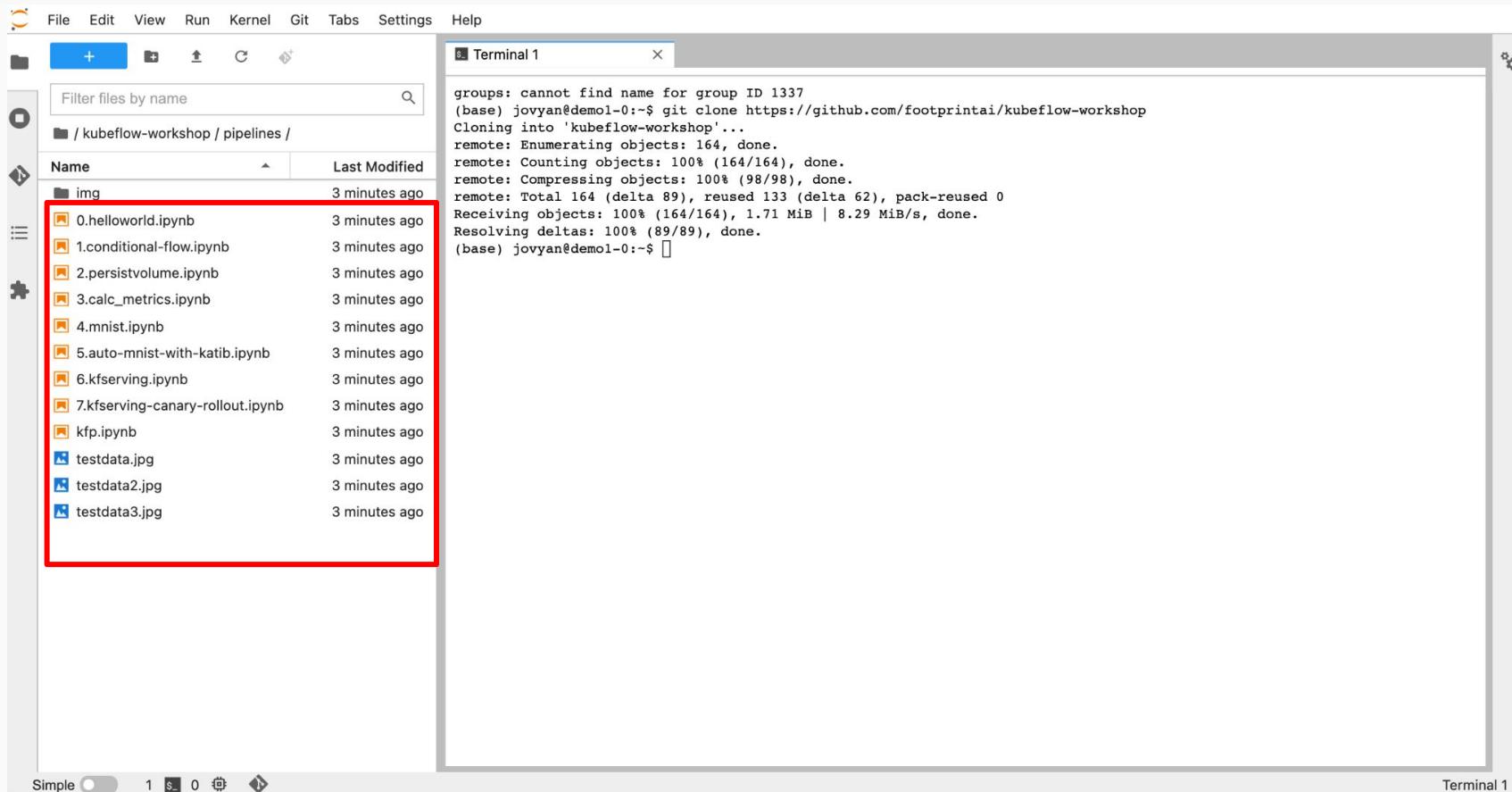


The screenshot shows the Jupyter Notebook interface. On the left is a file browser sidebar. In the center is a terminal window titled "Terminal 1". A red box highlights the command "git clone https://github.com/footprintai/kubeflow-workshop". A large red arrow points upwards from a green box containing the same command towards the terminal window.

```
groups: cannot find name for group ID 1222
(base) jovyan@demo1-0:~$ git clone https://github.com/footprintai/kubeflow-workshop
Cloning into 'kubeflow-workshop'...
remote: Enumerating objects: 164, done.
remote: Counting objects: 100% (164/164), done.
remote: Compressing objects: 100% (98/98), done.
remote: Total 164 (delta 89), reused 133 (delta 62), pack-reused 0
Receiving objects: 100% (164/164), 1.71 MiB | 8.29 MiB/s, done.
Resolving deltas: 100% (89/89), done.
(base) jovyan@demo1-0:~$
```

