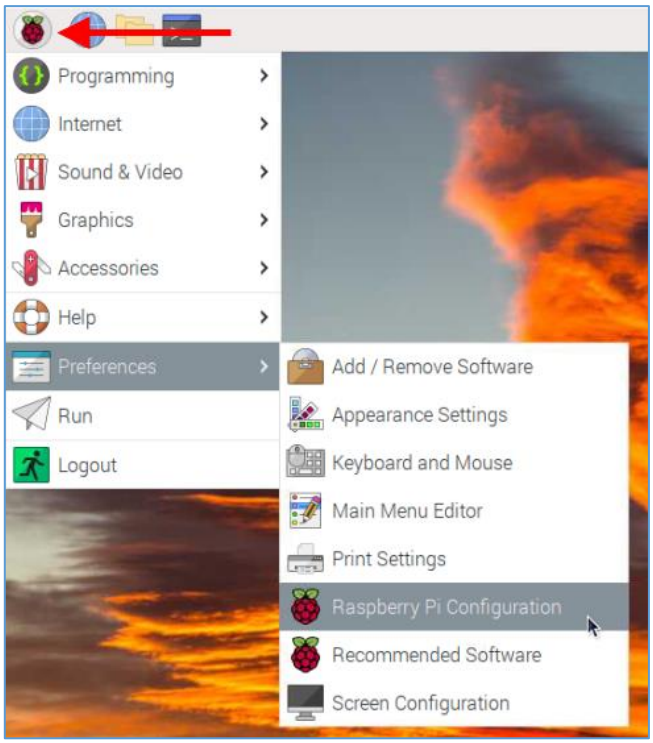
Machine Learning for Raspberry Pi (64 บิต)

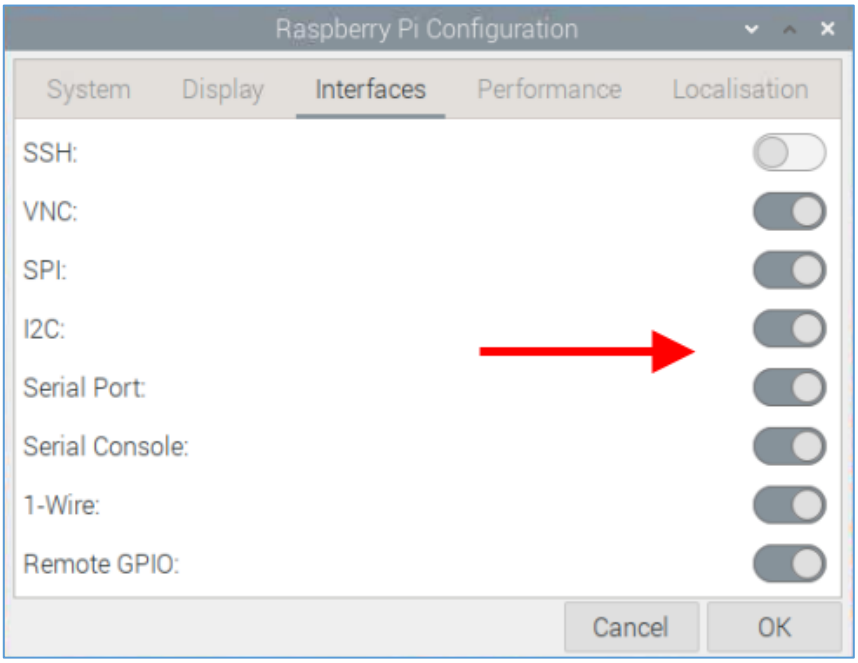
1. ติดตั้งระบบปฏิบัติการ Raspbian 64 บิต

2. เปิดใช้งานอินเทอร์เฟส

2.1 คลิกที่ --> Preferences --> Raspberry Pi Configuration

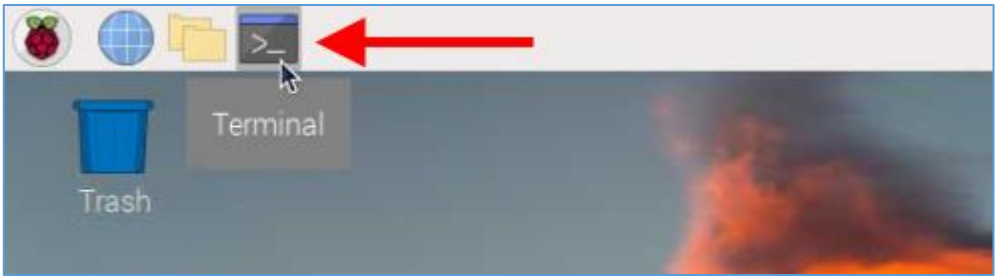


2.2 คลิกเปิดการใช้งาน แล้วคลิกปุ่ม OK



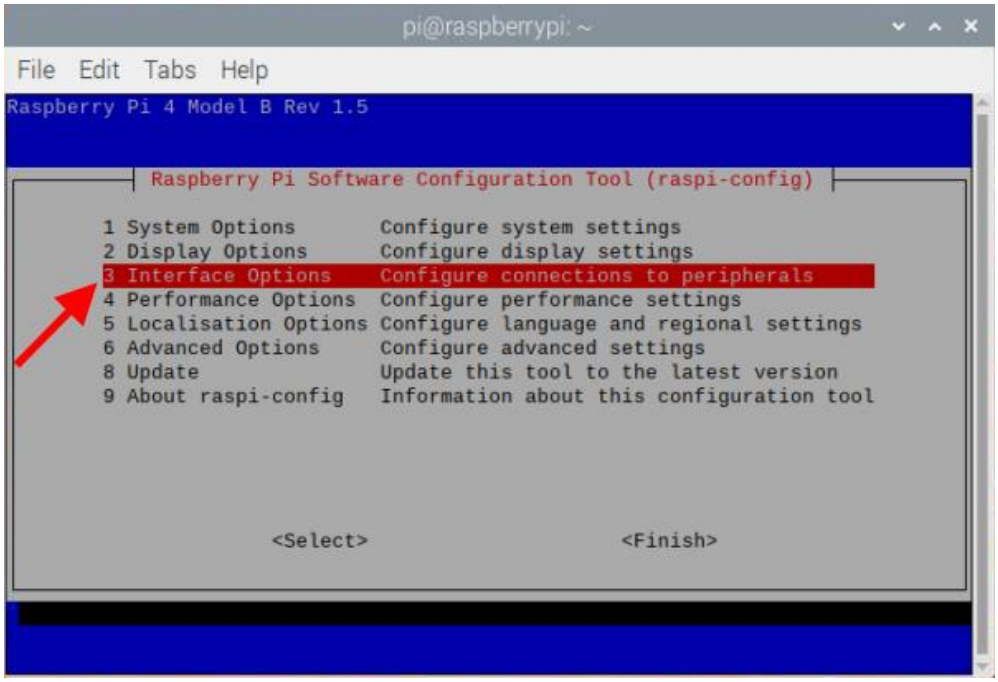
3. เปิดใช้งานกล้อง

3.1 เปิดโปรแกรม Terminal

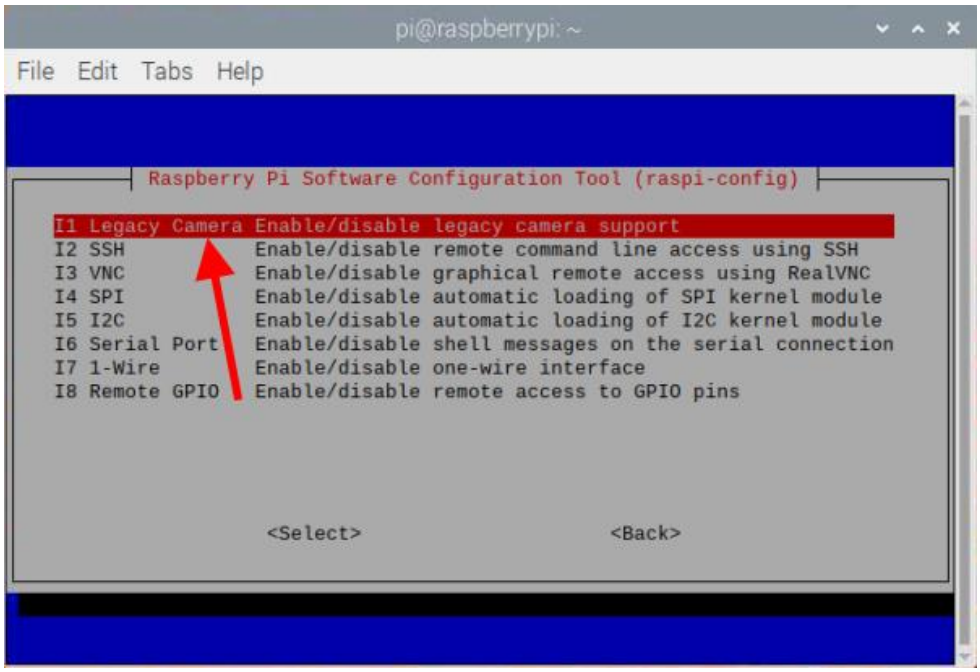


3.2 พิมพ์คำสั่ง sudo raspi-config

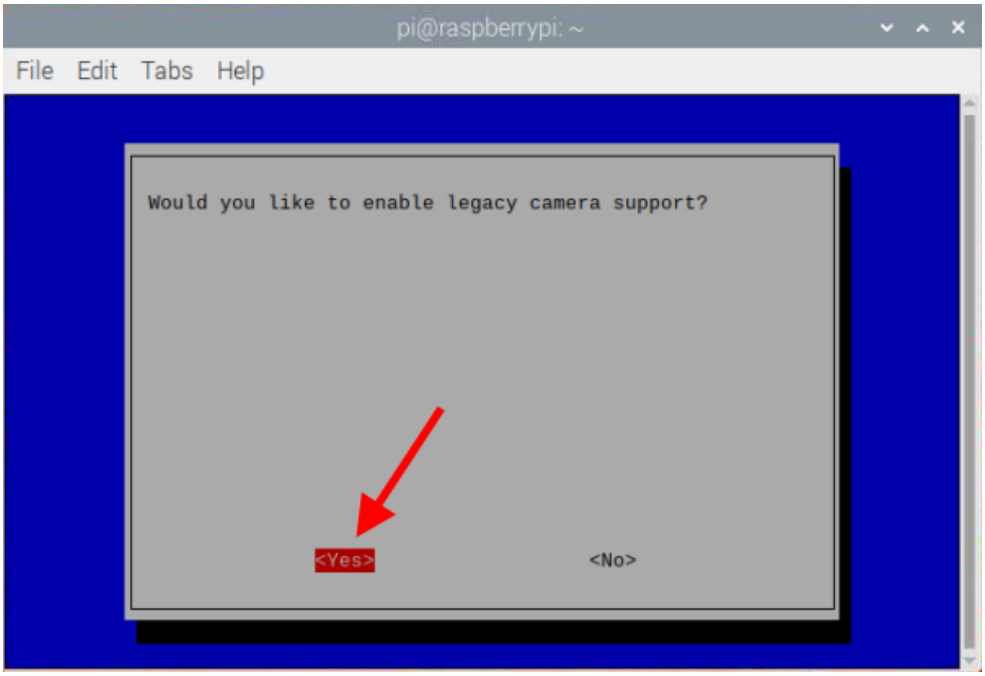
3.3 เลือก Interface Options แล้ว Enter



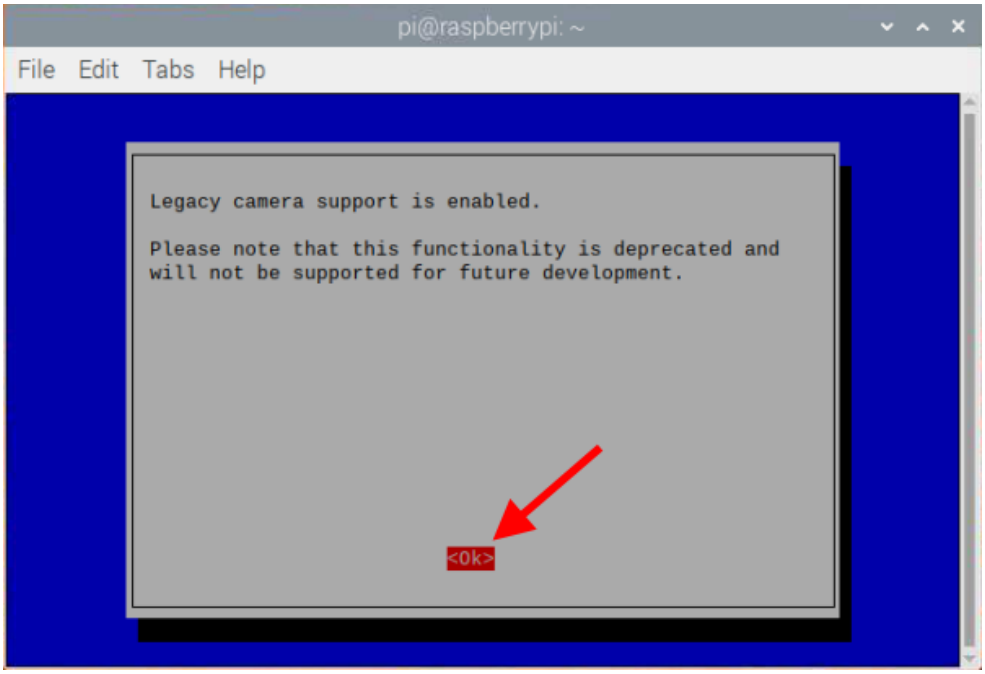
3.4 เลือก I1 Legacy Camera แล้ว Enter



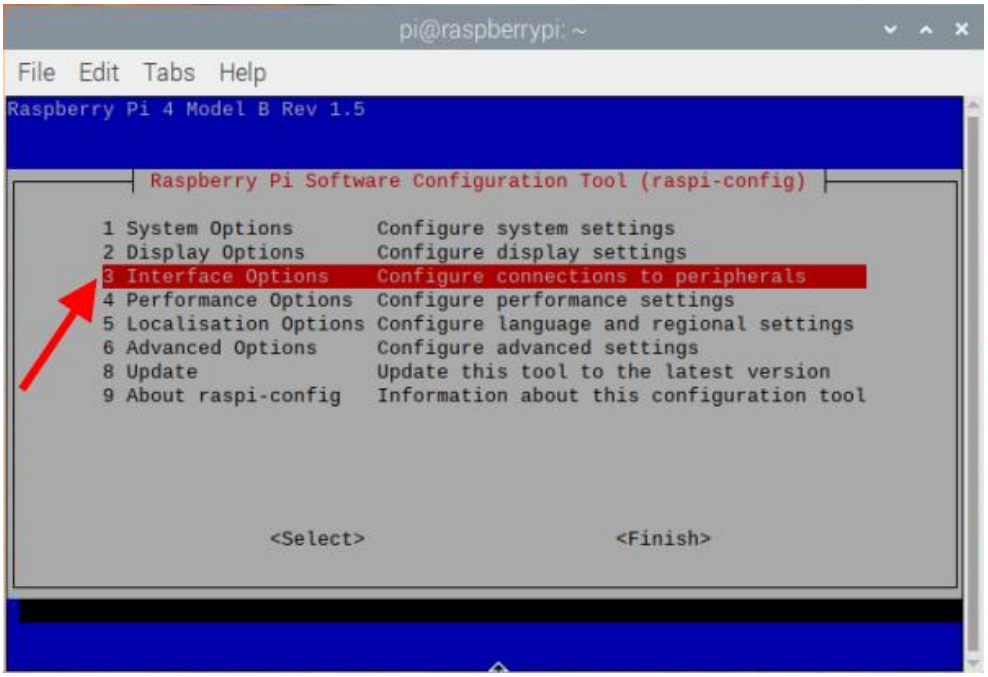
3.5 เลือก <Yes> แล้ว Enter



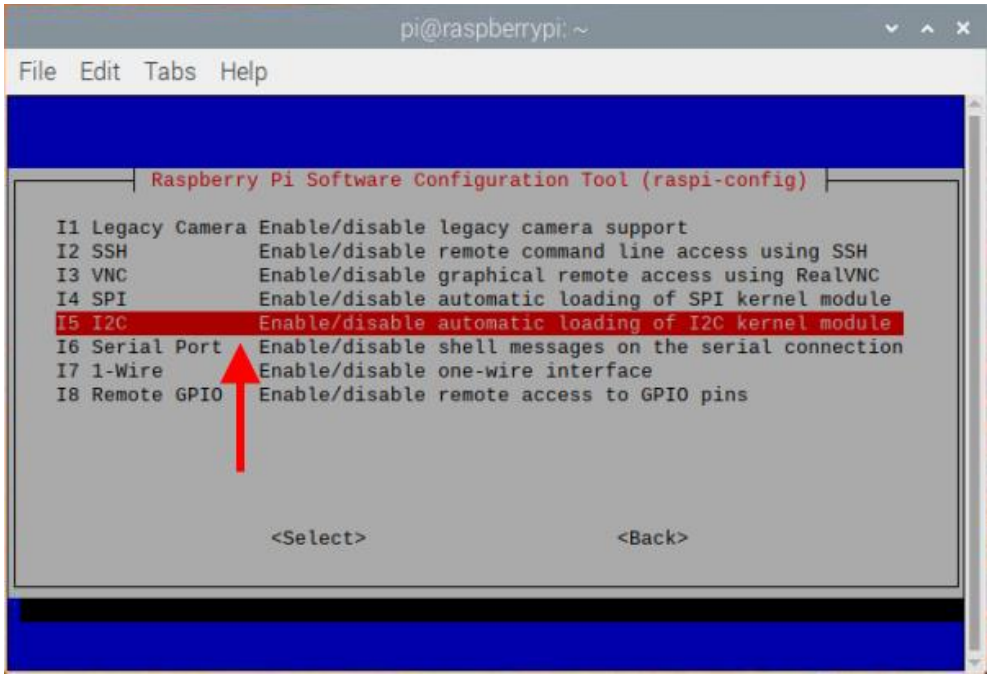
3.6 เลือก <Ok > แล้ว Enter



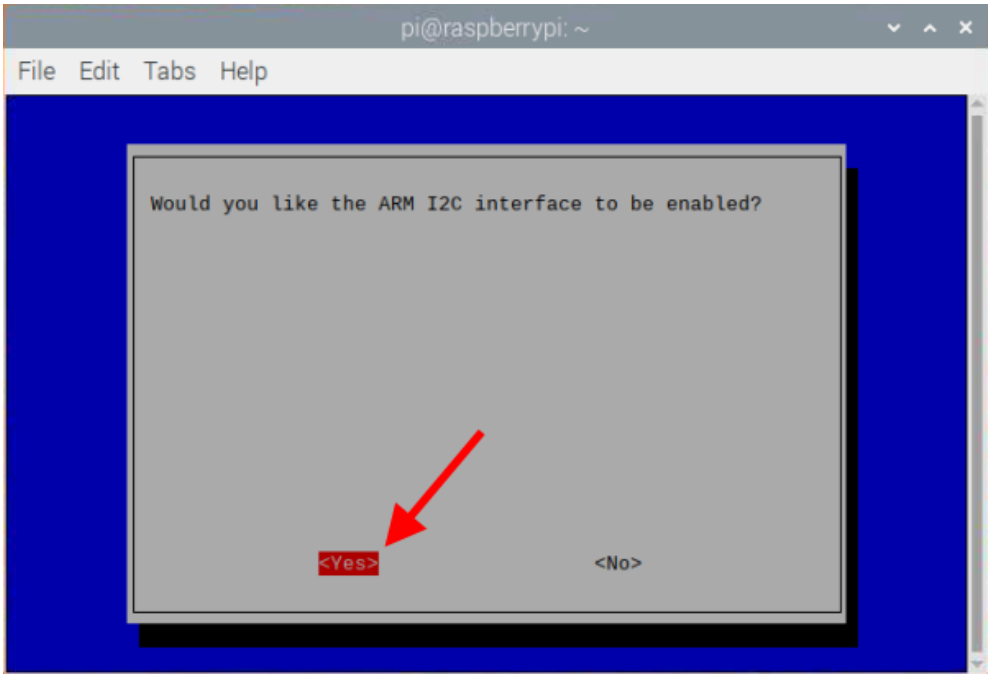
3.7 เลือก Interface Options อีกครั้ง แล้ว Enter