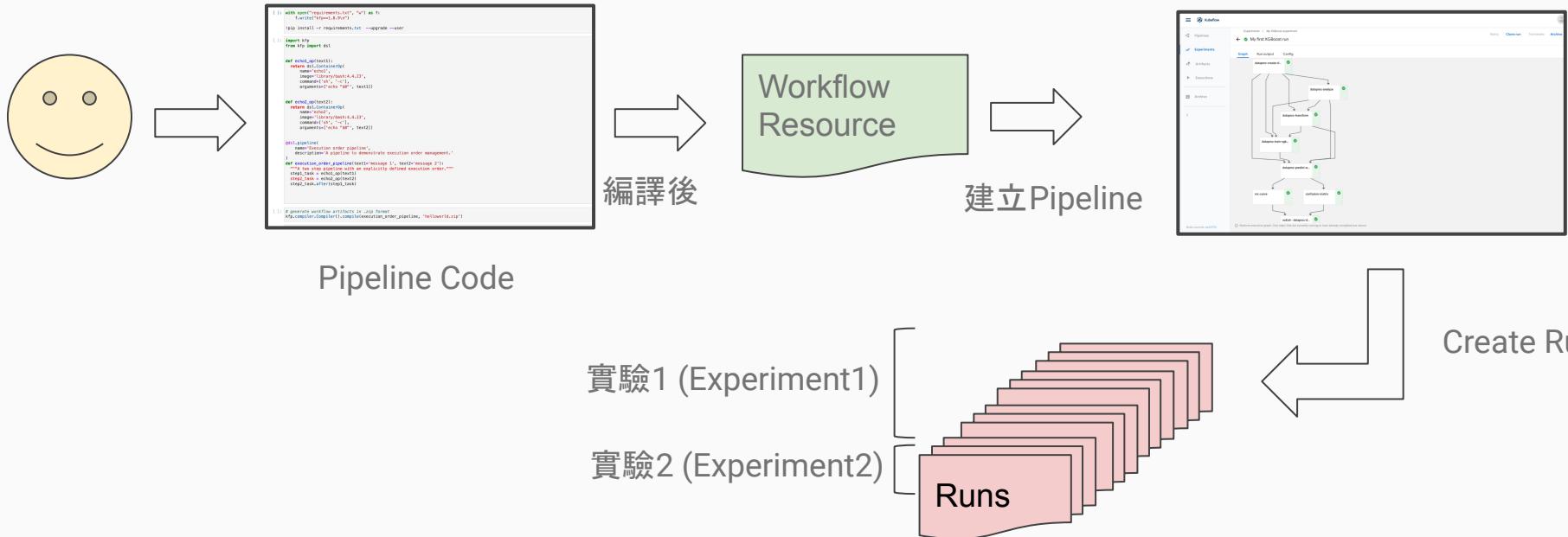
git clone https://github.com/footprintai/kubeflow-workshop

## Step2: 開啟Terminal下載範例程式(3/3)



# 詞彙說明

# 詞彙說明



# 範例1 Hello World!

## Step3: 編譯helloworld.ipynb (1/2)

The screenshot shows a Jupyter Notebook environment with the following details:

- File Bar:** File, Edit, View, Run, Kernel, Git, Tabs, Settings, Help.
- Left Sidebar:** Shows a file tree under the path `/kubeflow-workshop / pipelines /`. The current file selected is `0.helloworld.ipynb`.
- Terminal 1:** A terminal window titled `0.helloworld.ipynb` running Python 3. It displays the following code:

```
[ ]: with open("requirements.txt", "w") as f:  
    f.write("kfp==1.8.9\n")  
  
[ ]: !pip install -r requirements.txt --upgrade --user  
  
[ ]: import kfp  
from kfp import dsl  
  
def echo1_op(text1):  
    return dsl.ContainerOp(  
        name='echo1',  
        image='library/bash:4.4.23',  
        command=['sh', '-c'],  
        arguments=['echo "$0"', text1])  
  
def echo2_op(text2):  
    return dsl.ContainerOp(  
        name='echo2',  
        image='library/bash:4.4.23',  
        command=['sh', '-c'],  
        arguments=['echo "$0"', text2])  
  
@dsl.pipeline(  
    name='Execution order pipeline',  
    description='A pipeline to demonstrate execution order management.')  
def execution_order_pipeline(text1='message 1', text2='message 2'):  
    """A two step pipeline with an explicitly defined execution order."""  
    step1_task = echo1_op(text1)  
    step2_task = echo2_op(text2)  
    step2_task.after(step1_task)  
  
[ ]: # generate workflow artifacts in .zip format  
kfp.compiler.Compiler().compile(execution_order_pipeline, 'helloworld.zip')
```

A red box highlights the line `kfp.compiler.Compiler().compile(execution_order_pipeline, 'helloworld.zip')`.