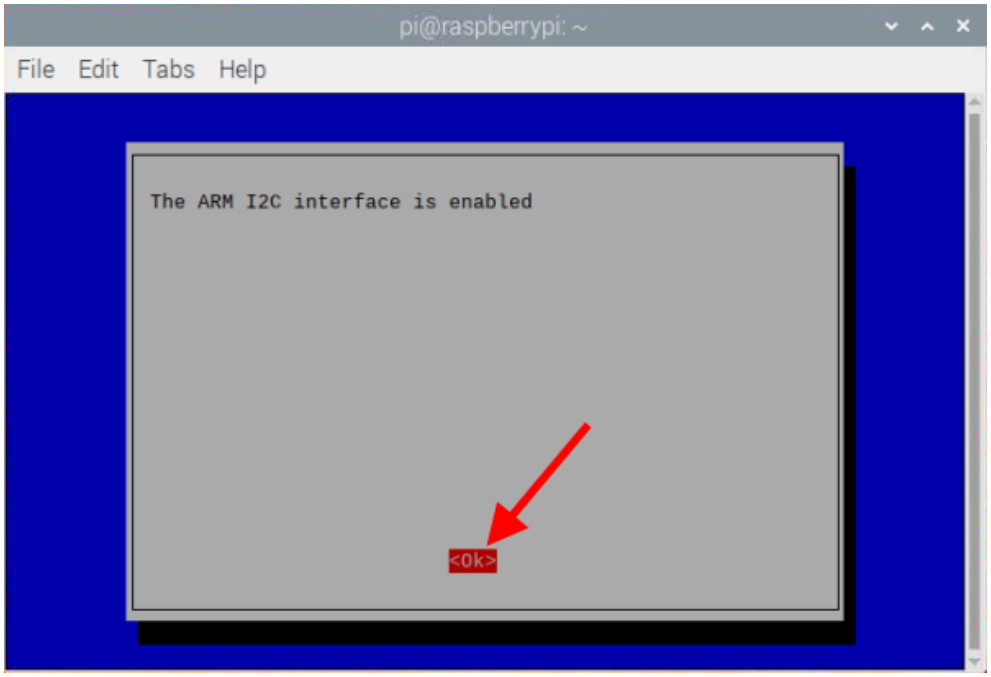
3.8 เปิดใช้งาน I2C โดยเลือก I5 I2C แล้ว Enter



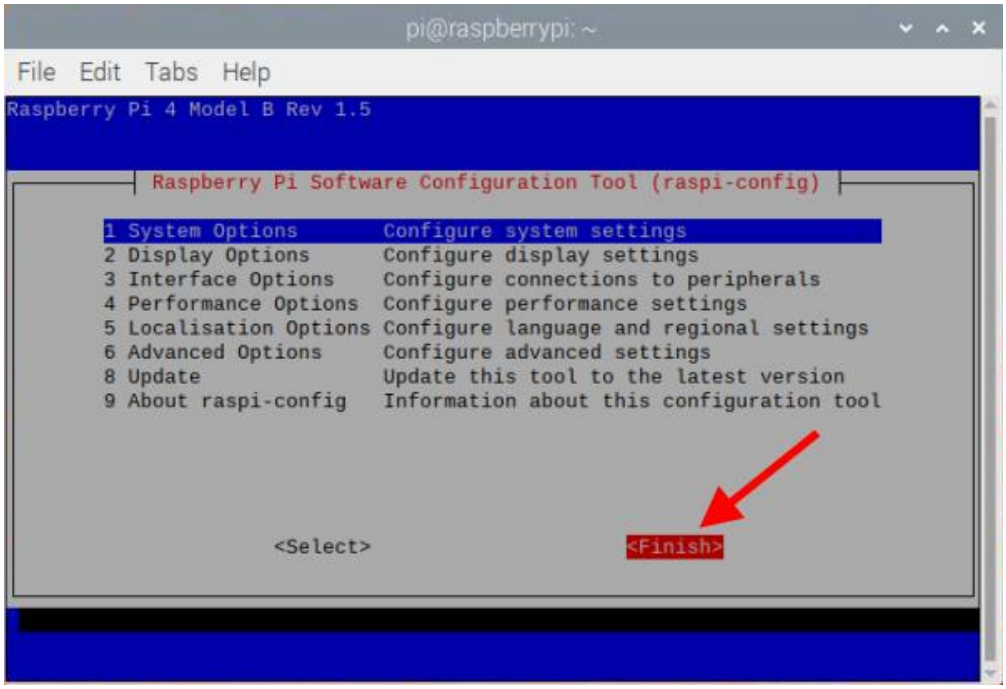
3.9 เลือก <Yes> แล้ว Enter



3.10 เลือก <Ok> แล้ว Enter



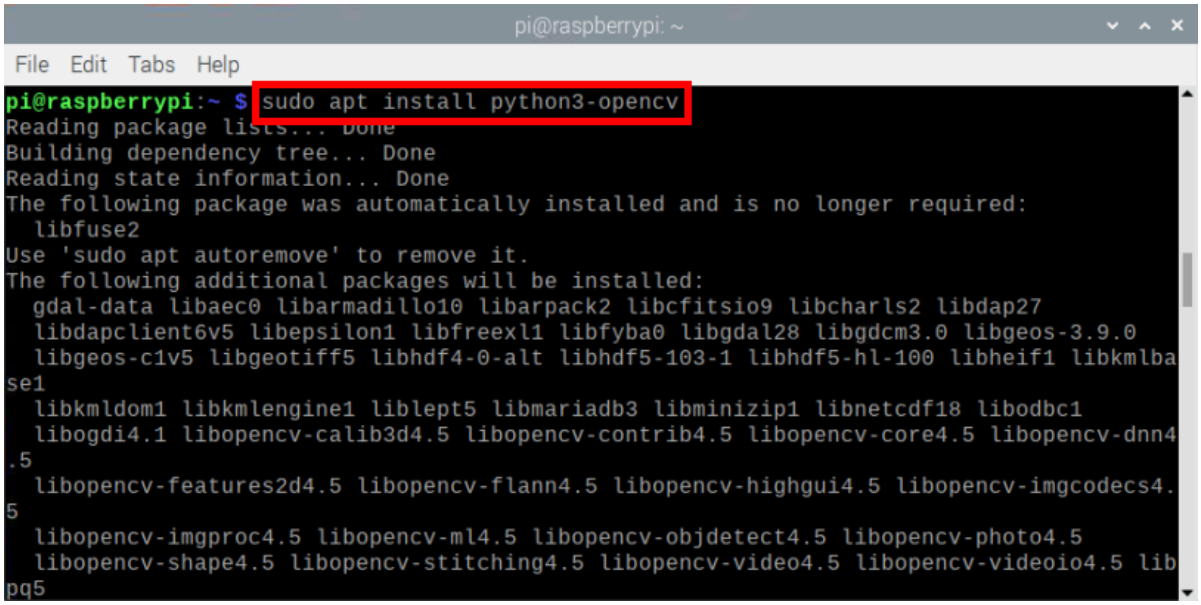
3.11 เลือก <Finish> แล้ว Enter



3.12 ระบบจะให้ Reboot 1 ครั้ง

4. ติดตั้งไลบรารี่ OpenCV

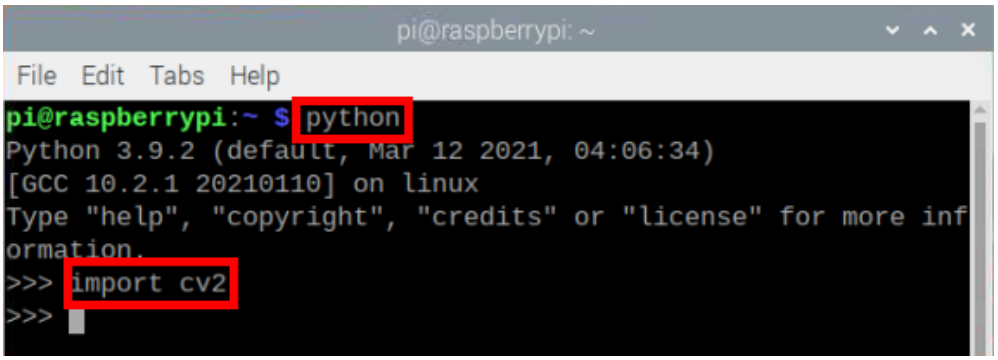
4.1 พิมพ์คำสั่ง sudo apt install python3-opencv



4.2 ทดสอบโดยพิมพ์คำสั่ง

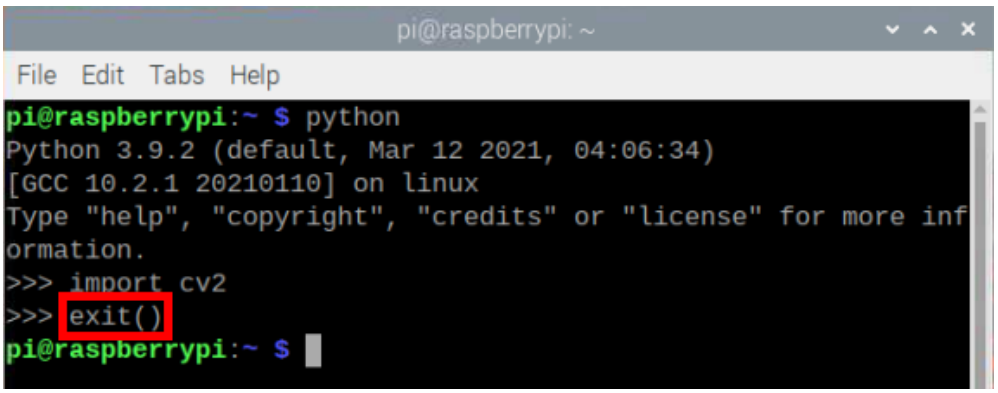
python

import cv2

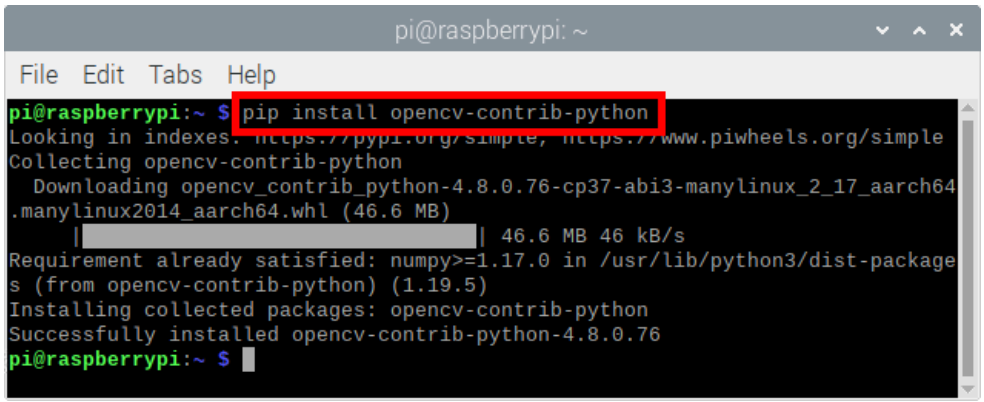


4.3 ตรวจสอบเวอร์ชั่น พิมพ์คำสั่ง cv2.__version__

4.4 ออกจากคำสั่ง python โดยพิมพ์คำสั่ง exit()



4.5 พิมพ์คำสั่ง pip install opencv-contrib-python

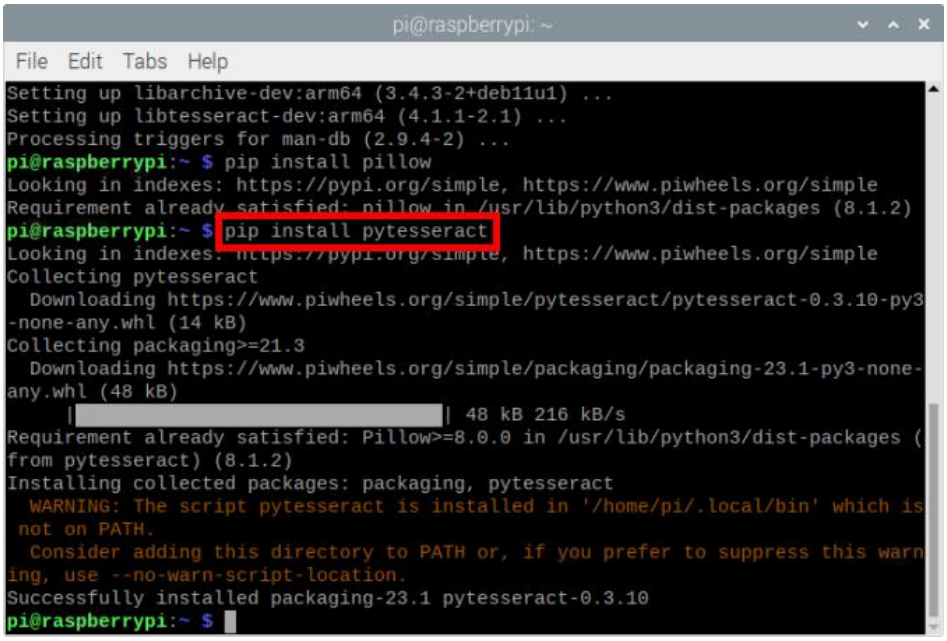


Workshop 1 Text Detection การแปลงข้อความที่อยู่ในรูปภาพให้เป็นข้อความ

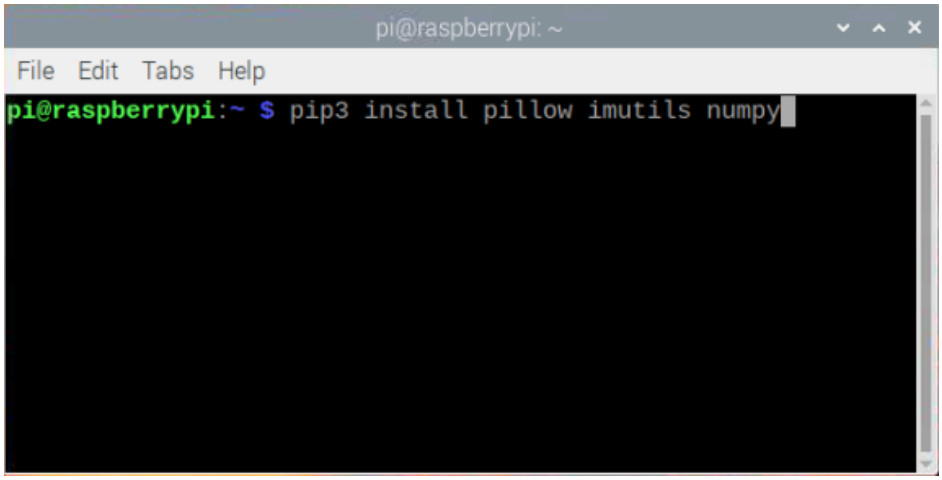
1. ติดตั้งไลบรารี่ pytesseract

1.1 เปิดโปรแกรม Terminal จากนั้นพิมพ์คำสั่ง sudo apt install tesseract-ocr

1.2 พิมพ์คำสั่ง pip install pytesseract

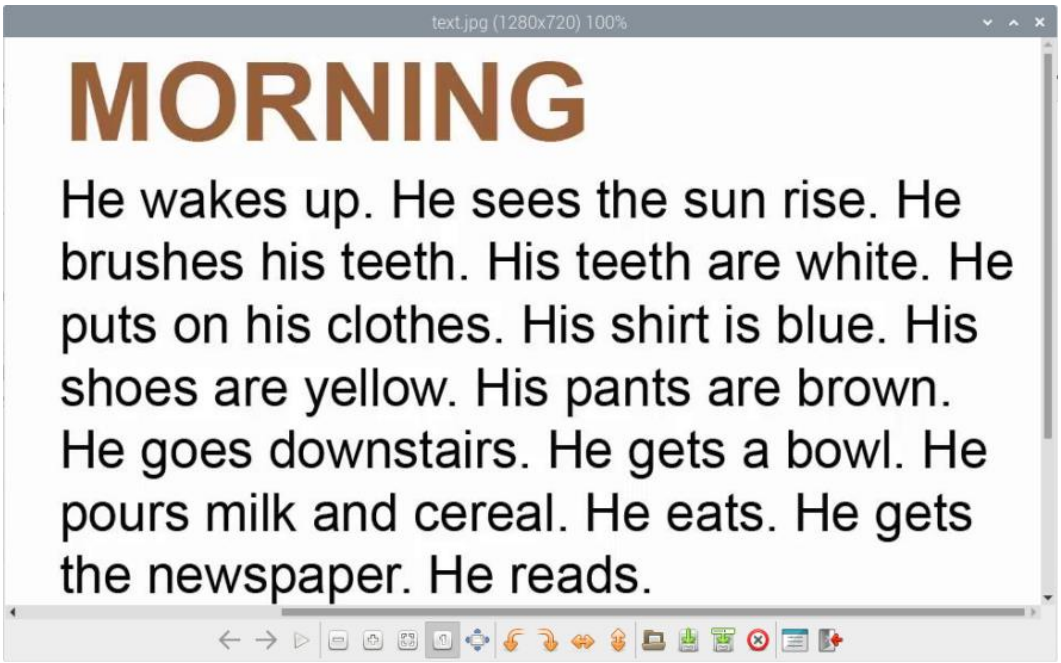


2. ติดตั้ง pillow, imutils และ numpy โดยพิมพ์คำสั่ง pip3 install pillow imutils numpy

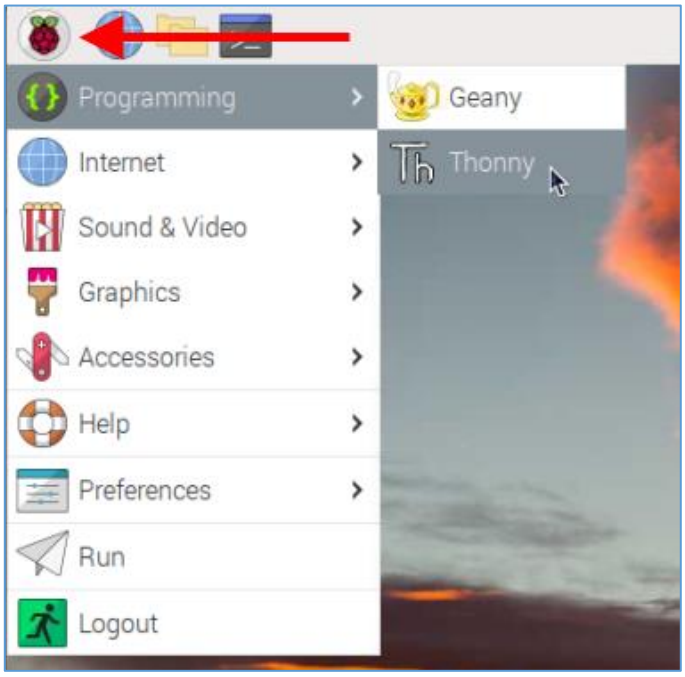


3. Workshop 1.1 โปรแกรมแปลงข้อความจากรูปภาพให้เป็นข้อความ

3.1 ค้นหารูปภาพที่มีข้อความจากอินเทอร์เน็ต บันทึกในโฟลเดอร์ /home/pi โดยตั้งชื่อว่า text.jpg ตัวอย่างดังรูป



3.2 เปิดโปรแกรม Thonny โดยคลิกที่ --> Programming --> Thonny

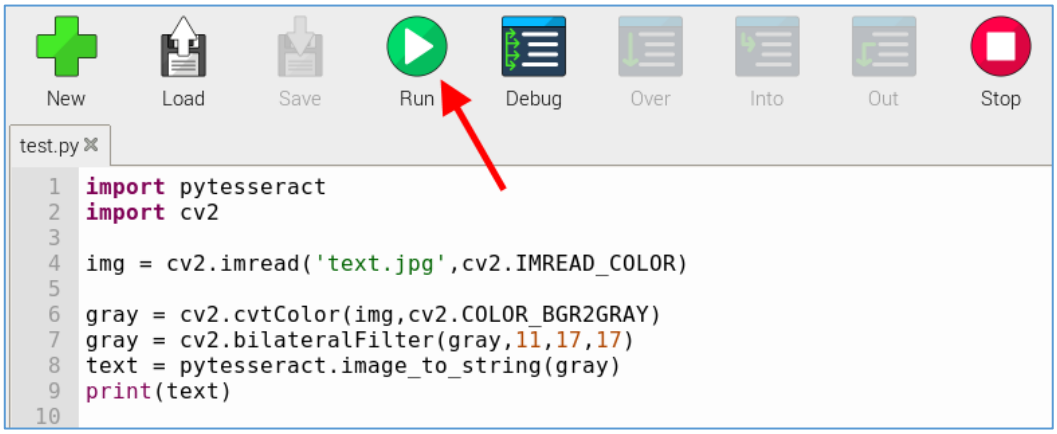


3.3 เขียนโค้ดคำสั่ง ดังรูป

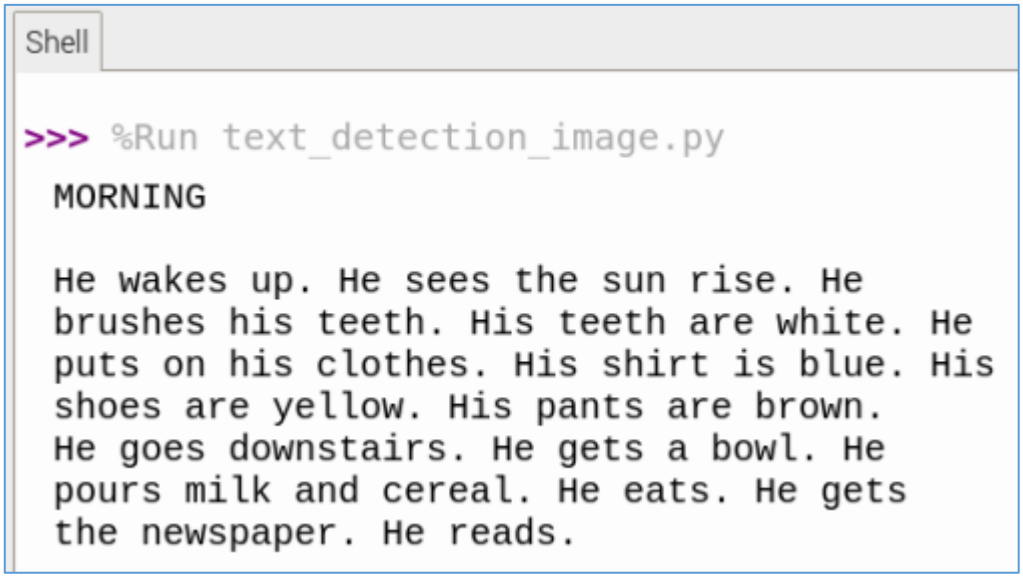
test.py

```
1  import pytesseract
2  import cv2
3
4  img = cv2.imread('text.jpg',cv2.IMREAD_COLOR)
5
6  gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
7  gray = cv2.bilateralFilter(gray,11,17,17)
8  text = pytesseract.image_to_string(gray)
9  print(text)
```

3.4 คลิกปุ่ม Run



3.5 จะได้ข้อความที่อยู่ในรูปภาพ text.jpg ปรากฏขึ้นมาในช่อง Shell



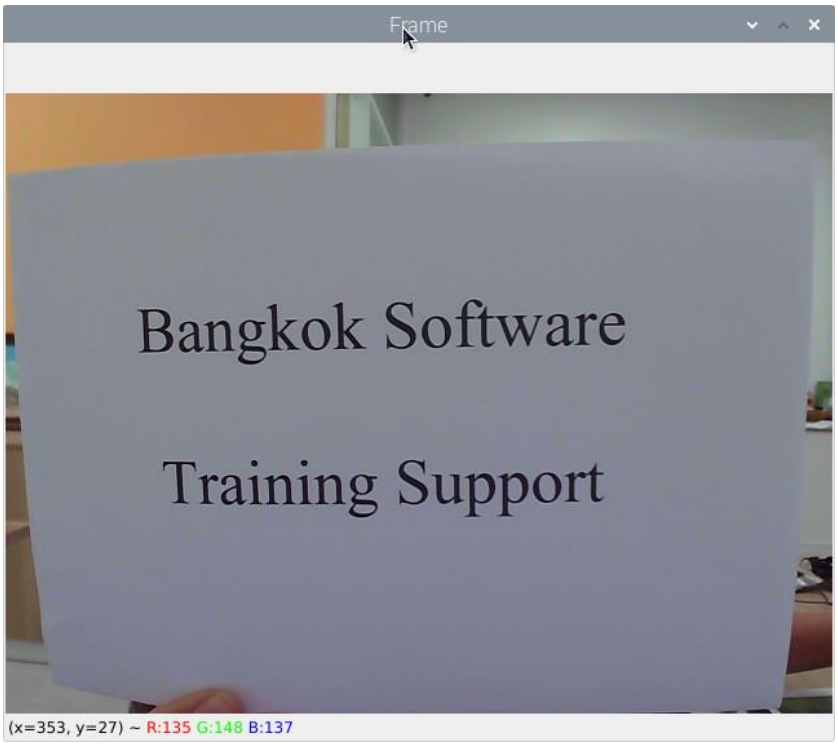
4. Workshop 1.2 โปรแกรมแปลงข้อความจากรูปภาพที่ส่องผ่านกล้องเว็บแคมให้เป็นข้อความ แบบที่ 1

4.1 เขียนโค้ดคำสั่ง ดังรูป

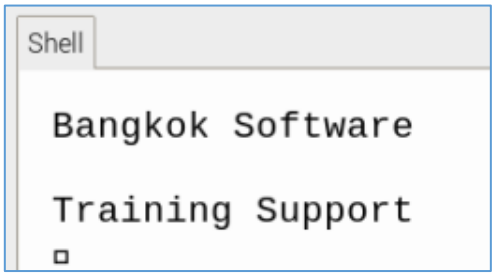
test2.py

```
1  import pytesseract
2  import cv2
3
4  cam = cv2.VideoCapture(0)
5
6  while True:
7      img,frame = cam.read()
8      gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
9      gray = cv2.bilateralFilter(gray,11,17,17)
10     cv2.imshow('Show',frame)
11
12     key = cv2.waitKey(1) & 0xFF
13     if key == ord('s'):
14         text = pytesseract.image_to_string(gray)
15         print(text)
16     if key == ord('q'):
17         break
18 cam.release()
19 cv2.destroyAllWindows()
```

4.2 คลิกปุ่ม Run กล้องจะเปิดขึ้นมา นำกระดาษที่มีข้อความภาษาอังกฤษส่องหน้ากล้อง



4.3 ถ้ากดปุ่ม s บนคีย์บอร์ดจะแปลงข้อความขึ้นมาในช่อง Shell ถ้ากดปุ่ม q จะปิดหน้าต่างลง



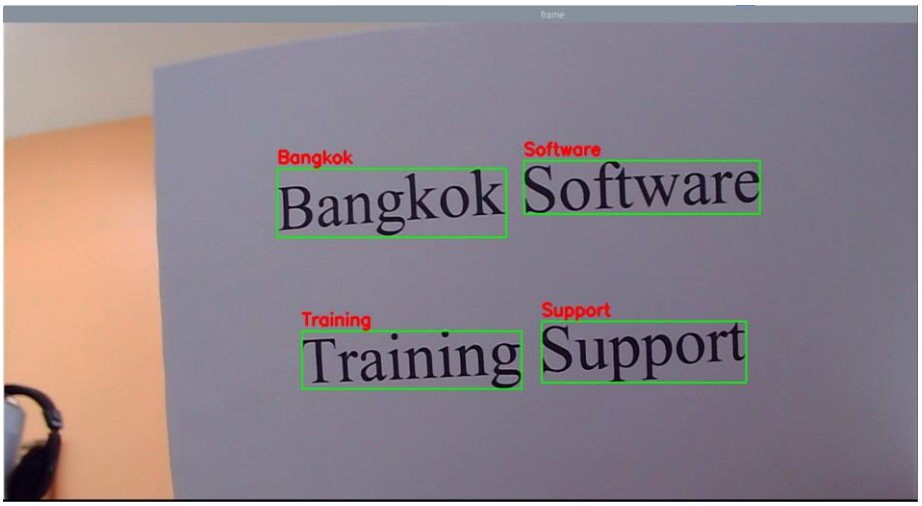
5. Workshop 1.3 โปรแกรมแปลงข้อความจากรูปภาพที่ส่องผ่านกล้องเว็บแคมให้เป็นข้อความ แบบที่ 2

5.1 เขียนโค้ดคำสั่ง ดังรูป

text_detection.py

```
import cv2
import pytesseract
from pytesseract import Output
cap = cv2.VideoCapture(0)
cap.set(cv2.CAP_PROP_BUFFERSIZE, 1)
while True:
    # Capture frame-by-frame
    ret, frame = cap.read()
    d = pytesseract.image_to_data(frame, output_type=Output.DICT)
    n_boxes = len(d['text'])
    for i in range(n_boxes):
        if int(d['conf'][i]) > 60:
            (text, x, y, w, h) = (d['text'][i], d['left'][i], d['top'][i], d['width'][i], d['height'][i])
            # don't show empty text
            if text and text.strip() != "":
                frame = cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)
                frame = cv2.putText(frame, text, (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 1.0, (0, 0, 255), 3)
    cv2.imshow('frame', frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release() #When everything done, release the capture
cv2.destroyAllWindows()
```

5.2 คลิกปุ่ม Run กล้องจะเปิดขึ้นมา นำกระดาษที่มีข้อความภาษาอังกฤษส่องหน้ากล้อง

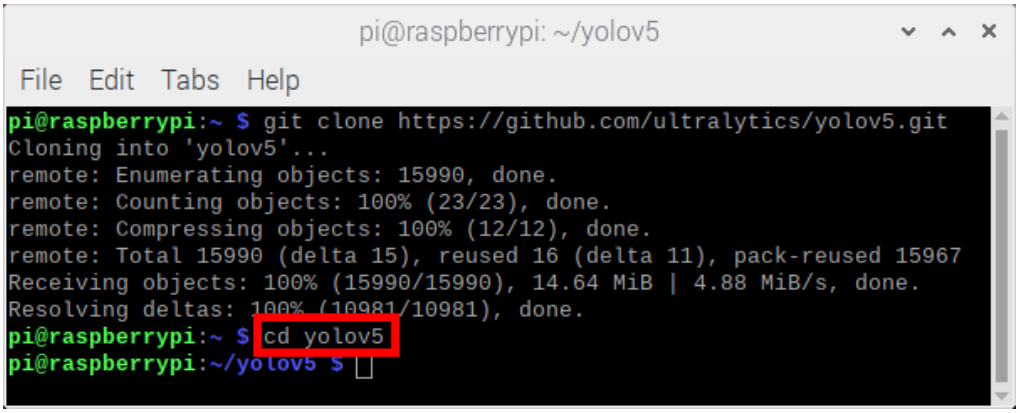


Workshop 2 Object Detection ตรวจจับวัตถุโดยใช้ YOLOv5

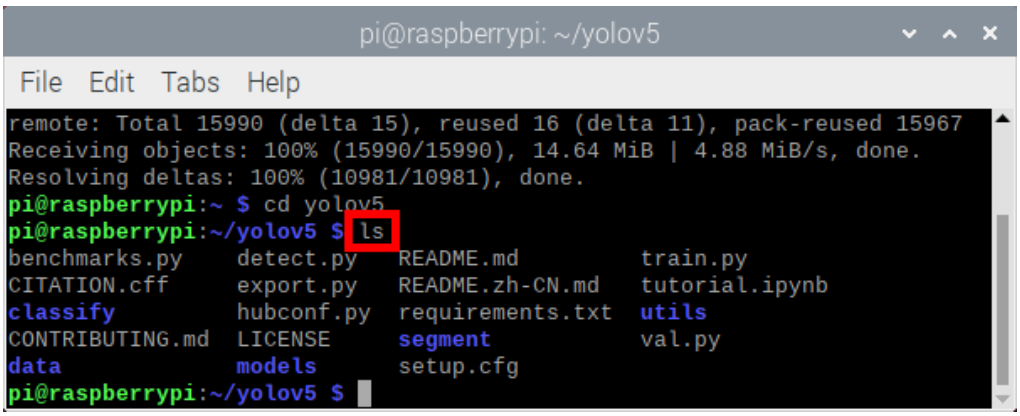
1. การทำ Object Detection จากโมเดลตัวอย่าง

1.1 ดาวน์โหลดโมเดล yolov5 โดยเปิด Terminal แล้วพิมพ์คำสั่ง git clone https://github.com/ultralytics/yolov5.git

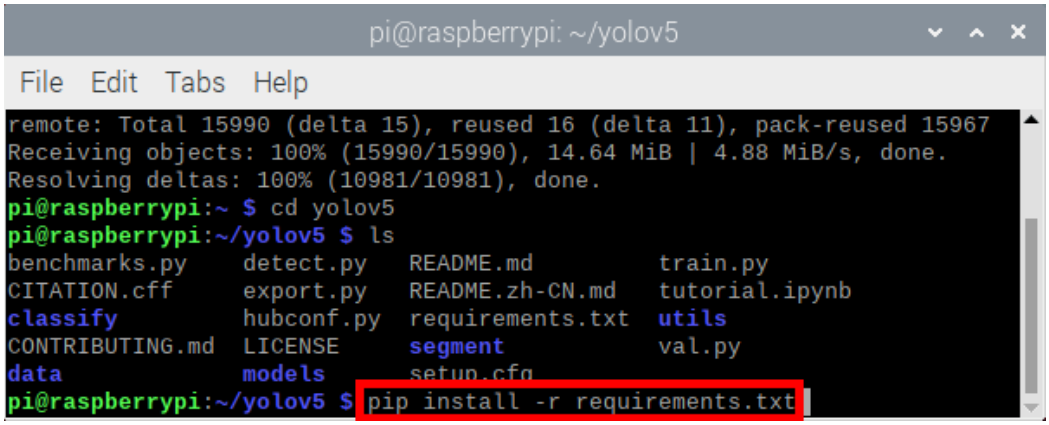
1.2 เข้าไปในโฟลเดอร์ yolov5 โดยพิมพ์คำสั่ง cd yolov5



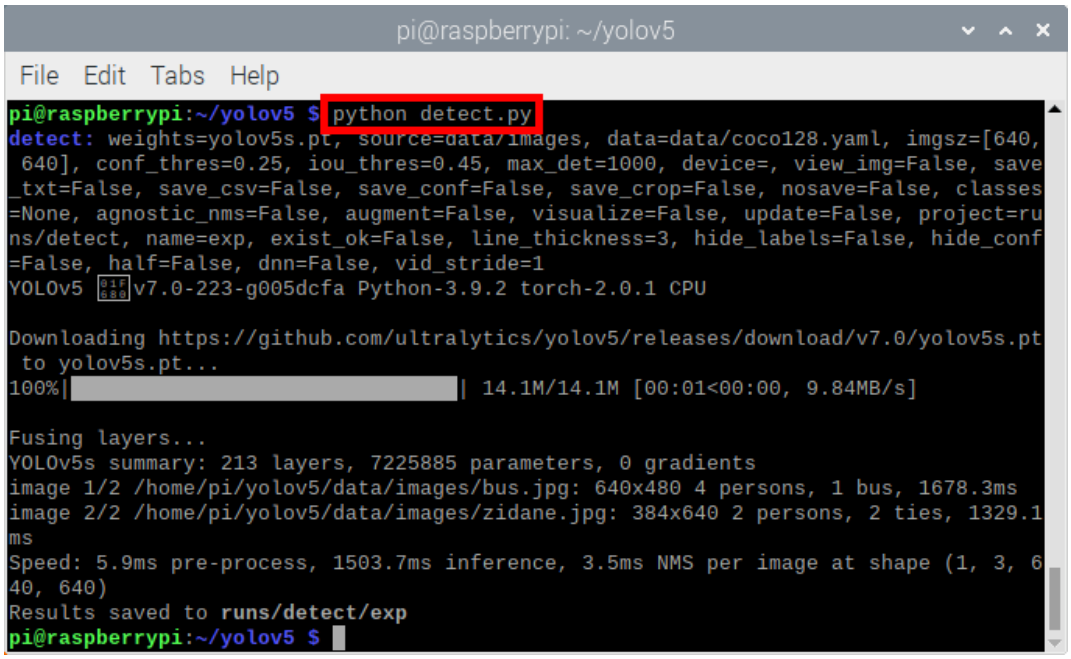
1.3 พิมพ์คำสั่ง ls เพื่อเรียกดูไฟล์ทั้งหมดในโฟลเดอร์ yolov5



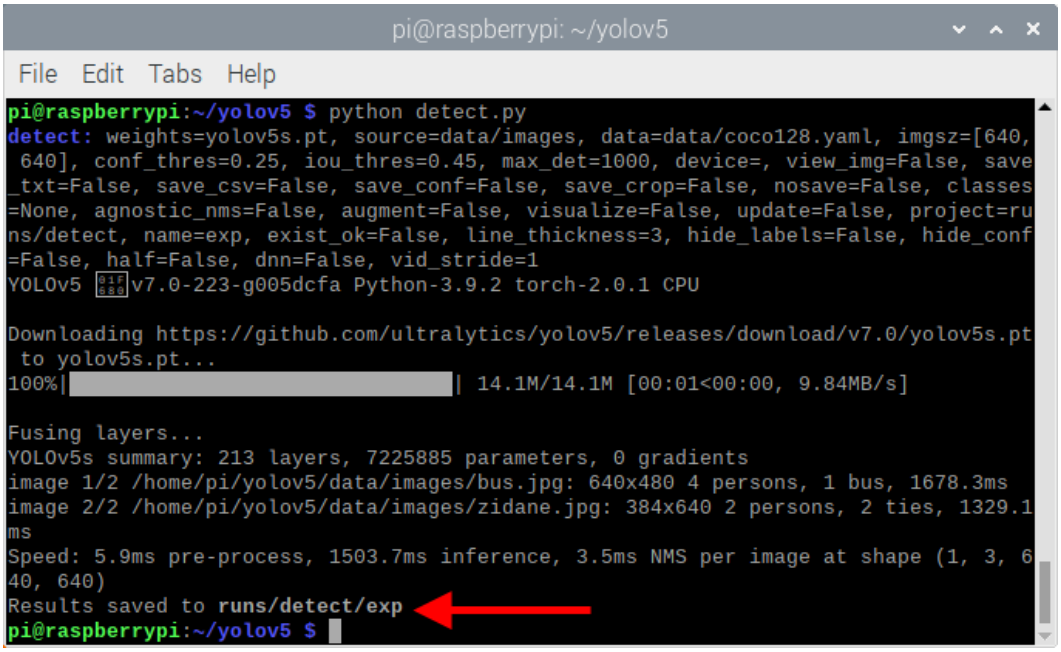
1.4 ติดตั้งไลบรารี่ที่ต้องการ โดยพิมพ์คำสั่ง pip install -r requirements.txt



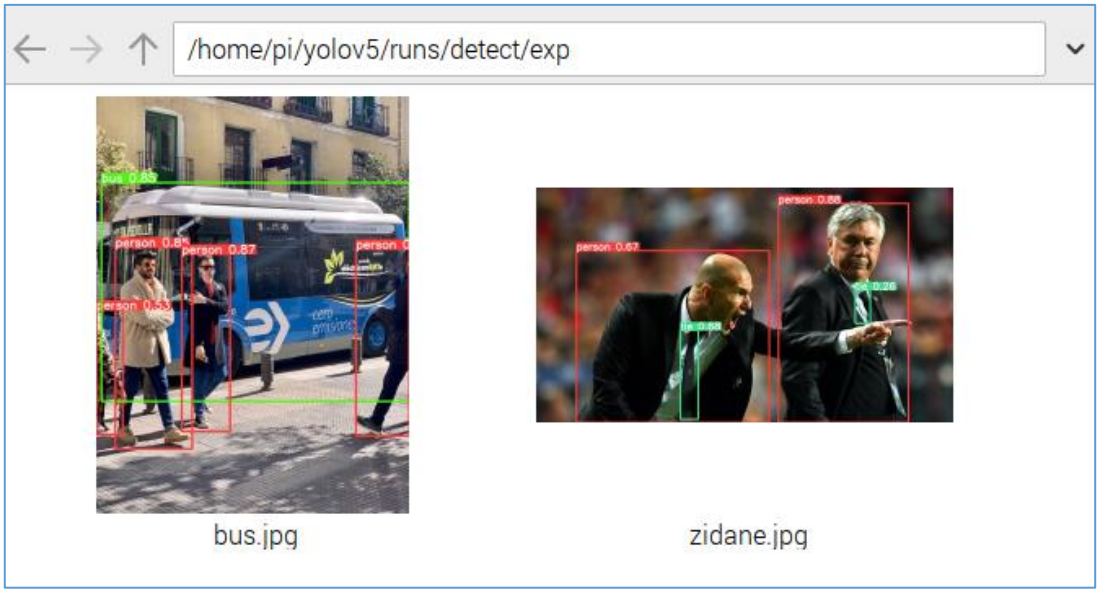
1.5 ทดสอบรันไฟล์ detect.py ซึ่งจะนำรูปภาพต้นแบบมาตรวจจับวัตถุ โดยพิมพ์คำสั่ง python detect.py