**Bottom Status Bar:** Simple, Python 3 | Idle, Mode: Command, Ln 1, Col 1, 0.helloworld.ipynb.

## Step3: 編譯helloworld.ipynb (2/2)

The screenshot shows a Jupyter Notebook interface. On the left, the file browser displays a list of files and folders in the directory `/kubeflow-workshop/pipelines`. A red arrow points upwards from the bottom of the file list towards the terminal window. The terminal window (Terminal 1) shows Python code defining a pipeline and its components. The code includes definitions for `echo1_op`, `echo2_op`, and an `@dsl.pipeline` block named `Execution order pipeline`. It also defines a step function `execution_order_pipeline` that runs two steps sequentially. At the bottom of the terminal, a warning message is displayed about the `ContainerOp` class.

```
name='echo1',
image='library/bash:4.4.23',
command=['sh', '-c'],
arguments=['echo "$0"', text1]

def echo2_op(text2):
    return dsl.ContainerOp(
        name='echo2',
        image='library/bash:4.4.23',
        command=['sh', '-c'],
        arguments=['echo "$0"', text2])

@dsl.pipeline(
    name='Execution order pipeline',
    description='A pipeline to demonstrate execution order management.')
def execution_order_pipeline(text1='message 1', text2='message 2'):
    """A two step pipeline with an explicitly defined execution order."""
    step1_task = echo1_op(text1)
    step2_task = echo2_op(text2)
    step2_task.after(step1_task)

[3]: # generate workflow artifacts in .zip format
kfp.compiler.Compiler().compile(execution_order_pipeline, 'helloworld.zip')

/opt/conda/lib/python3.8/site-packages/kfp/dsl/_container_op.py:1150: FutureWarning: Please create reusable components instead of constructing ContainerOp instances directly. Reusable components are shareable, portable and have compatibility and support guarantees. Please see the documentation on: https://www.kubeflow.org/docs/pipelines/sdk/component-development/#writing-your-component-definition-file The components can be created manually (or, in case of python, using kfp.components.create_component_from_func or func_to_container_op) and then loaded using kfp.components.load_component_from_file, load_component_from_uri or load_component_from_text: https://kubeflow-pipelines.readthedocs.io/en/stable/source/kfp.components.html#kfp.components.load\_component\_from\_file
warnings.warn(
```

生成編譯檔  
helloworld.zip

## Step4: 建立Pipeline (1/7)

The screenshot shows the Kubeflow Pipelines interface. On the left, a sidebar menu lists various options: Home, Notebooks, Tensorboards, Volumes, Models, Experiments (AutoML), Experiments (KFP), Pipelines (highlighted with a red box), Runs, Recurring Runs, Artifacts, and Executions. At the bottom of the sidebar is a 'Manage Contributors' link. The main area is titled 'Pipelines' and contains a table of existing pipelines. The table has columns for Pipeline name, Description, and Last modified. A red box highlights the 'Upload pipeline' button in the top right corner of the main area. A large red arrow points upwards towards this button. The table data is as follows:

Pipeline name	Description	Last modified
[Tutorial] V2 lightweight Python com...	source code Shows different component input and output options for KFP v2 components.	11/30/2021, 1:02:25 PM
[Tutorial] DSL - Control structures	source code Shows how to use conditional execution and exit handlers. This pipeline will randomly fail to demonstr...	11/30/2021, 1:02:24 PM
[Tutorial] Data passing in python co...	source code Shows how to pass data between python components.	11/30/2021, 1:02:23 PM
[Demo] TFX - Taxi tip prediction mod...	source code GCP Permission requirements. Example pipeline that does classification with model analysis based on...	11/30/2021, 1:02:22 PM
[Demo] XGBoost - Iterative model tra...	source code This sample demonstrates iterative training using a train-eval-check recursive loop. The main pipeline ...	11/30/2021, 1:02:21 PM

Rows per page: 10 < >

## Step4: 建立Pipeline (2/7)

The screenshot shows the Kubeflow Pipeline creation interface. On the left is a dark sidebar with navigation links: Home, Notebooks, Tensorboards, Volumes, Models, Experiments (AutoML), Experiments (KFP), Pipelines, Runs, Recurring Runs, Artifacts, Executions, Manage Contributors, Create, and Cancel.

The main area is titled "Upload Pipeline or Pipeline Version". It has two radio button options: "Create a new pipeline" (selected) and "Create a new pipeline version under an existing pipeline".

Below this, it says "Upload pipeline with the specified package." and shows a "Pipeline Name" field containing "0.helloworld" (highlighted by a red box). A large red arrow points to this field with the text "1.輸入Pipeline名稱".

Further down, there's a "Pipeline Description" field containing "0.helloworld".

Next, it says "Choose a pipeline package file from your computer, and give the pipeline a unique name. You can also drag and drop the file here." Below this is a "File\*" input field containing "helloworld (19).zip" (highlighted by a red box). A large red arrow points to this field with the text "2.指定編譯完後Zip位置".

At the bottom, there are "Import by url" and "Package Url" fields, and a "Code Source (optional)" field.

At the very bottom, there are "Create" and "Cancel" buttons (the "Create" button is highlighted by a red box). A large red arrow points to these buttons with the text "3.建立".

## Step4: 建立Pipeline (3/7)

The screenshot shows the Kubeflow interface for creating a new experiment. On the left, a sidebar lists various options: Home, Notebooks, Tensorboards, Volumes, Models, Experiments (AutoML), Experiments (KFP), Pipelines, Runs, Recurring Runs, Artifacts, and Executions. Below these are Manage Contributors and a GitHub integration section.

The main area is titled "Pipelines" and shows a pipeline named "0.helloworld (0.helloworld)". It displays a "Graph" view with two nodes: "echo1" at the top and "echo2" below it, connected by a downward arrow. There is also a "YAML" tab. A red box highlights the "+ Create experiment" button in the top right corner. A large red arrow points upwards from the bottom right towards this button. A red text overlay on the right side reads "建立實驗以便管理 Run".