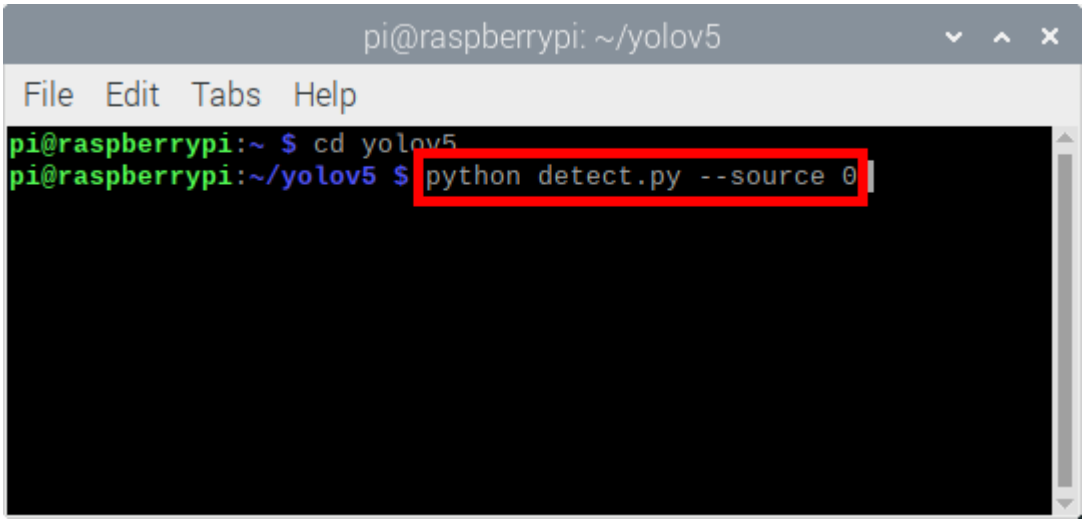
1.6 ดูผลลัพธ์ในโฟลเดอร์ /home/pi/yolov5/runs/detect/exp/



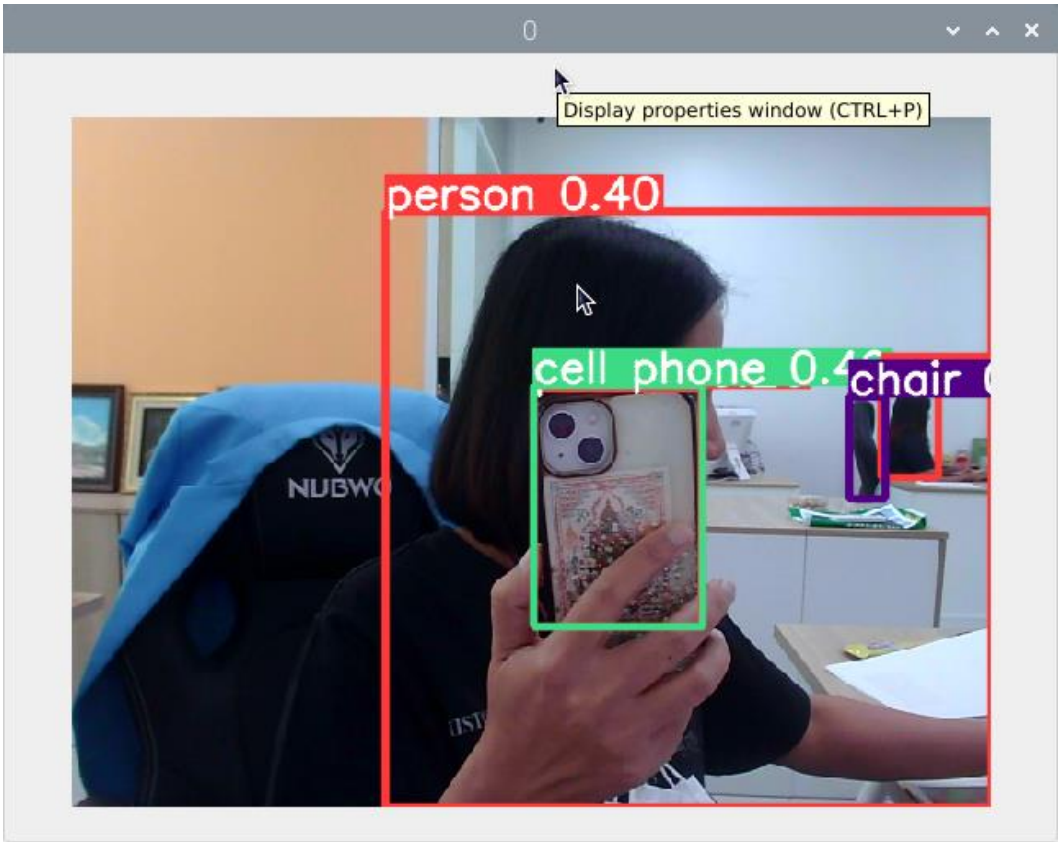
จะปรากฏผลลัพธ์ในโฟลเดอร์ /home/pi/yolov5/runs/detect/exp/ ดังรูป



1.7 รันโค้ดคำสั่งตรวจจับวัตถุโดยใช้กล้อง โดยพิมพ์คำสั่ง python detect.py --source 0



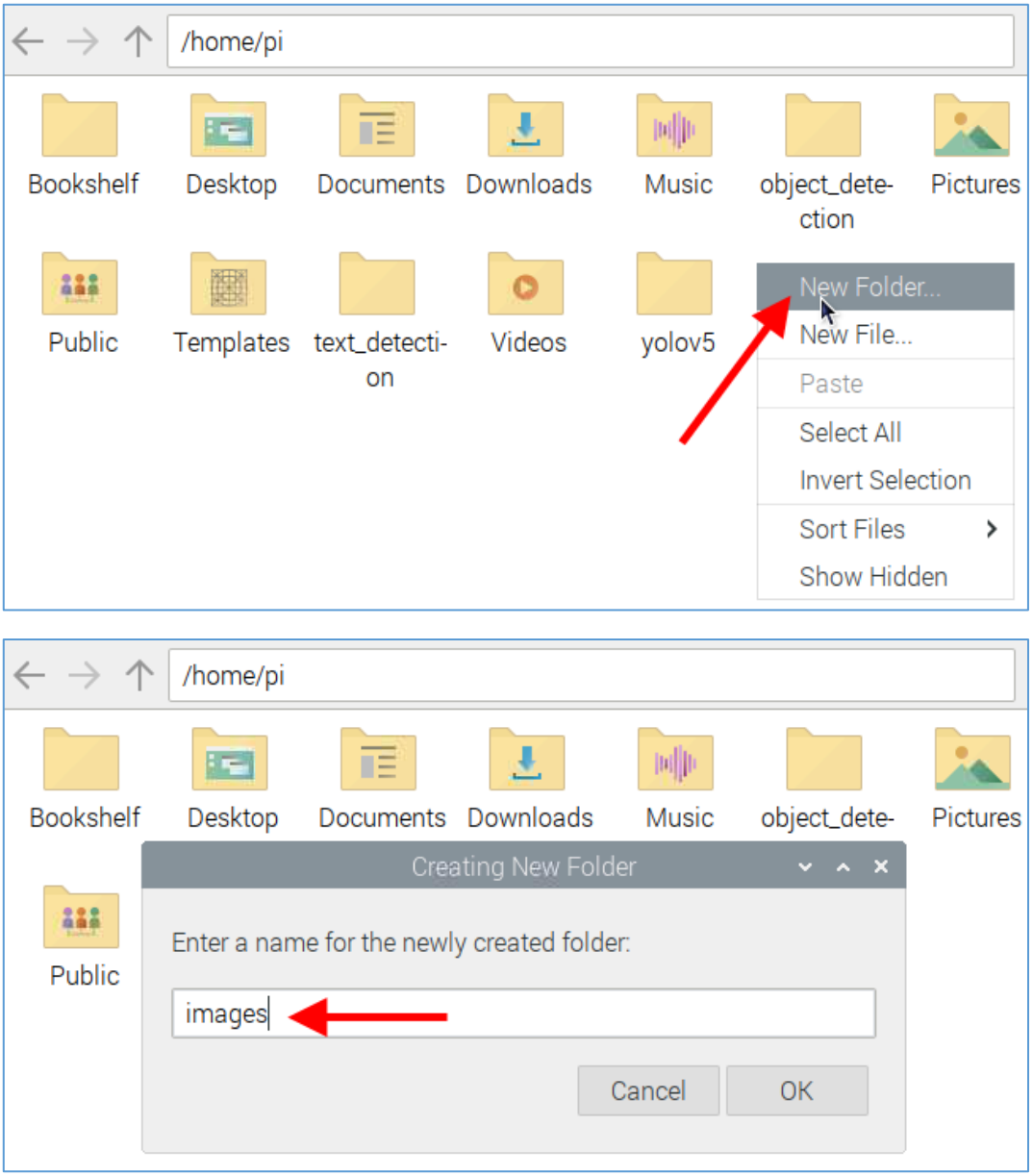
จะได้ผลลัพธ์ ตัวอย่างดังรูป



2. การทำ Custom Object Detection ตรวจจับวัตถุด้วยการสร้างโมเดลของตนเอง

ขั้นตอนที่ 1 ถ่ายรูปวัตถุที่ต้องการนำมาใช้ในการตรวจจับวัตถุ

1.1 สร้างโฟลเดอร์ images สำหรับเก็บรูปภาพที่ถ่ายไว้ใน /home/pi



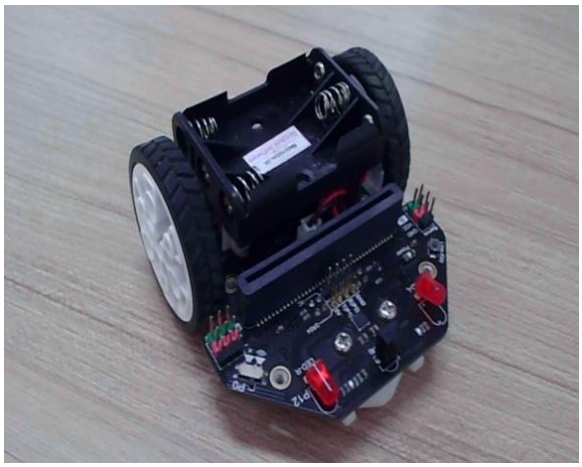
1.2 เปิดโปรแกรม Thonny

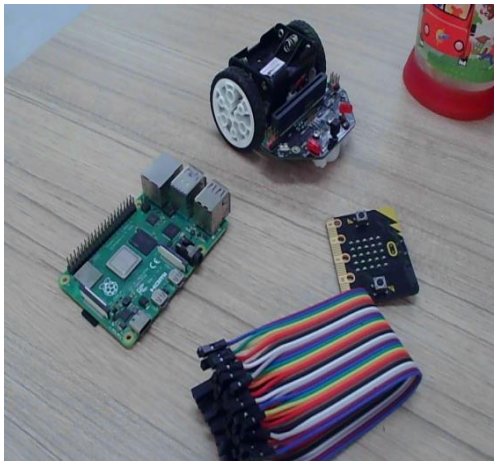
1.3 พิมพ์โค้ดคำสั่งสำหรับถ่ายรูป ให้ได้รูปภาพจำนวน 50 รูป (ยิ่งเยอะยิ่งดี) จากนั้นคลิกปุ่ม Run

capture1.py

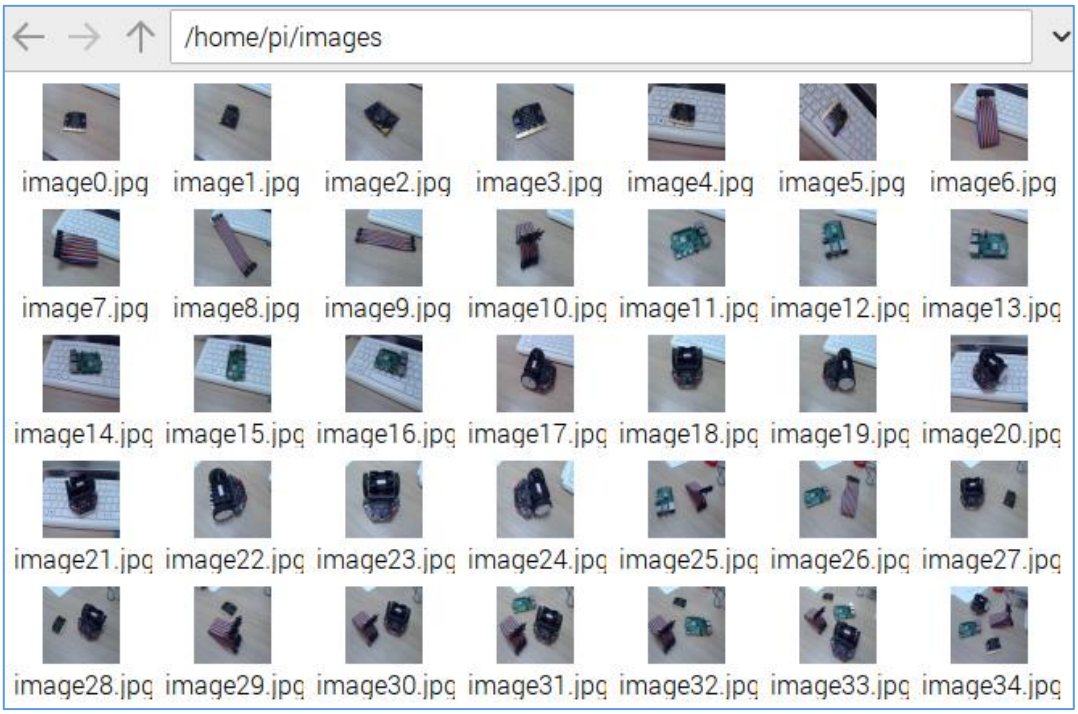
```
1  import cv2
2  import time
3
4  i = 0
5
6  cap = cv2.VideoCapture(0)
7
8  while True:
9      img,frame = cap.read()
10     frame = cv2.resize(frame,(640,640))
11     cv2.imshow("Capture",frame)
12
13     if cv2.waitKey(1) & 0xFF == ord('s'):
14         cv2.imwrite("/home/pi/images/image%d.jpg" %i, frame)
15         i += 1
16     if cv2.waitKey(1) & 0xFF == 27:
17         break
18 cap.release()
19 cv2.destroyAllWindows()
```

1.4 กล้องจะเปิดขึ้นมา ให้นำกล้องมาส่องถ่ายภาพวัตถุ โดยกดปุ่ม s บนคีย์บอร์ดเพื่อถ่ายแต่ละรูป ซึ่งจะต้องปรับเปลี่ยนรูปภาพให้หลากหลายมุมมอง โดยถ่ายทั้งแบบเดี่ยวและแบบกลุ่ม ตัวอย่างเช่น