**Summary** (Details shown in the bottom-left box):

- ID: 6f25028f-01e3-4acd-9389-7ec2031fb04b
- Version: 0.helloworld
- Version source

## Step4: 建立Pipeline (4/7)

The screenshot shows the Kubeflow interface for creating a new experiment. On the left is a dark sidebar with navigation links: Home, Notebooks, Tensorboards, Volumes, Models, Experiments (AutoML), Experiments (KFP) (which is expanded to show Pipelines, Runs, Recurring Runs, Artifacts, and Executions), Manage Contributors, and GitHub integration.

The main area is titled "Experiments" and "← New experiment". It contains "Experiment details" with a placeholder text: "Think of an Experiment as a space that contains the history of all pipelines and their associated runs".

A red box highlights the input field "0.helloworld.exp" under "Experiment name". A large red arrow points from this field towards the "Next" button at the bottom left. Another red box highlights the "Next" button itself. To its right is the "Cancel" button.

On the right side of the main area, there is a large red text overlay: "輸入實驗名稱並建立實驗資源" (Input experiment name and create experiment resources).

At the bottom of the page, there are links for "Privacy • Usage Reporting" and "build version dev\_local".

## Step4: 建立Pipeline (5/7)

The screenshot shows the Kubeflow UI for creating a pipeline run. The left sidebar includes options like Home, Notebooks, Tensorboards, Volumes, Models, Experiments (AutoML), Experiments (KFP) (selected), Pipelines, Runs, Recurring Runs, Artifacts, and Executions. The main area is titled 'Run details' and contains the following fields:

- Pipeline\***: A dropdown menu showing '0.helloworld' with a 'Choose' button.
- Pipeline Version\***: A dropdown menu showing '0.helloworld' with a 'Choose' button.
- Run name\***: An input field containing 'Run of 0.helloworld (1e261)'.
- Description (optional)**: A text input field.
- This run will be associated with the following experiment**: A section with a dropdown menu showing '0.helloworld.exp' with a 'Choose' button.
- Service Account (Optional)**: An input field.
- Run Type**: Radio buttons for 'One-off' (selected) and 'Recurring'.
- Run parameters**: A section for specifying pipeline parameters:
  - text1**: Input field containing 'message 1'.
  - text2**: Input field containing 'message 2'.
- Start**: A blue button at the bottom left, highlighted with a red border.
- Skip this step**: A link at the bottom left.

Two large red arrows point from the right towards the 'Pipeline' and 'Experiment' sections, indicating the steps described in the text.

1. 運行此 Pipeline 成為 Run 資源

2. 指定其歸屬的實驗資源

## Step4: 建立Pipeline (6/7)

The screenshot shows the Kubeflow interface for managing experiments. On the left, a sidebar navigation bar includes links for Home, Notebooks, Tensorboards, Volumes, Models, Experiments (AutoML), Experiments (KFP), Pipelines, Runs (selected), Recurring Runs, Artifacts, and Executions. Below these are Manage Contributors, GitHub integration, and Documentation links.

The main content area is titled "Experiments" and shows the details for "0.helloworld.exp". It includes sections for "Recurring run configs" (0 active) and "Experiment description".

The "Runs" section displays a table of active runs. The columns are: Run name, Status, Duration, Pipeline Version, Recurring Run, and Start time. A single row is visible, labeled "Run of 0.helloworld (1e261)". This row is highlighted with a red border. A large red arrow points upwards from the bottom of the page towards this highlighted row.

Run name	Status	Duration	Pipeline Version	Recurring Run	Start time
Run of 0.helloworld (1e261)	?	-	0.helloworld	-	12/2/2021, 4:33:10 PM

At the bottom right, there are buttons for "Create run" and "Create recurring run", along with links for Compare runs, Clone run, and Archive.

**Run清單列表**

# Step4: 建立Pipeline (7/7)

The screenshot shows the Kubeflow interface for managing machine learning pipelines. On the left, a sidebar lists various resources like Home, Notebooks, Tensorboards, Volumes, Models, Experiments (AutoML), Experiments (KFP), Pipelines, Runs, Recurring Runs, Artifacts, and Executions. Below these are Manage Contributors and GitHub integration.

The main area displays a pipeline run titled "Run of 0.helloworld (1e261)". It includes tabs for Graph, Run output, and Config. The Graph tab shows a simple flow from "echo1" to "echo2". A red box highlights this graph area.

A modal window provides detailed information about the execution:

- Input/Output** tab is selected, showing:
  - Input parameters**: text1 (message 1)
  - Input artifacts**: None
  - Output parameters**: None
  - Output artifacts**: main-logs (minio://mipeline/artifacts/execution-order-pipeline-qvlt5/2021/11/30/execution-order-pipeline-qvlt5-137025724/main.log, message 1)
- Visualizations**, **Details**, **Volumes**, **Logs**, **Pod**, **Events**, and **ML Metadata** tabs are also present.

A large red arrow points upwards from the "Output artifacts" section towards the right, with the text "運行結果輸出" (Execution Result Output) written next to it.

# 範例2 Fairness Study with What-If Tool and Tensorboard

## Step5: 建立fairness pipeline (1/3)

The screenshot shows a Jupyter Notebook interface with a terminal window open. The terminal window is titled "Terminal 1" and has a tab labeled "fairness-study-with-tensorboard". The code in the terminal is as follows:

```
[ ]: ### Study ML Model Fairness with Whatif tool

# In this study, we use kubeflow pipeline to construct:
# 1. persistent volume claim for holding data from https://storage.googleapis.com/what-if-tool-resources/uci-census-demo/uci-census-demo.zip
# 2. create an ETL job to convert the downloaded zip file
# 3. launch a kserve for holding pretrain models for later inference
# 4. launch tensorflow and navigate to what if tool
#   for tensorflow issue: please refer to https://github.com/tensorflow/tensorboard/issues/5472
#   for RBAC issue: please apply allow-all authorization policy under hack folder.
#
# for more details on each step, please refers to our slide section: https://github.com/FootprintAI/kubeflow-workshop/tree/main/slides
#
# reference: https://pair-code.github.io/what-if-tool/learn/tutorials/walkthrough/

[ ]: with open("requirements.txt", "w") as f:
    f.write("kubernetes>=12.0.0\n")
    f.write("kfp==1.8.9\n")
    f.write("requests\n")

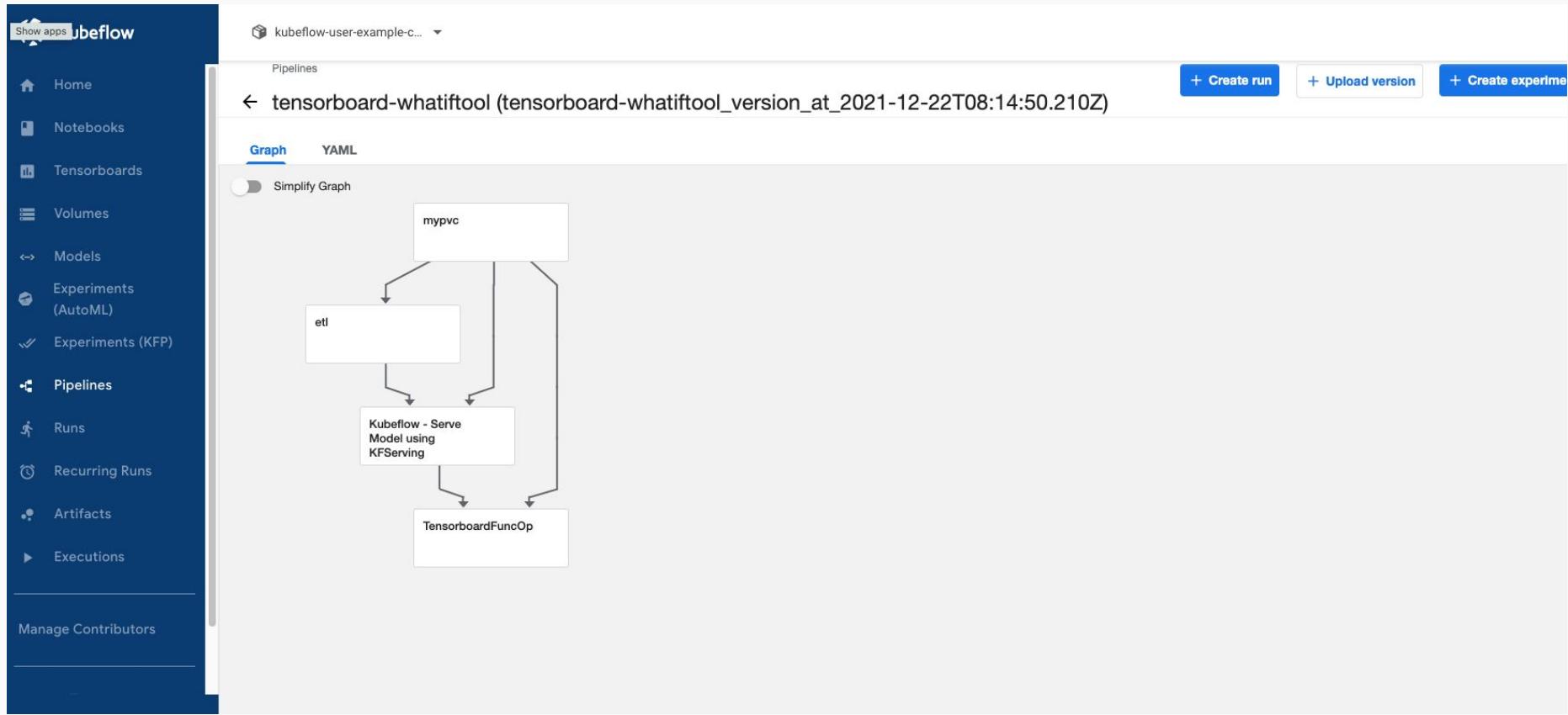
!pip install -r requirements.txt --upgrade --user

[ ]: import json
import kfp.dsl as dsl
import kfp
from kfp import components
from kfp.components import func_to_container_op
from typing import NamedTuple

kfserving_op = components.load_component_from_url('https://raw.githubusercontent.com/kubeflow/pipelines/master/components/kubeflow/kfserver')

@func_to_container_op
def tensorflow_func_op() -> NamedTuple('Outputs', [('mlpipeline_ui_metadata', 'UI_metadata')]):
    metadata = {
        'outputs' : [
            {
                'type': 'tensorboard',
                'source': 'gs://what-if-tool-resources/uci-census-demo/uci-census-demo.zip',
            }
        ]
    }
    import json
    return (json.dumps(metadata))
```