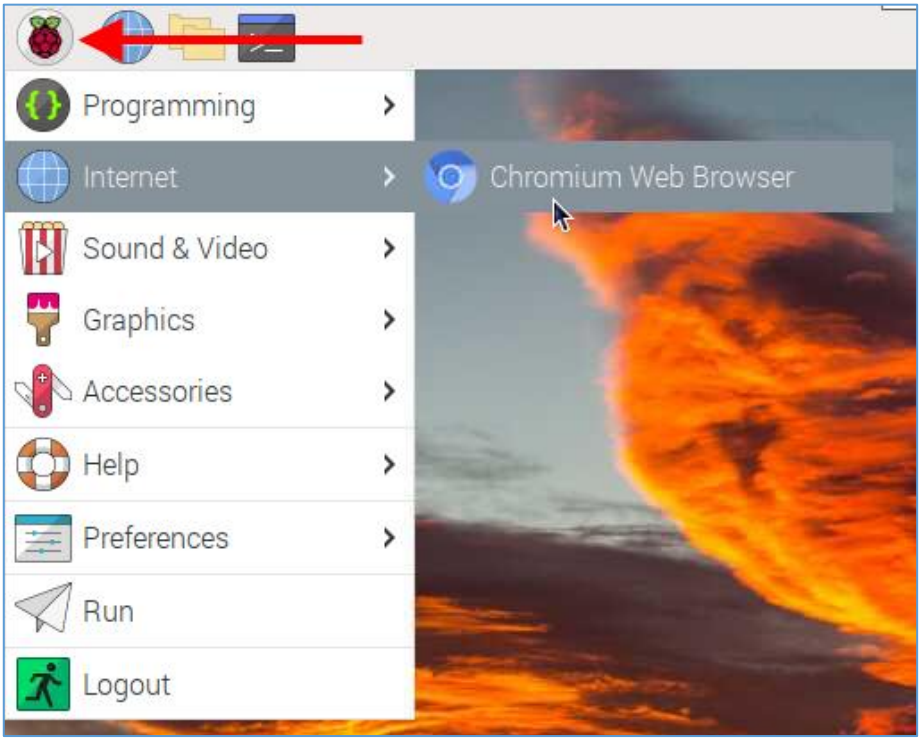


1.5 เปิดโฟลเดอร์ /home/pi/images หากมีรูปภาพใดไม่ต้องการก็สามารถลบทิ้งได้

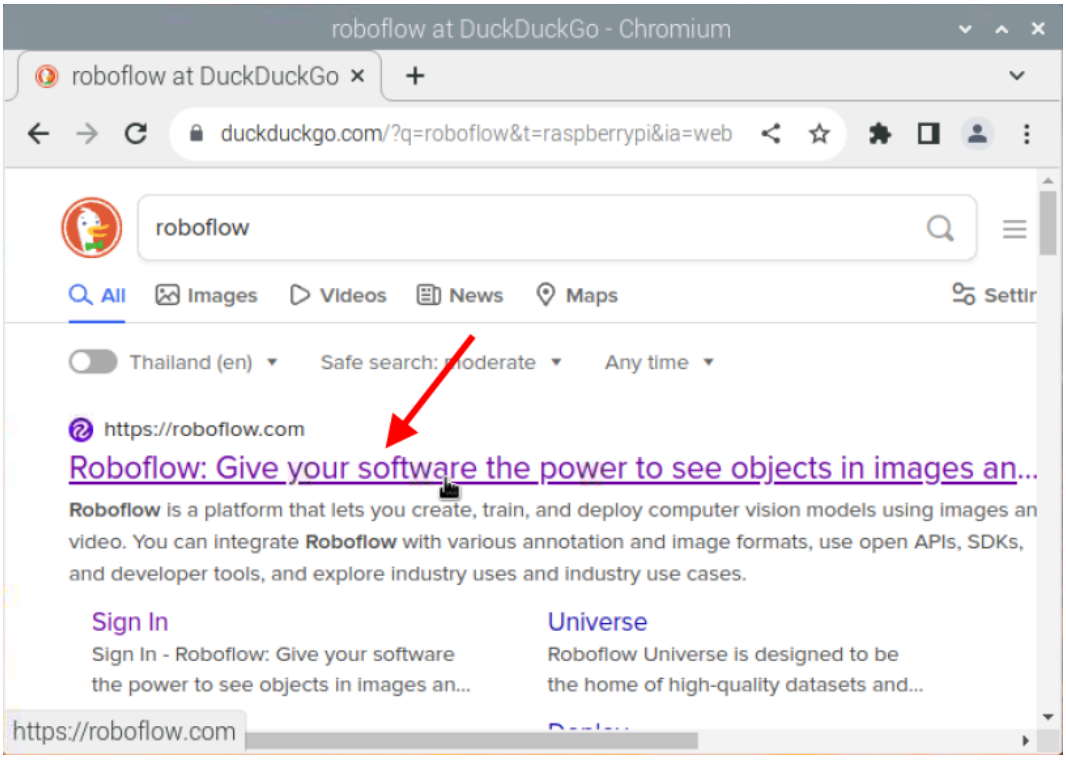


ขั้นตอนที่ 2 ติดป้ายกำกับรูปภาพด้วย RoboFlow

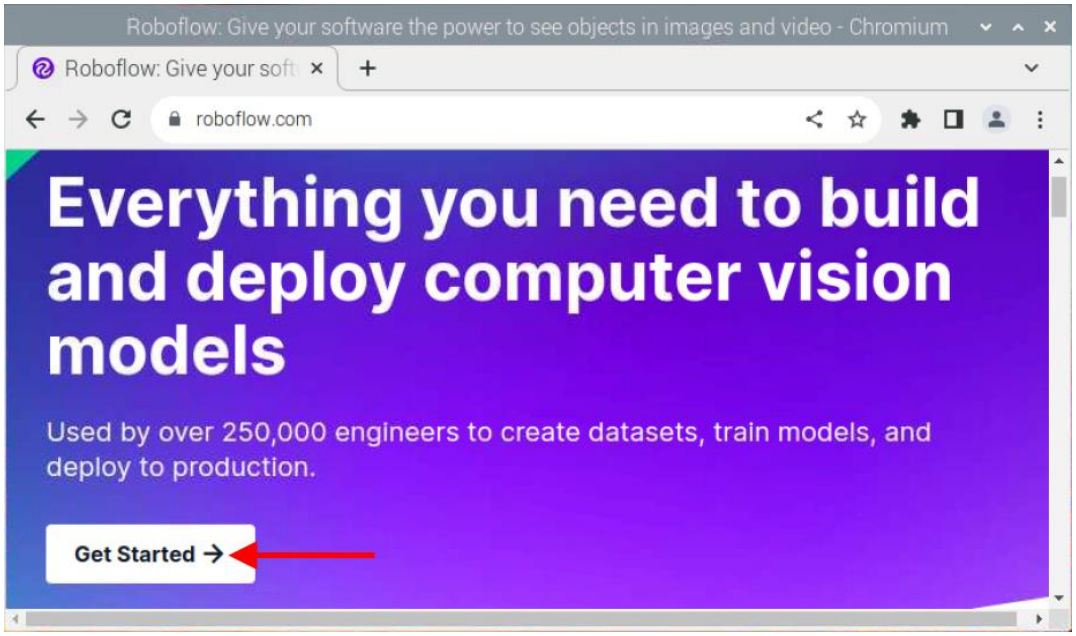
2.1 เปิดโปรแกรม Chromium Web Browser



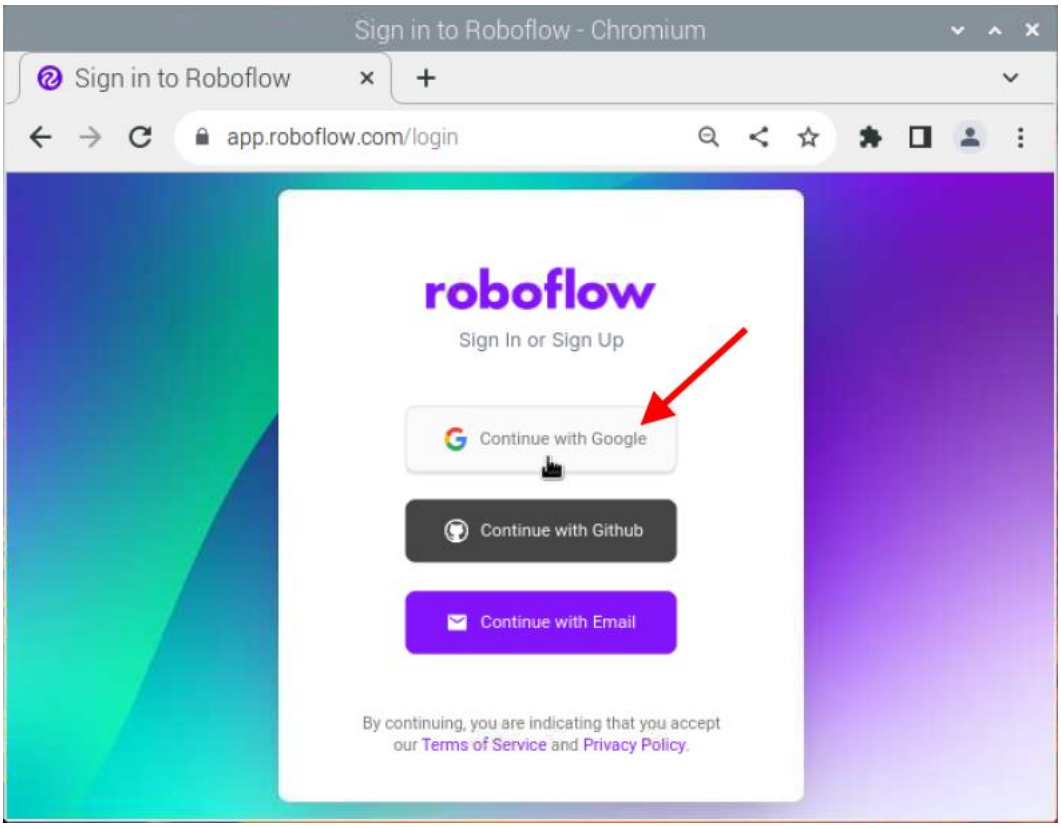
2.2 พิมพ์คำค้นหาว่า “roboflow” จากนั้นคลิกเข้าเว็บไซต์ https://roboflow.com



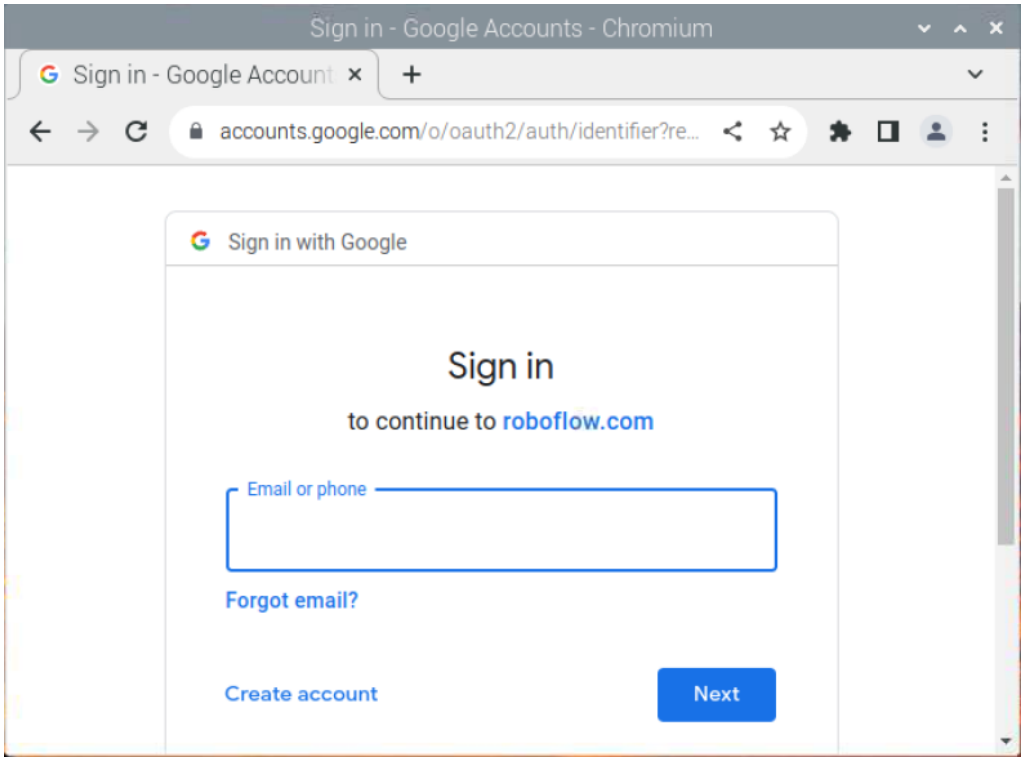
2.3 คลิกปุ่ม Get Started -->



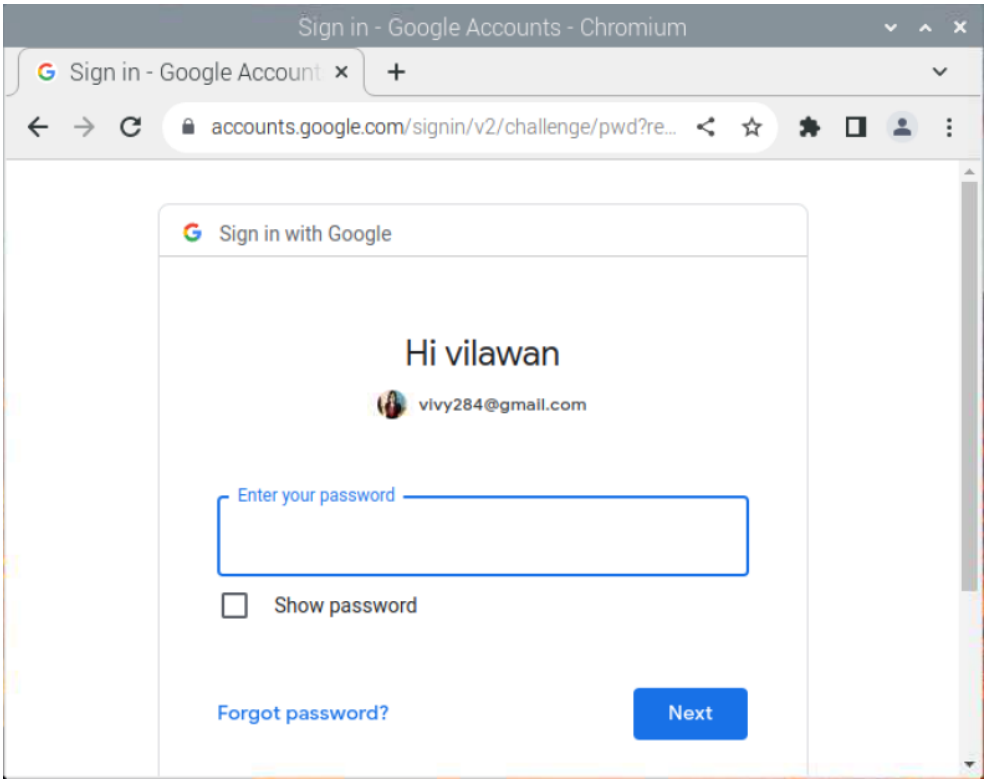
2.4 เลือกล็อกอินด้วยบัญชี Gmail โดยคลิกปุ่ม Continue with Google



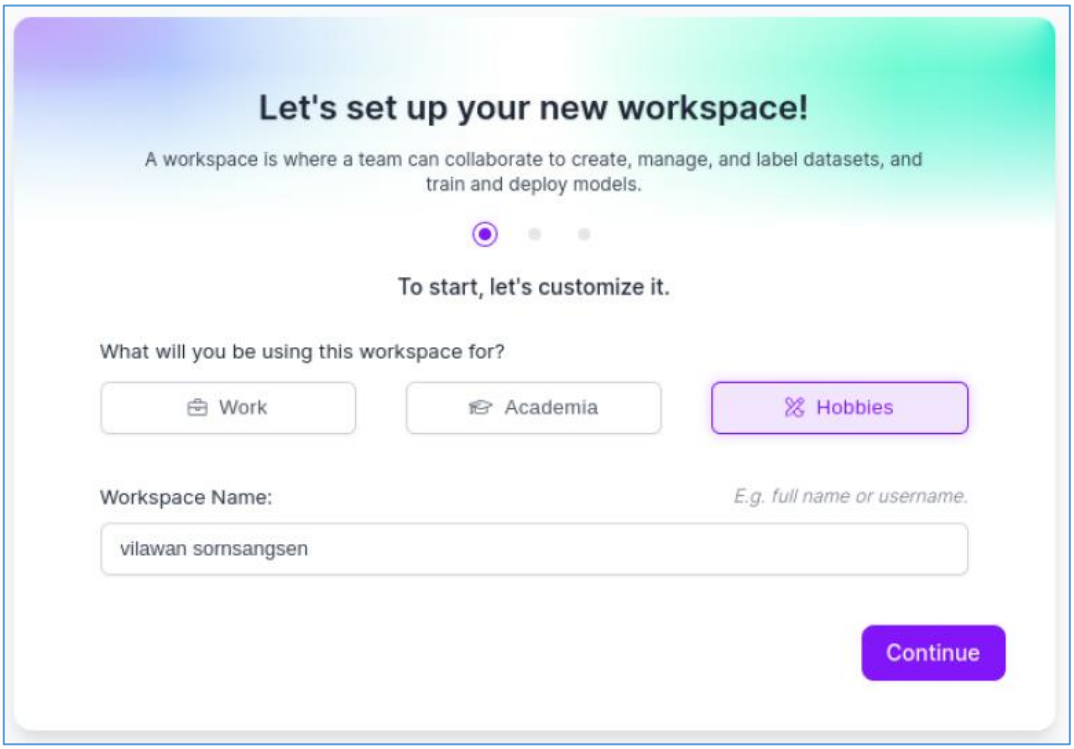
2.5 กรอกอีเมล แล้วคลิกปุ่ม Next



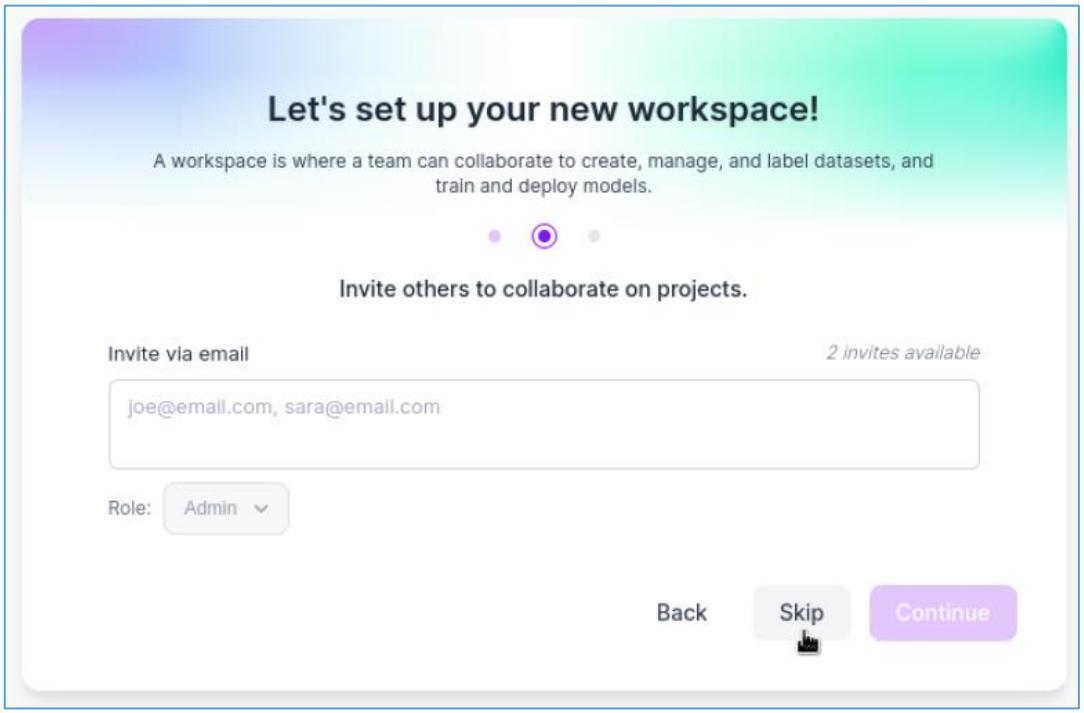
2.6 กรอกรหัสผ่าน แล้วคลิกปุ่ม Next



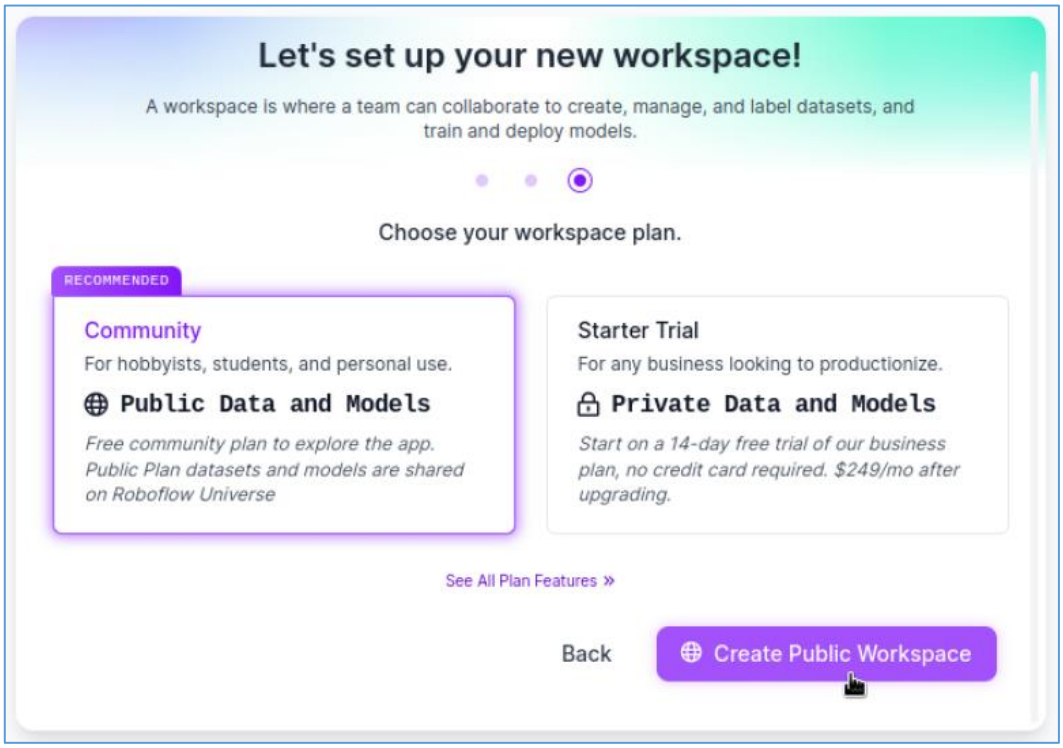
2.7 เลือกวัตถุประสงค์ในการใช้งาน จากนั้นคลิกปุ่ม Continue



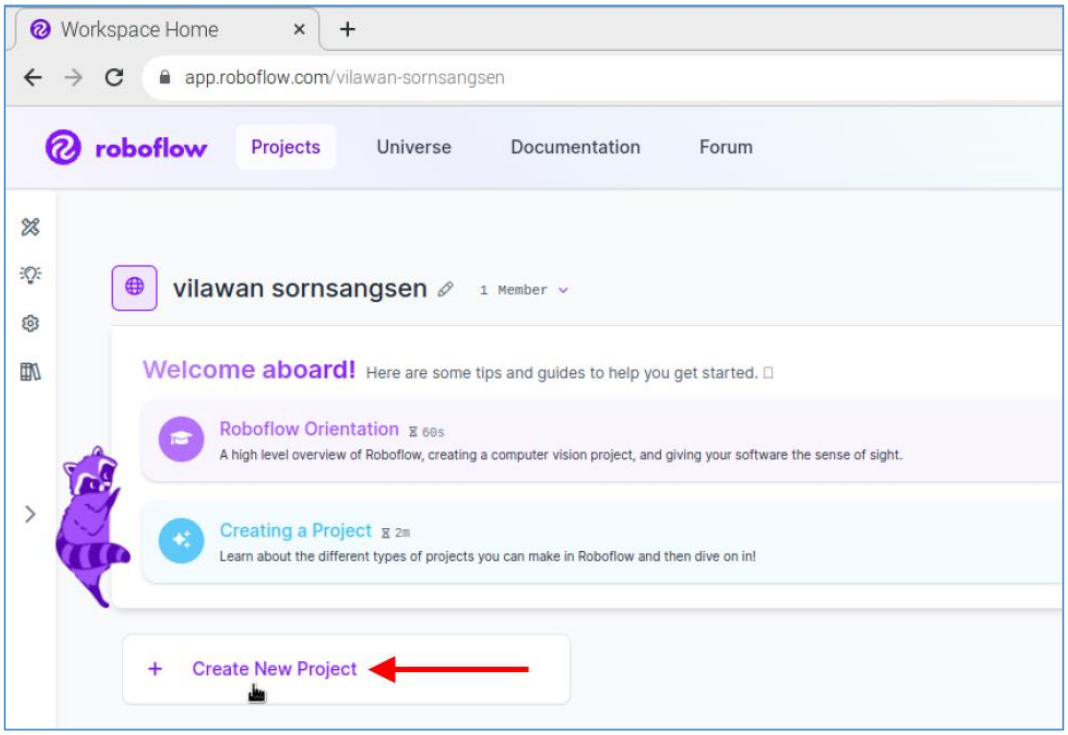
2.8 คลิก Skip เพื่อข้ามขั้นตอนการเชิญผู้อื่นมาร่วมใช้งาน