## Step5: 建立fairness pipeline (2/3)



# Step5: 建立fairness pipeline (3/3)

Kubeflow

kubeflow-user-example-c... ▾

Experiments > uci

Retry Clone run Terminate Archive

Graph Run output Config

Simplify Graph

mypvc

etl

Kubflow - Serve Model using KFServing

Tensorboard func op

Input/Output Visualizations Details Volumes Logs Pod Events ML Metadata

Tensorboard

TF Image

TensorFlow 2.2.2

Start Tensorboard

Visualization Creator

create visualizations manually

Add visualizations to your own components following instructions in [Visualize Results in the Pipelines UI](#).

Runtime execution graph. Only steps that are currently running or have already completed are shown.

## Step6: hack tensorboard (1/3)

(此步為進階功能，若無法實作也無妨 )

為了讓tensorboard可以查看本地端的資料，我們得將其對應資源建立起：

1. 先尋找pvc的位置以及viewer id
2. 修改至hack/tensorboard-use-local-volume.yaml底下
3. 刪掉舊有的viewer resource (hack/delete-all-viewer.sh)

```
root@instance-x:/home/hsinloyeh# kubectl delete viewers -n kubeflow-user-example-com --all  
viewer.kubeflow.org "viewer-a4e50b4f89b636c7e2d12bebd3da533901b3ea8a" deleted
```

4. 然後kubectl apply -f hack/tensorboard-use-local-volume.yaml

```
root@instance-x:/home/hsinloyeh# kubectl get viewer -n kubeflow-user-example-com  
NAME          AGE  
viewer-a4e50b4f89b636c7e2d12bebd3da533901b3ea8a  13s
```

```
root@instance-1:~# kubectl get pvc -n kubeflow-user-example-com  
NAME          STATUS    VOLUME                                     CAPACITY  ACCESS MODES  STORAGECLASS  AGE  
volumeop-sequential-mv8sz-newpvc Bound     pvc-17431607-83bd-44f3-abe9-79b0108ae1cc  1Gi      RWO         standard      2d21h  
workspace-demo1 Bound     pvc-cf2d97d1-0202-4e2e-bafe-f398562627e2  5Gi      RWO         standard      19h  
workspace-demo2 Bound     pvc-f1489494-999e-461c-8fe0-96654b625104  5Gi      RWO         standard      58m  
workspace-tester1 Bound     pvc-c5fa720d-3ae7-4a2e-95d7-7edc4f668d13  5Gi      RWO         standard      2d22h
```

## Step6: hack tensorboard (2/3)

(此步為進階功能，若無法實作也無妨)

~~修改tensorboard deployment launch script~~

~~1. 先找到deployment的名字 ( kubectl get deployment -n kubeflow user example com )~~

```
root@instance-x:/home/hsinhyeh# kubectl get deployment -n kubeflow-user-example-com
NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
ml-pipeline-ui-artifact        1/1     1           1           2d2h
ml-pipeline-visualizationserver 1/1     1           1           2d2h
uci-census-predictor-default-00001-deployment 1/1     1           1           66m
viewer-a4e50b4f89b636c7e2d12bebd3da533901b3ea8a-deployment 1/1     1           1           2m11s
```

~~2. 修改deployment ( kubectl edit deployment <viewer id> -n kubeflow user example com )~~

```
spec:
  containers:
    - args:
        - tensorboard
        - --logdir=/data
        - - path_prefix /tensorboard/viewer_a4e50b4f89b636c7e2d12bebd3da533901b3ea8a/
        - --bind_all
```

~~3. 然後存檔~~

## Step6: hack tensorboard (3/3)

(此步為進階功能，若無法實作也無妨 )

1. 修改authorization policy, 改成不檢查權限 ( kubectl apply -f hack/allow-all-authorization-policy.yaml )
2. 連線到Tensorboard via port forward

1. 取得svc名稱 ( kubectl get svc -n kubeflow user example com)

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
ml-pipeline-ui-artifact	ClusterIP	10.96.108.10	<none>	80/TCP	2d3h
ml-pipeline-visualizationserver	ClusterIP	10.96.42.220	<none>	8888/TCP	2d3h
test1	ClusterIP	10.96.224.201	<none>	80/TCP	29h
uci-census	ExternalName	<none>	knative-local-gateway.istio-system.svc.cluster.local	<none>	72m
uci-census-predictor-default	ExternalName	<none>	knative-local-gateway.istio-system.svc.cluster.local	80/TCP	72m
uci-census-predictor-default-00001	ClusterIP	10.96.171.220	<none>	81/TCP	72m
uci-census-predictor-default-00001-private	ClusterIP	10.96.8.198	<none>	80/TCP,9090/TCP,9091/TCP,8022/TCP	72m
viewer-a4e50b4f89b636c7e2d12bebd3da533901b3ea8a-service	ClusterIP	10.96.29.148	<none>	80/TCP	7m50s

2. kubectl port forward

```
svc/viewer-a4e50b4f89b636c7e2d12bebd3da533901b3ea8a-service 8091:80  
--address=0.0.0.0 -n kubeflow user example com
```

3. Open <http://localhost:8091>

## Step7: configure what-if tool with tensorboard (1/3)

The screenshot shows the TensorBoard interface. At the top, there is an orange header bar with the 'TensorBoard' logo on the left and several icons on the right: 'UPLOAD', a gear icon, a circular arrow icon, a settings icon, and a help icon. Below the header, a large white area displays the message 'No dashboards are active for the current data set.' In the center, under 'Probable causes:', there is a bulleted list: '• You haven't written any data to your event files.' and '• TensorBoard can't find your event files.' To the right of this list is a vertical sidebar with several options: 'PROFILE', 'HPARAMS', 'MESH', 'TIME SERIES', 'PROJECTOR', and 'WHAT-IF TOOL'. The 'WHAT-IF TOOL' option is highlighted with a thick red rectangular border. At the bottom of the page, there is some smaller text: 'If you're new to using TensorBoard, and want to find out how to add data and set up your event files, check out the [README](#) and perhaps the [TensorBoard tutorial](#). If you think TensorBoard is configured properly, please see [the section of the README devoted to missing data problems](#) and consider filing an issue on GitHub.' followed by 'Last reload: Dec 22, 2021, 4:34:40 PM' and 'Log directory: ./data'.

## Step7: configure what-if tool with tensorboard (2/3)

### Set up your data and model

Inference address  
uci-census-predictor-default.kubeflow-user-example-com.svc.cluster.local:80

Model name  
**uci-census**

Model version (optional)  
1

Model signature (optional)

ADD ANOTHER MODEL FOR COMPARISON

Model Type  
 Classification    Regression    Uses Predict API

Path to examples  
/data/uci\_census/adult.tfrecord

SequenceExamples

Maximum number of examples to load  
1000

Sampling ratio (0.2 = sample ~20% of examples)  
1

Path to label dictionary (optional)

Maps predicted class indices to labels from text file

Max classes to display  
5

Multi-class classification model

Cancel   Accept

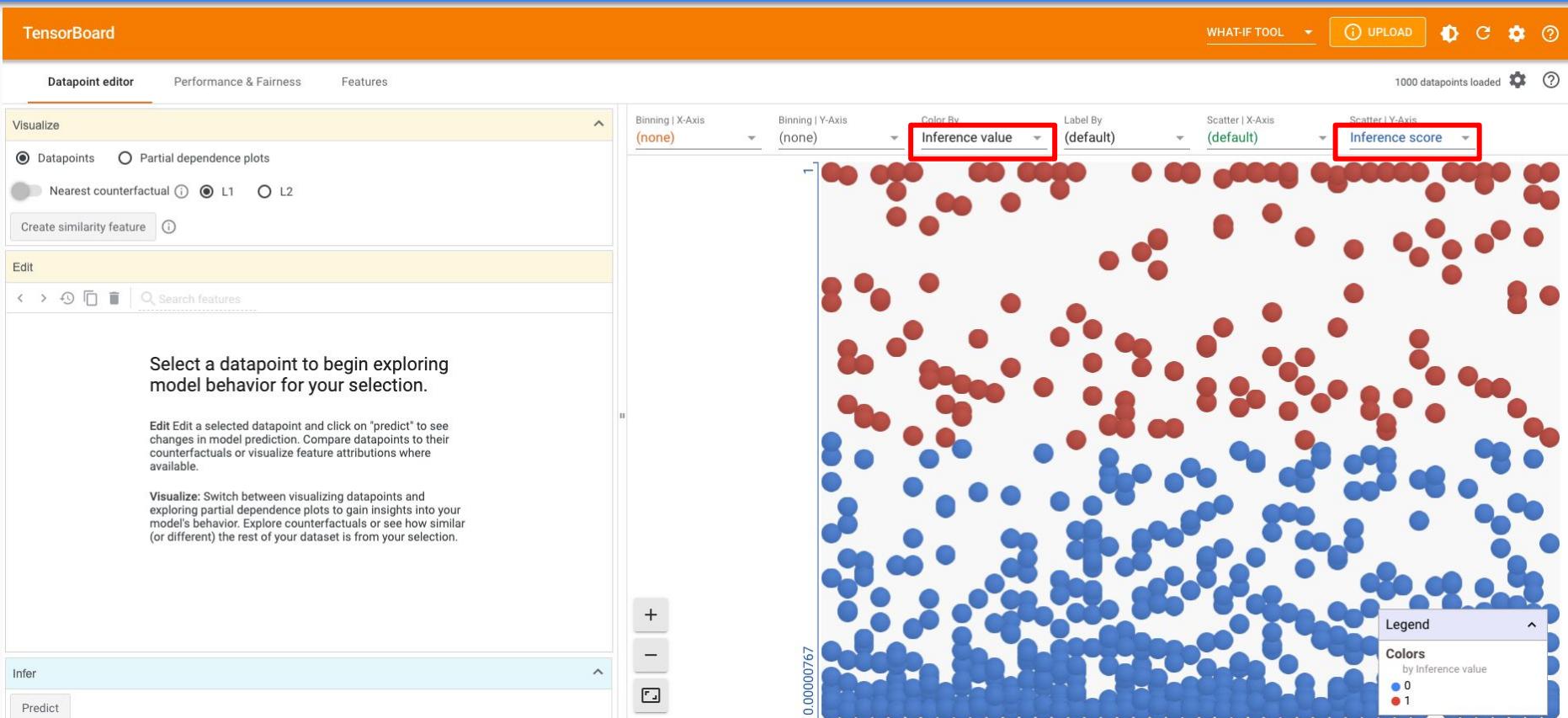
1. Our online model serving address:  
uci-census-predictor-default.kubeflow-user-example-com.svc.cluster.local:80