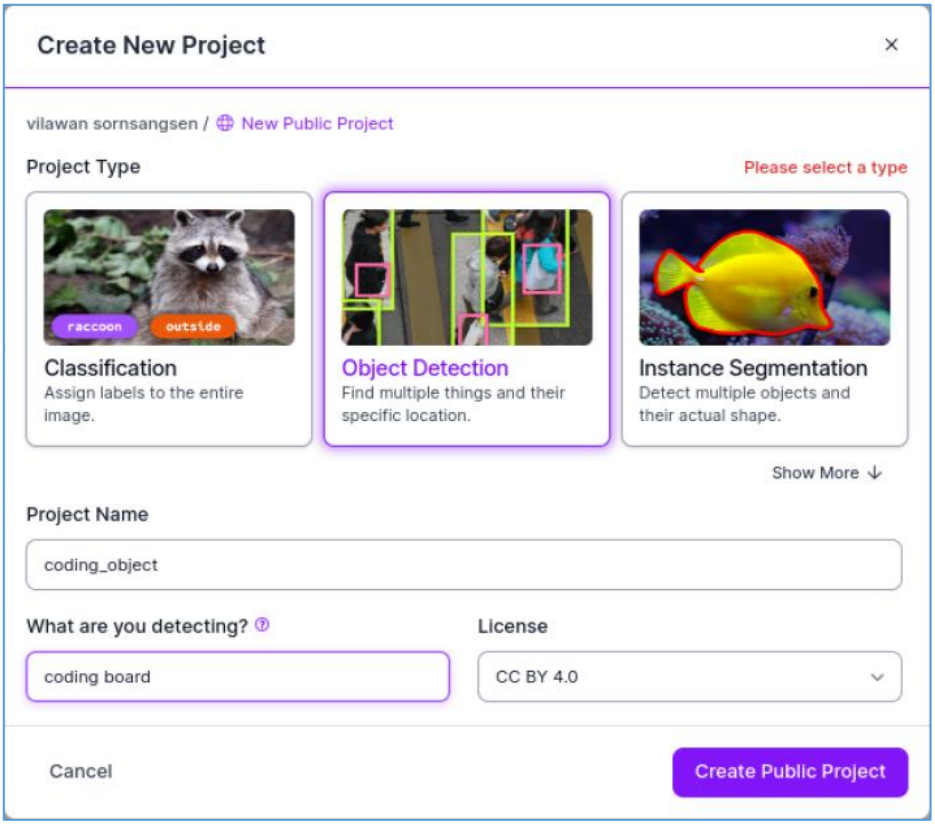
2.9 คลิกเลือก Public Data and Models จากนั้นคลิกปุ่ม Create Public Workspace



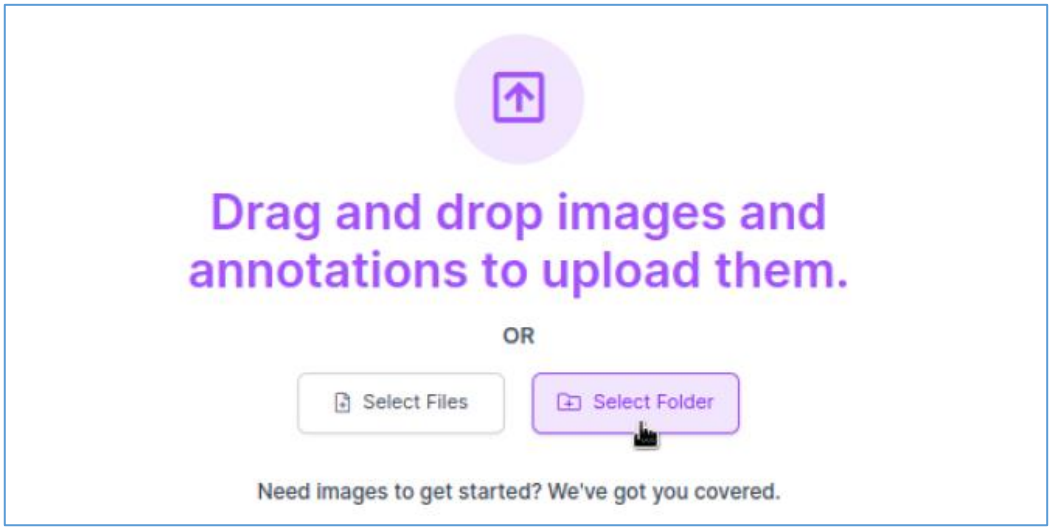
2.10 คลิกสร้างโปรเจ็กต์ใหม่ โดยคลิกที่ Create New Project



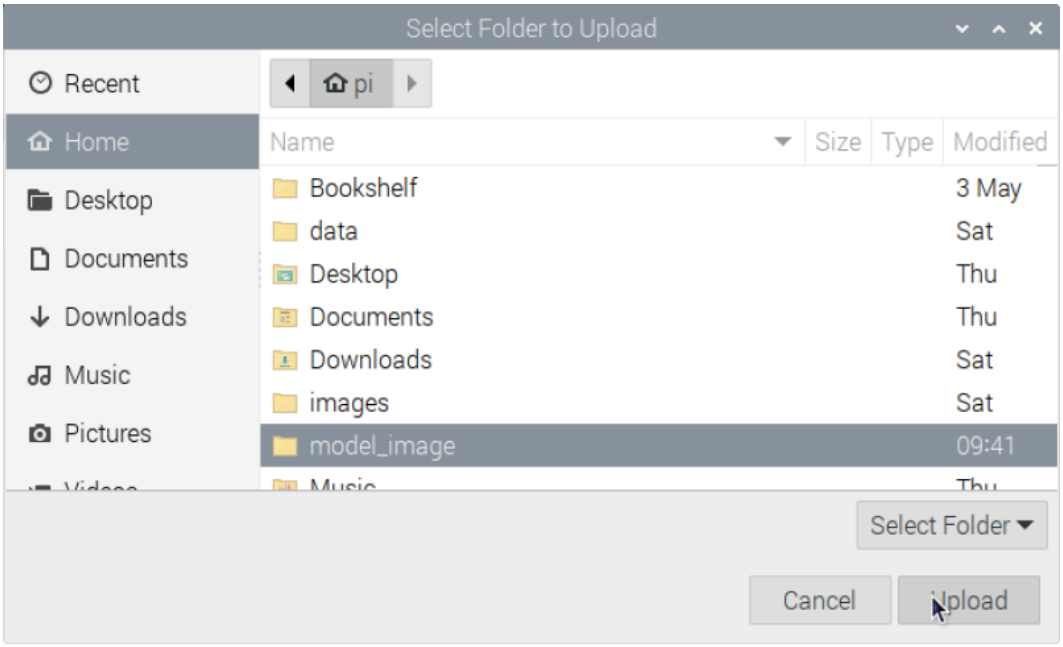
2.11 เลือก Project Type เป็นแบบ Object Detection ตั้งชื่อ Project Name และระบุต้องการตรวจจับสิ่งใด จากนั้นคลิกปุ่ม Create Public Project



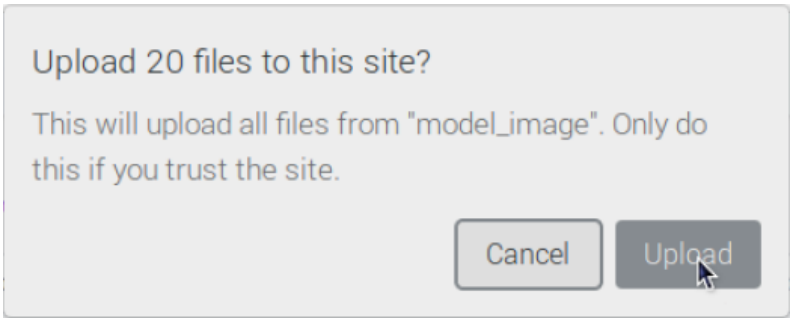
2.12 อัพโหลดโฟลเดอร์รูปภาพ โดยคลิก Select Folder



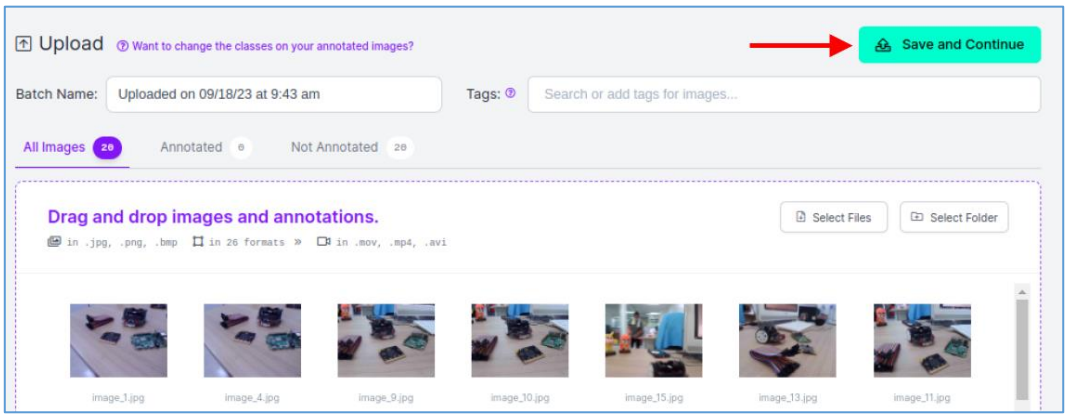
2.13 เลือกโฟลเดอร์ที่เก็บรูปภาพ จากนั้นคลิกปุ่ม Upload



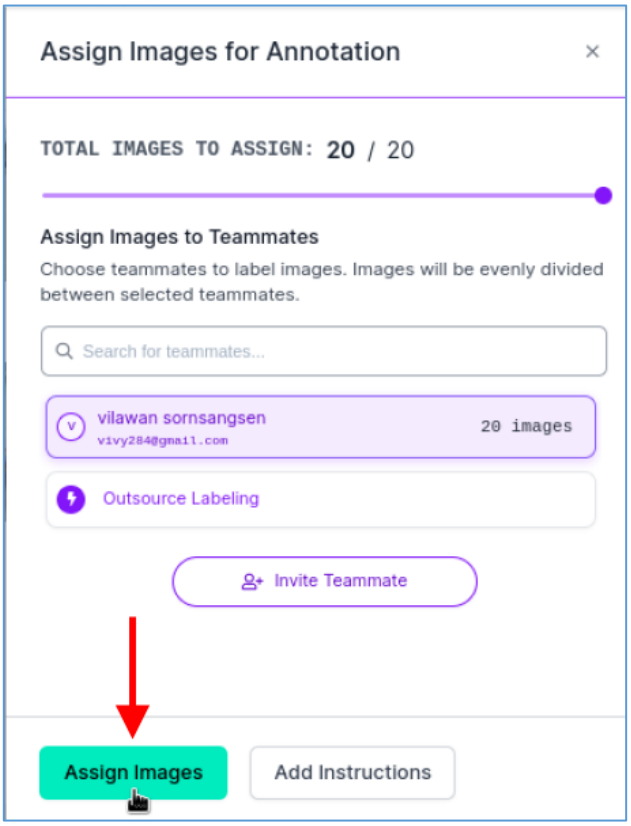
2.14 คลิกปุ่ม Upload



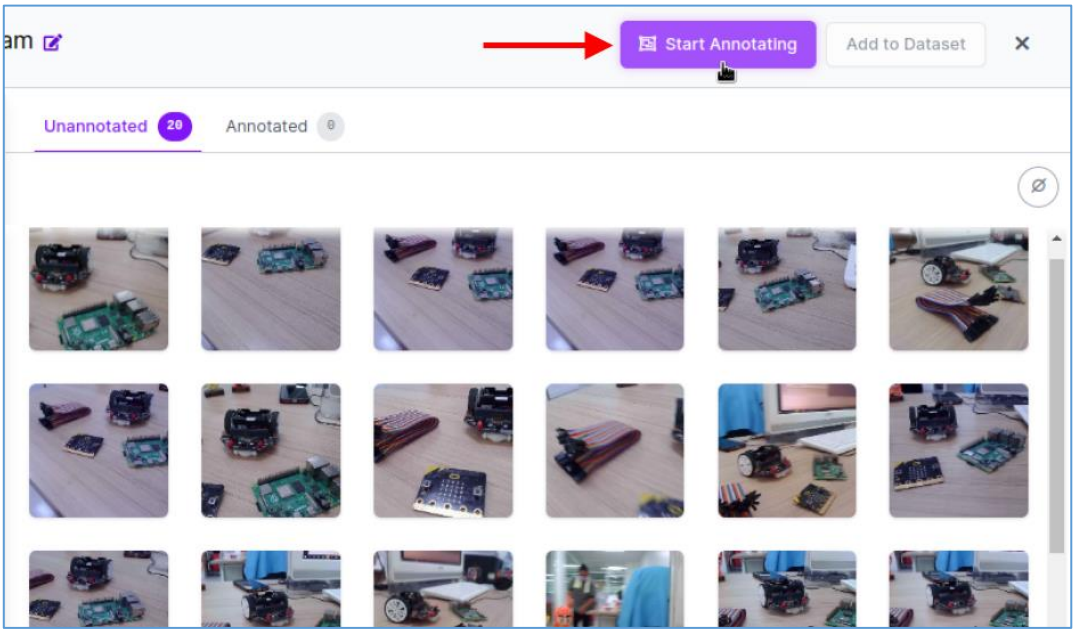
2.15 คลิกปุ่ม Save and Continue



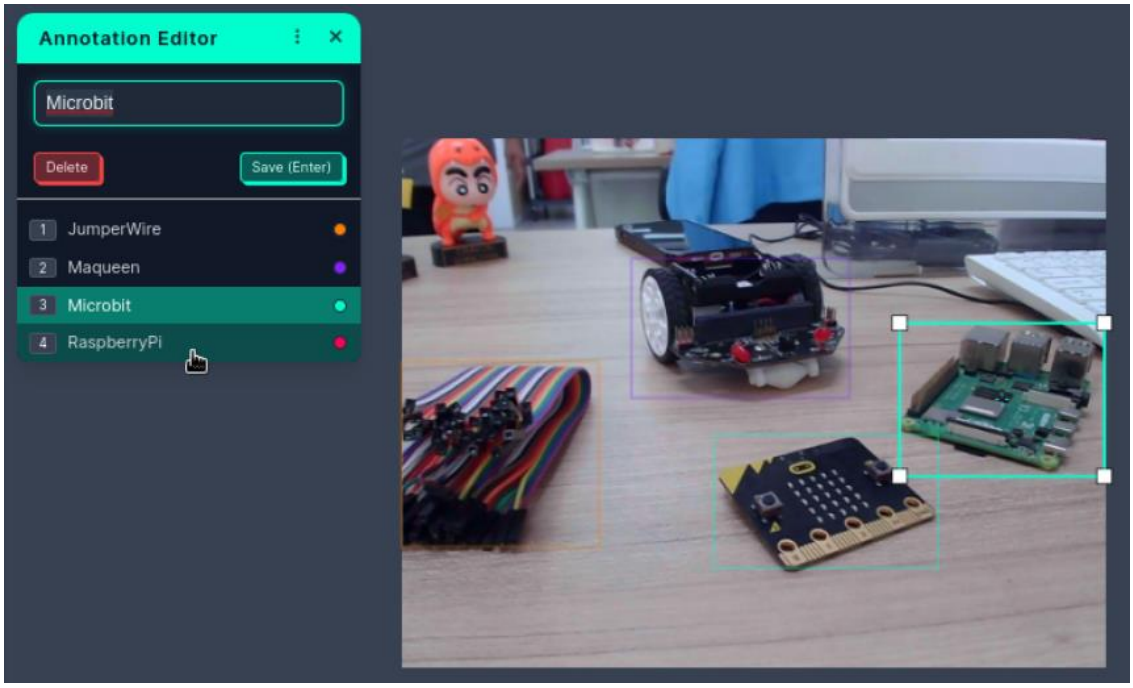
2.16 คลิกปุ่ม Assign Images



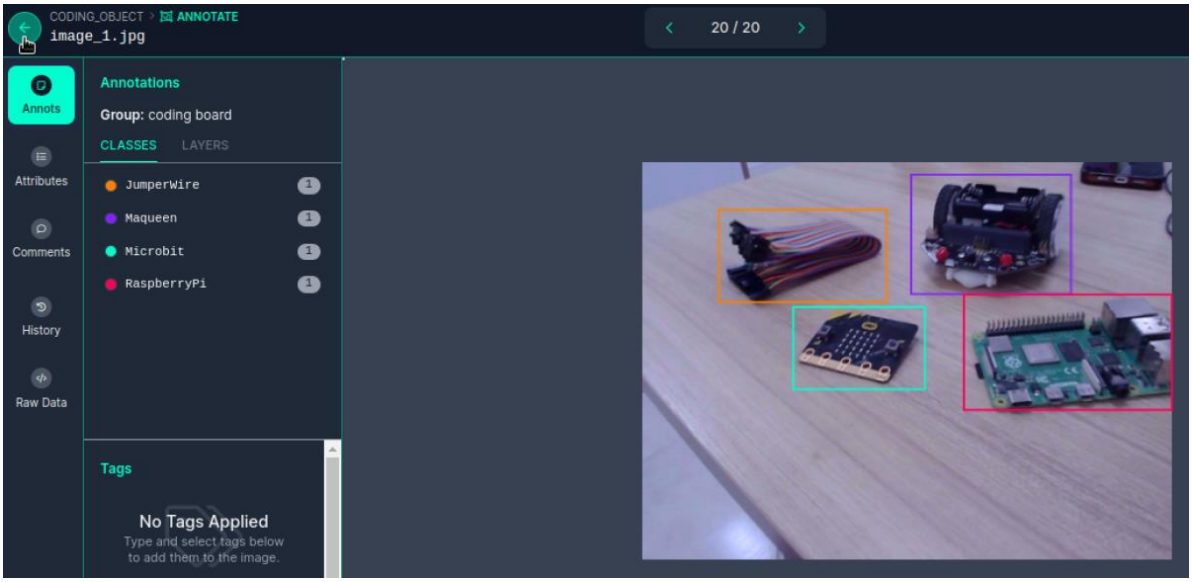
2.17 เริ่มต้นการสร้างป้ายกำกับ โดยคลิกปุ่ม Start Annotating



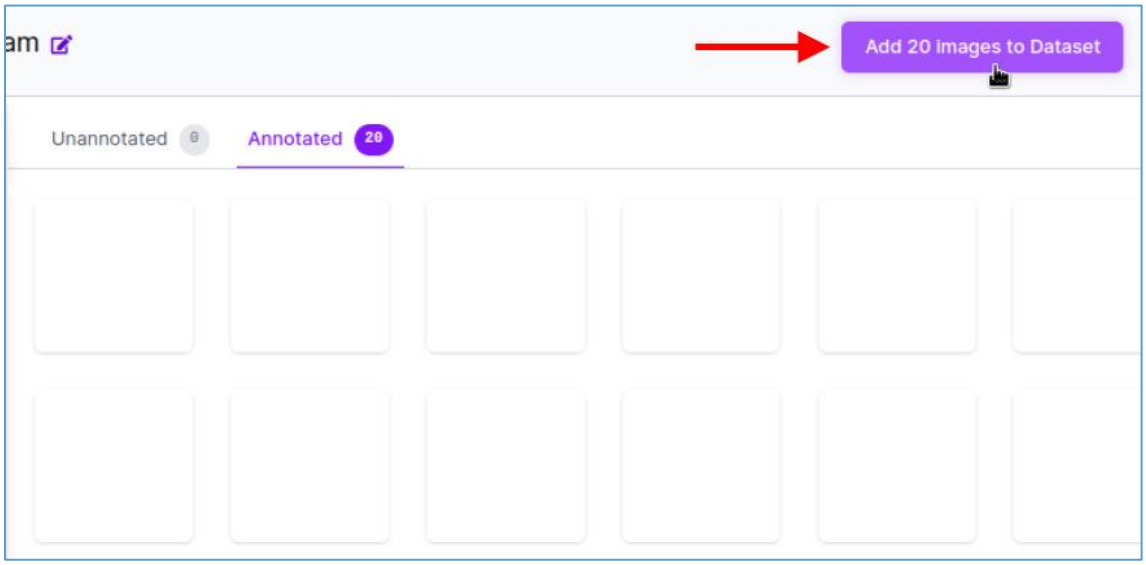
2.18 วาดกรอบล้อมรอบวัตถุ จากนั้นใส่ป้ายกำกับวัตถุลงไป



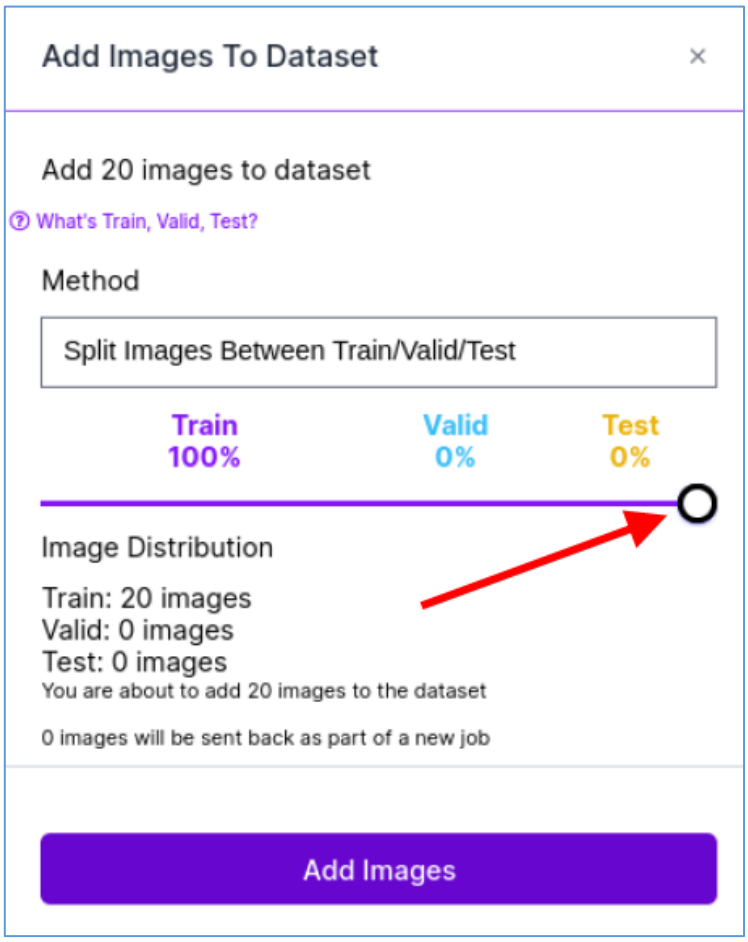
2.19 คลิกปุ่มลูกศร > เพื่อเลื่อนรูป สร้างป้ายกำกับให้ครบทุกรูป เสร็จแล้วคลิกปุ่ม <-- ย้อนกลับ



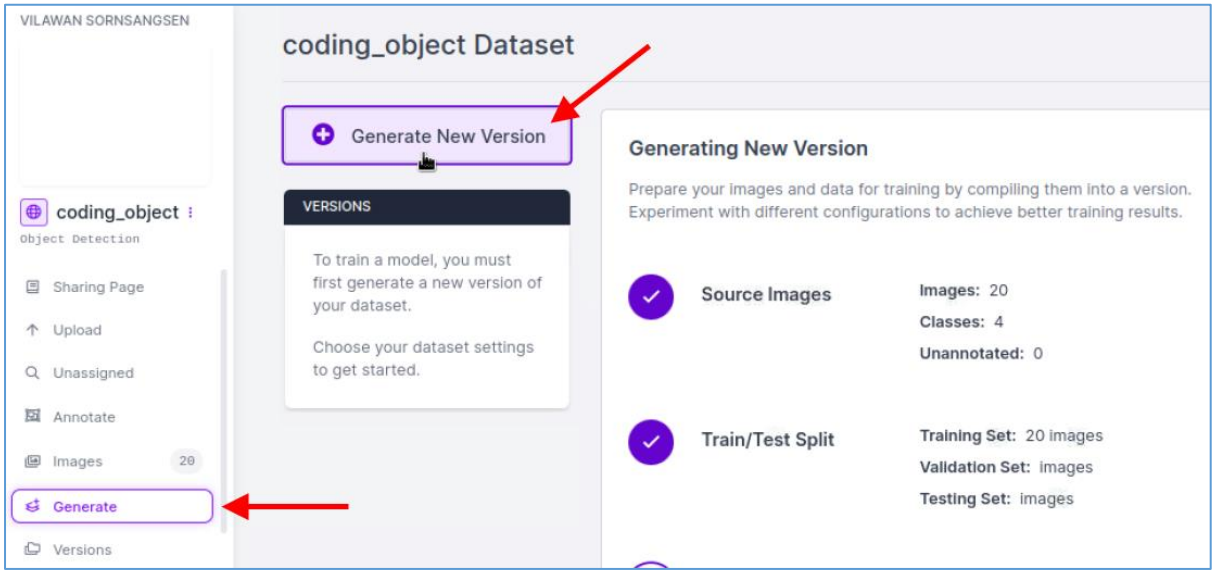
2.20 คลิกปุ่ม Add … images to Dataset



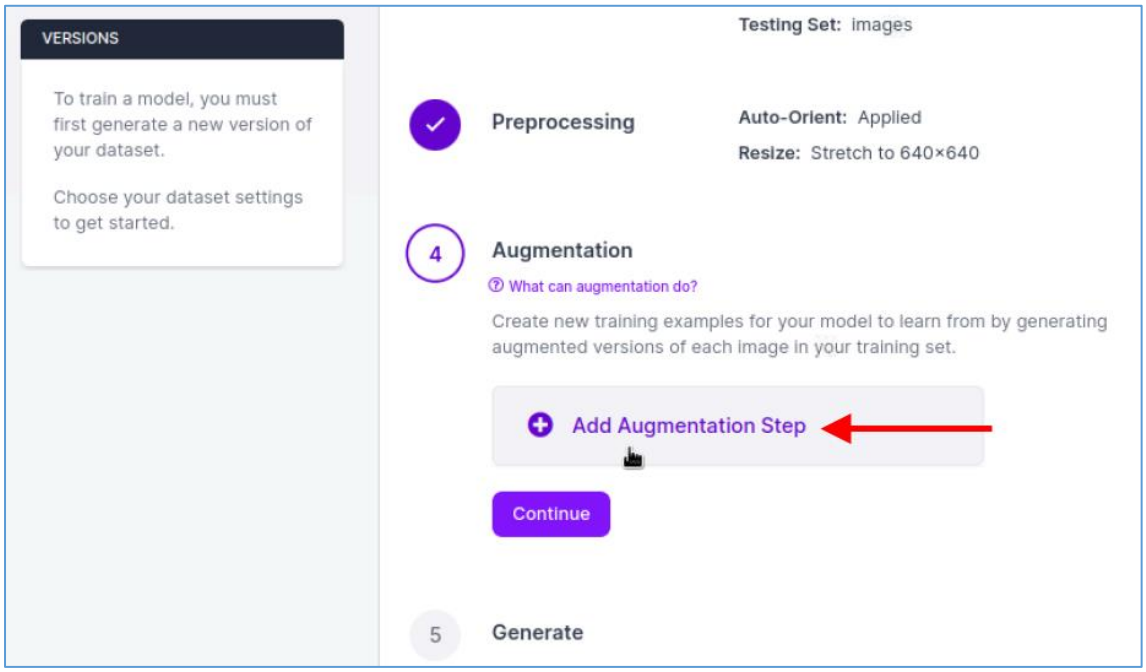
2.21 ปรับสัดส่วน Dataset โดยเลื่อนรูปภาพสำหรับการ Train เป็น 100% แล้วคลิกปุ่ม Add Images



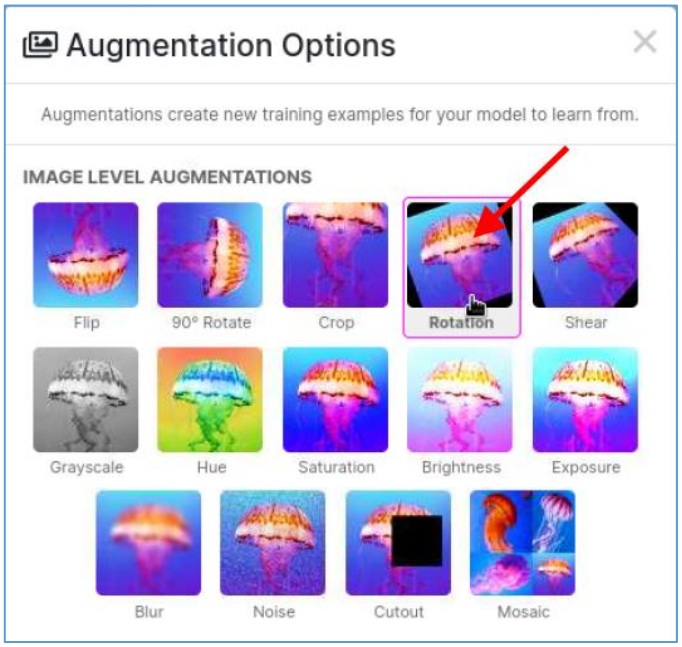
2.22 ขยายรูปภาพสำหรับการ Train ให้มากขึ้นเป็น 3 เท่า โดยคลิกแท็บ Generate ที่อยู่ด้านซ้ายมือ จากนั้นคลิกปุ่ม Generate New Version



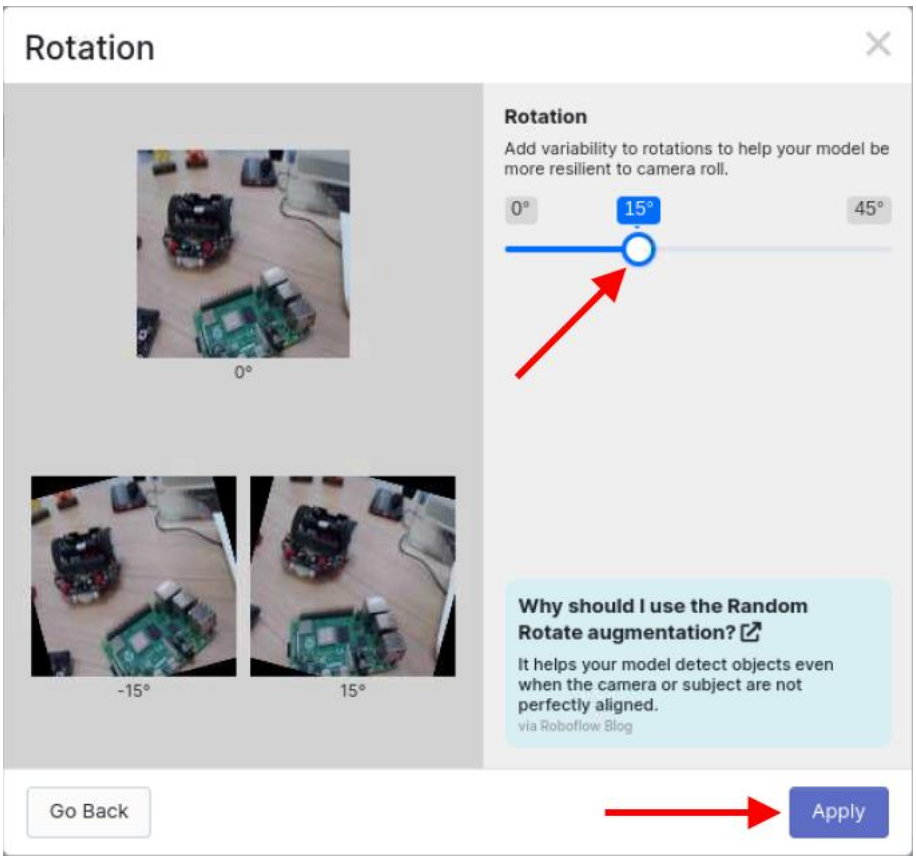
2.23 คลิกเลือกรูปแบบการเปลี่ยนแปลงรูปภาพ โดยคลิกที่ Add Augmentation Step



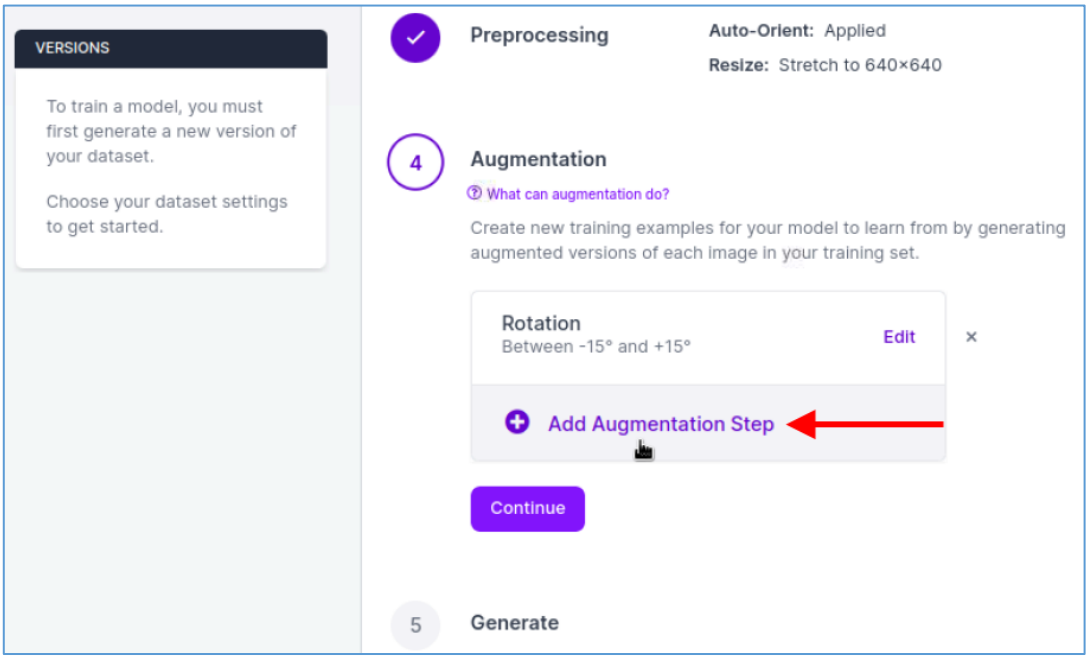
2.24 เลือกรูปแบบที่ต้องการ ในที่นี้จะเลือกการหมุนรูปภาพด้วย Rotation



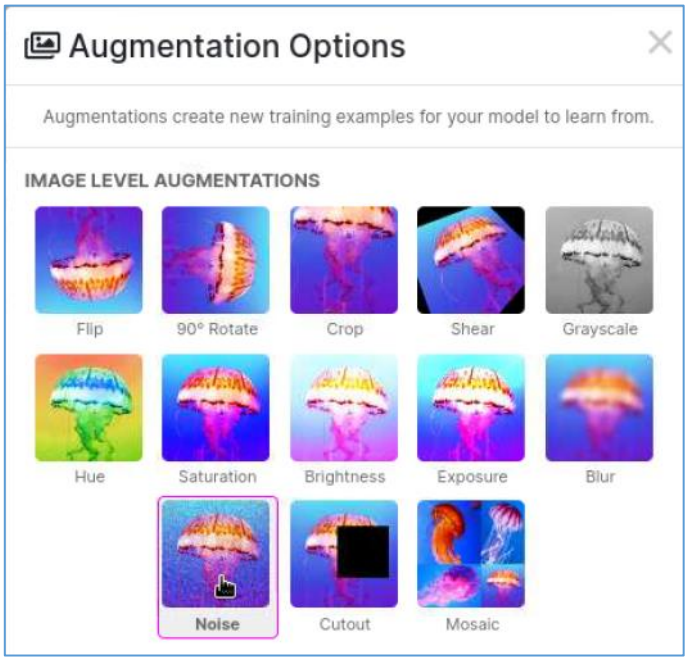
2.25 เลื่อนองศาในการหมุน โดยดูตัวอย่างได้จากรูปภาพด้านซ้ายมือ จากนั้นคลิกปุ่ม Apply



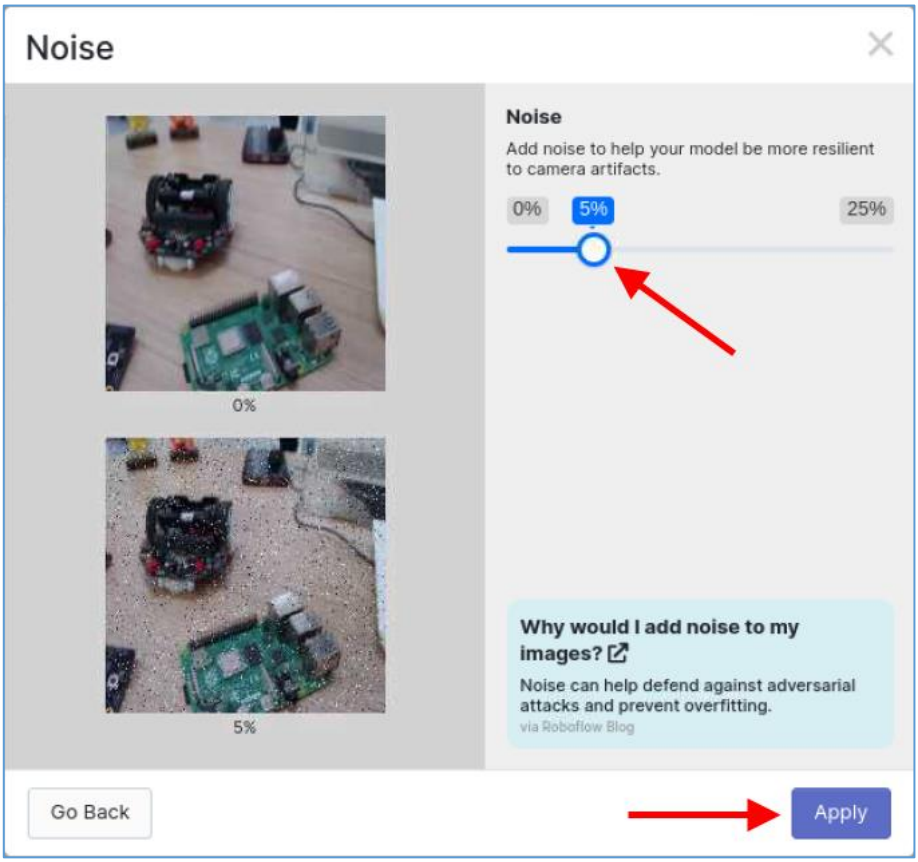
2.26 เพิ่มรูปแบบการเปลี่ยนแปลงรูปภาพ โดยคลิกที่ Add Augmentation Step



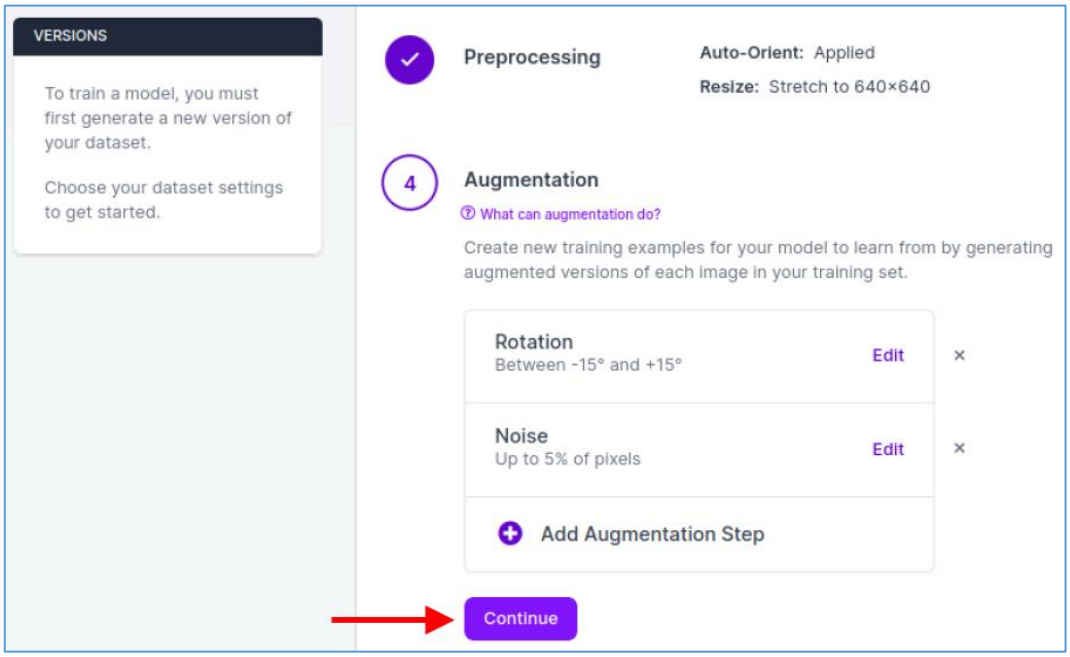
2.27 คลิกเลือกรูปแบบที่ต้องการ ในที่นี้จะเพิ่ม Noise เข้าไป



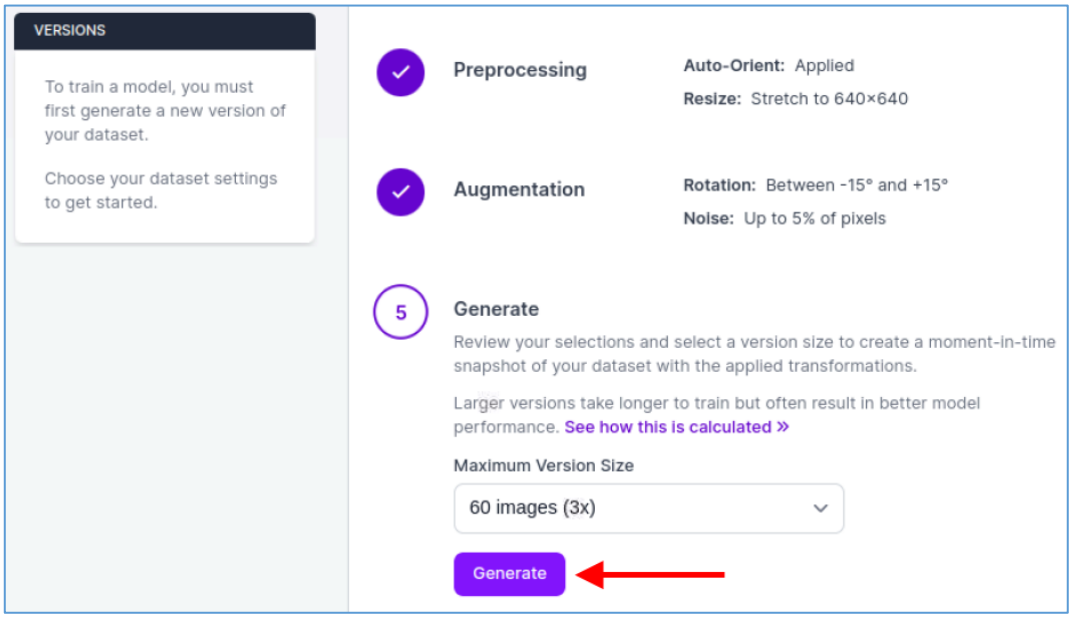
2.28 ปรับค่า Noise จากนั้นคลิกปุ่ม Apply