2. Model name:  
uci-census

3. Our test data used in this study:  
/data/uci\_census/adult.tfrecord

4. Number of samples used in this study: 1000

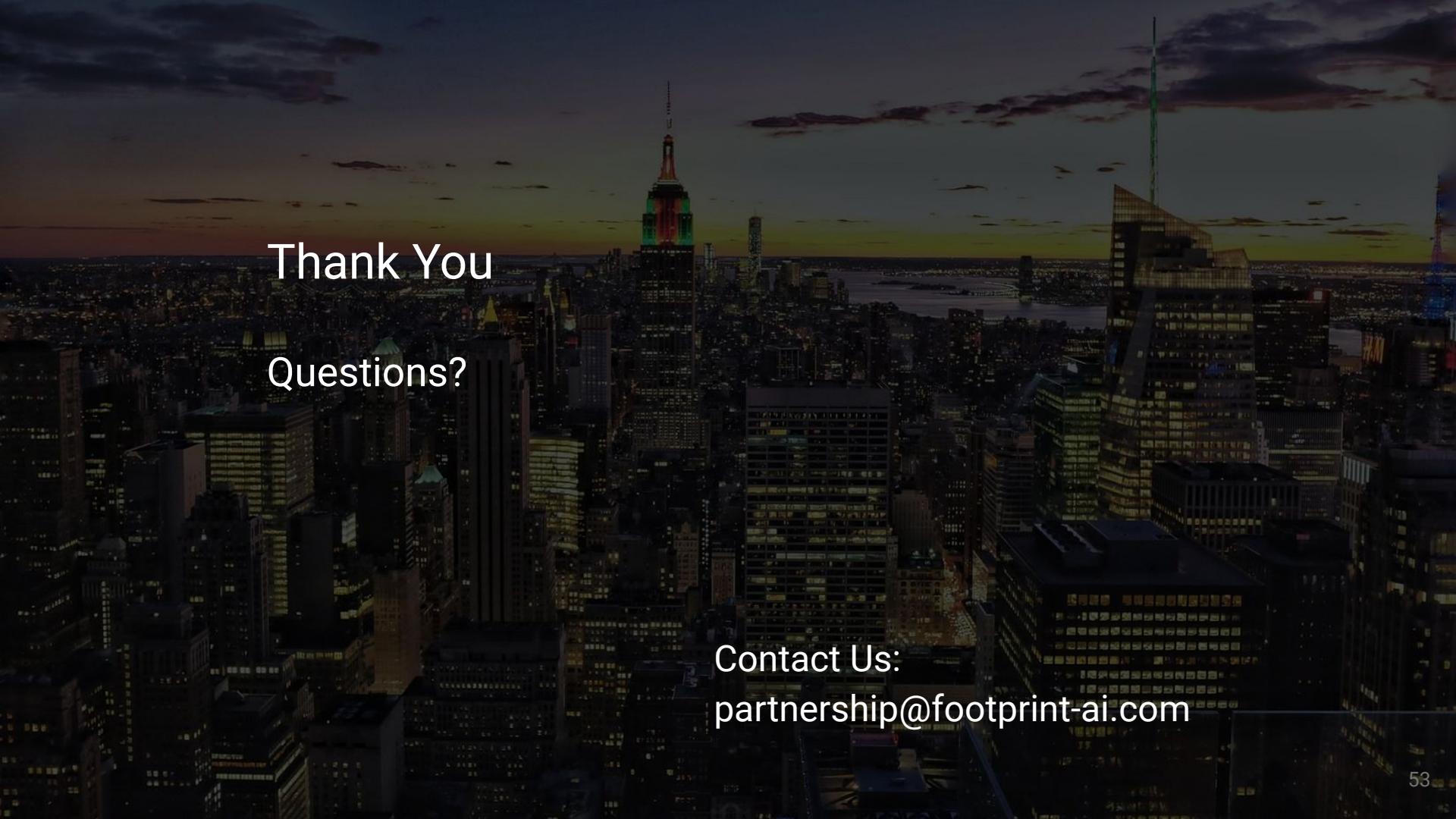
## Step7: configure what-if tool with tensorboard (3/3)



- [UCI Census Salary Prediction](#)
- [Smile Detection](#)
- [Flower Detection](#)

# What we have achieved...

- Understand fairness and explainable AI
- Build a Kubeflow pipeline for
  - Kserve for serving a pre-build machine learning models
  - Tensorboard for analyzing fairness and explainable on the given data
- Interactive with What-if Tool



Thank You

Questions?

Contact Us:  
[partnership@footprint-ai.com](mailto:partnership@footprint-ai.com)

# Reference

- Documentations
  - <https://www.kubeflow.org/>
- Kubectl cheatsheet
  - <https://kubernetes.io/docs/reference/kubectl/cheatsheet/>