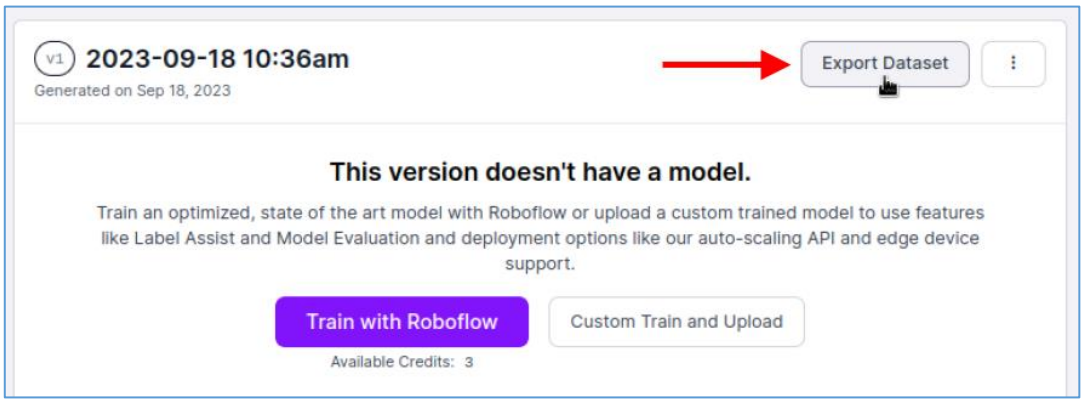
2.29 คลิกปุ่ม Continue



2.30 จะเห็นว่าระบบจะสร้างรูปภาพเพิ่มขึ้นมา 3 เท่าจากเดิม จากนั้นคลิกปุ่ม Generate



2.31 ส่งออกรูปภาพ Dataset โดยคลิกปุ่ม Export Dataset

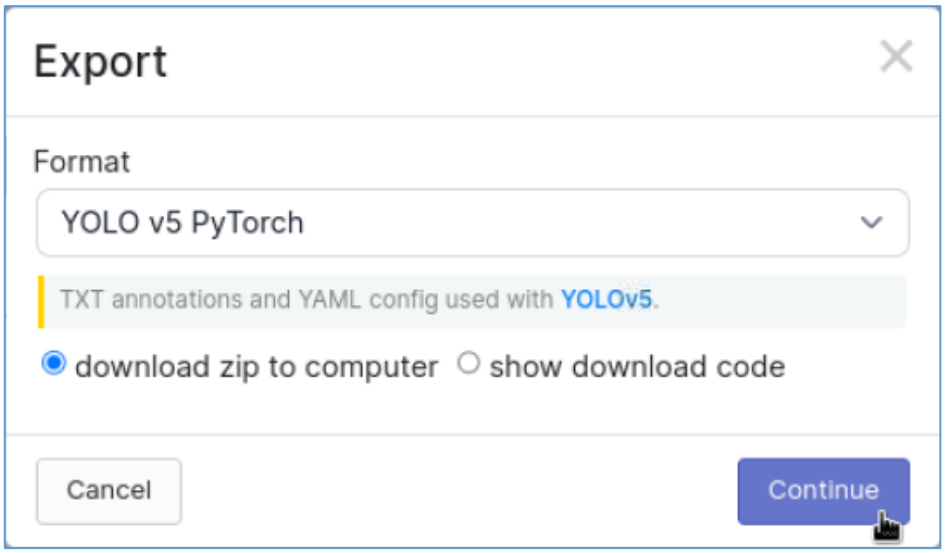


2.32 เลือกรูปแบบการดาวน์โหลด ดังนี้

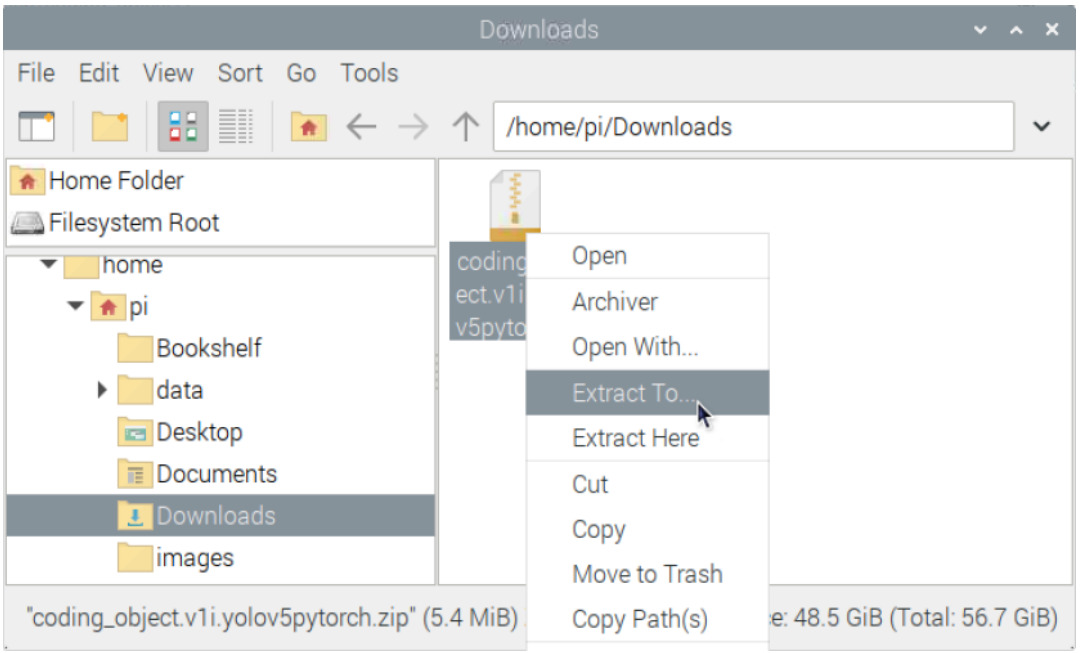
- Format: YOLO v5 PyTorch

- download zip to computer

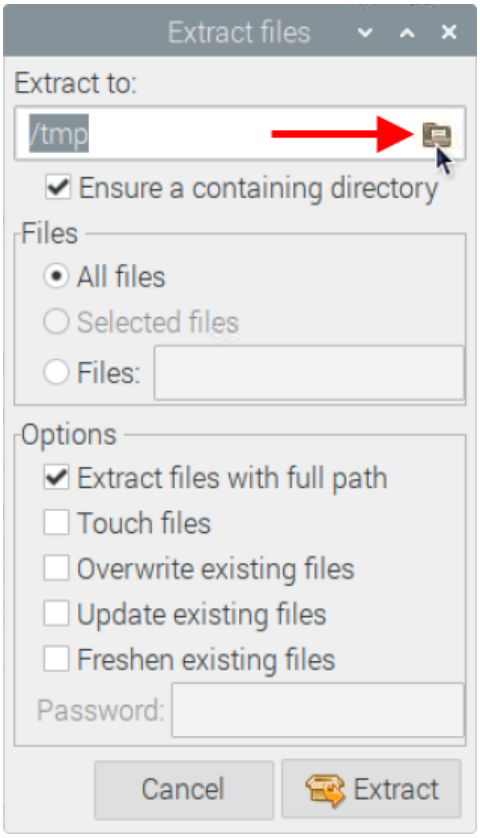
จากนั้นคลิกปุ่ม Continue



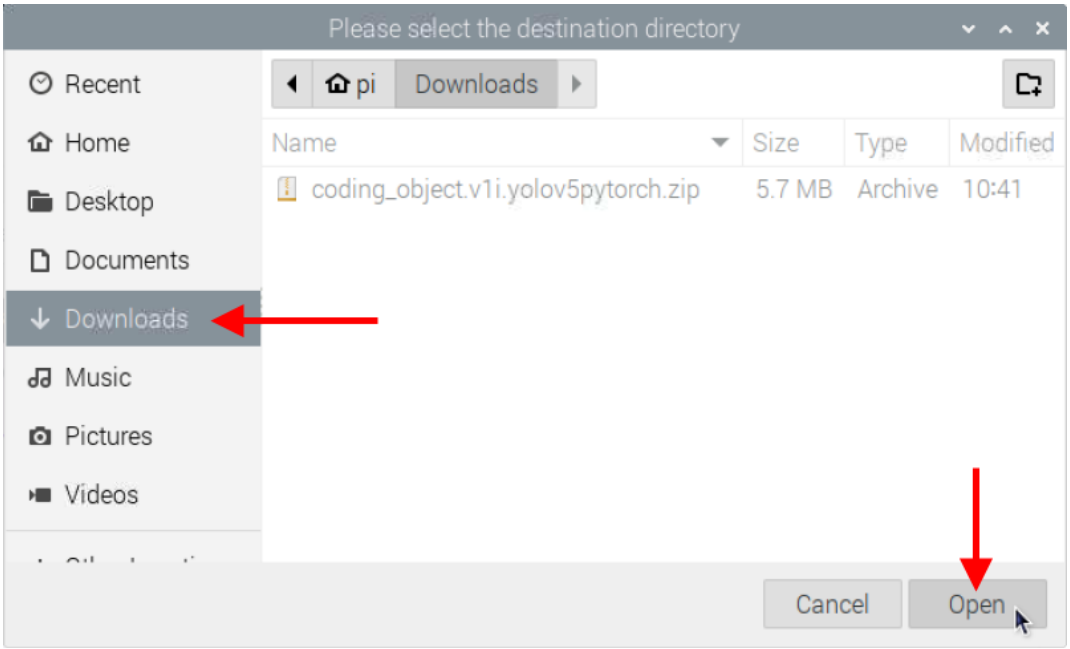
2.33 ไฟล์จะถูกดาวน์โหลดเข้ามาในโฟลเดอร์ /home/pi/Downloads จากนั้นให้คลิกขวาที่ไฟล์เลือกคำสั่ง Extract To….



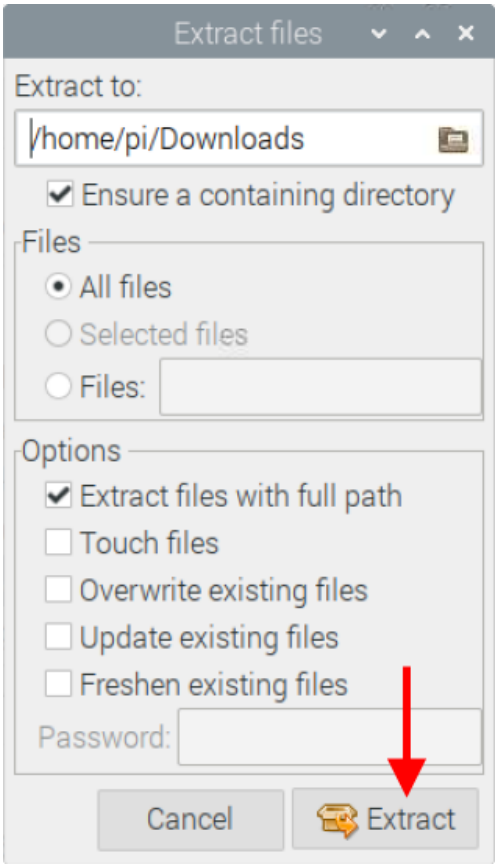
2.34 คลิกเลือกโฟลเดอร์ปลายทาง



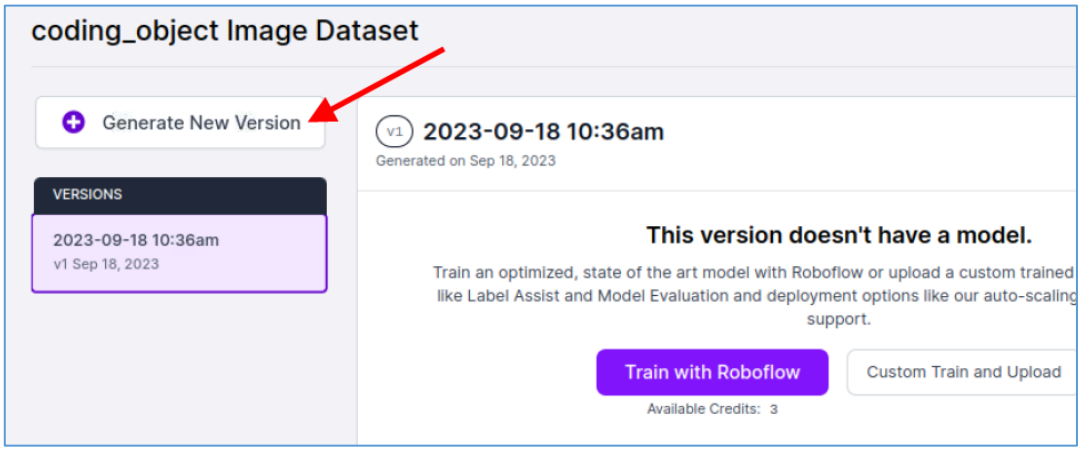
2.35 คลิกเลือกโฟลเดอร์ Downloads จากนั้นคลิกปุ่ม Open



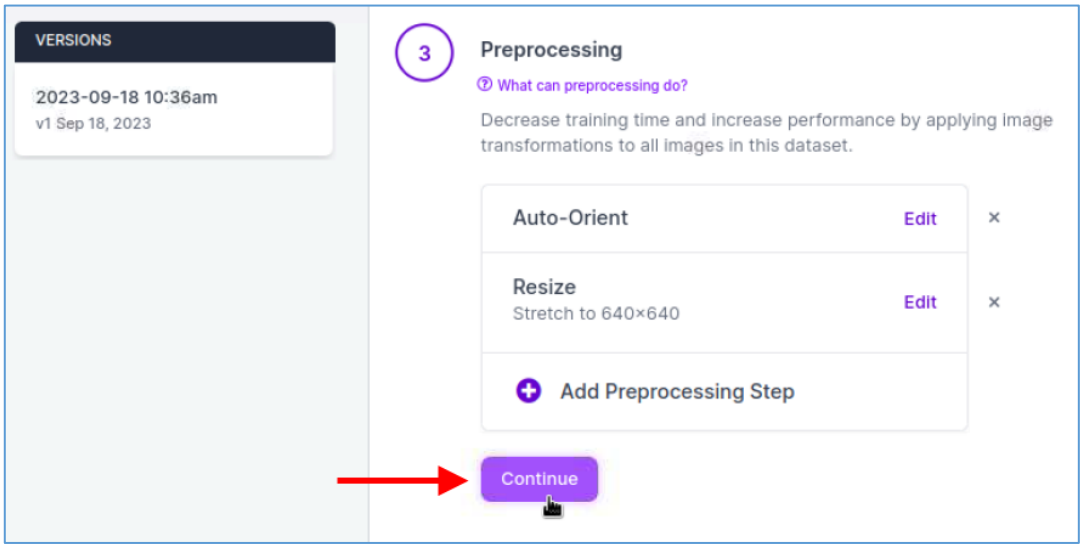
2.36 คลิกปุ่ม Extract



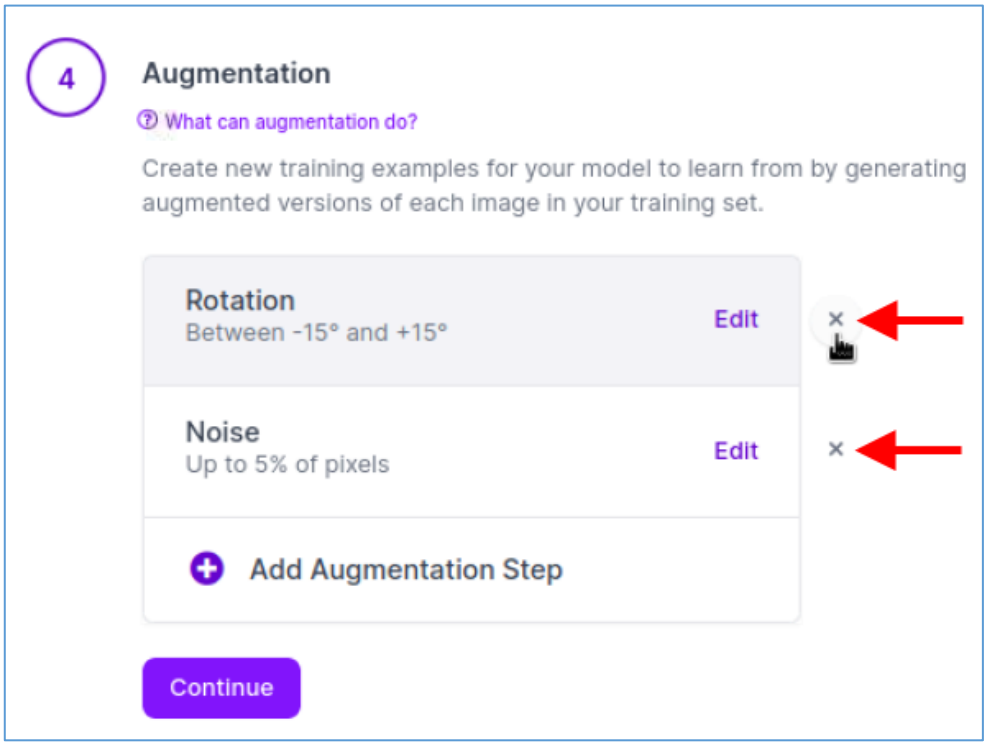
2.37 หากต้องการขยายรูปภาพในการ Train เพิ่ม สามารถคลิกที่ Generate New Version อีกครั้ง



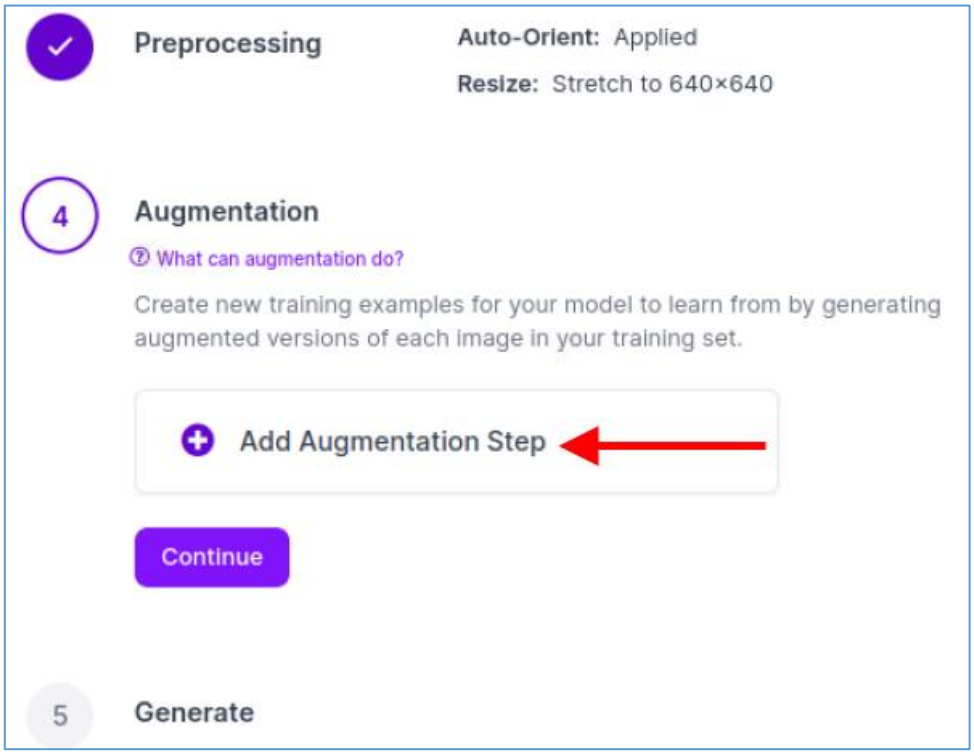
2.38 คลิกปุ่ม Continue



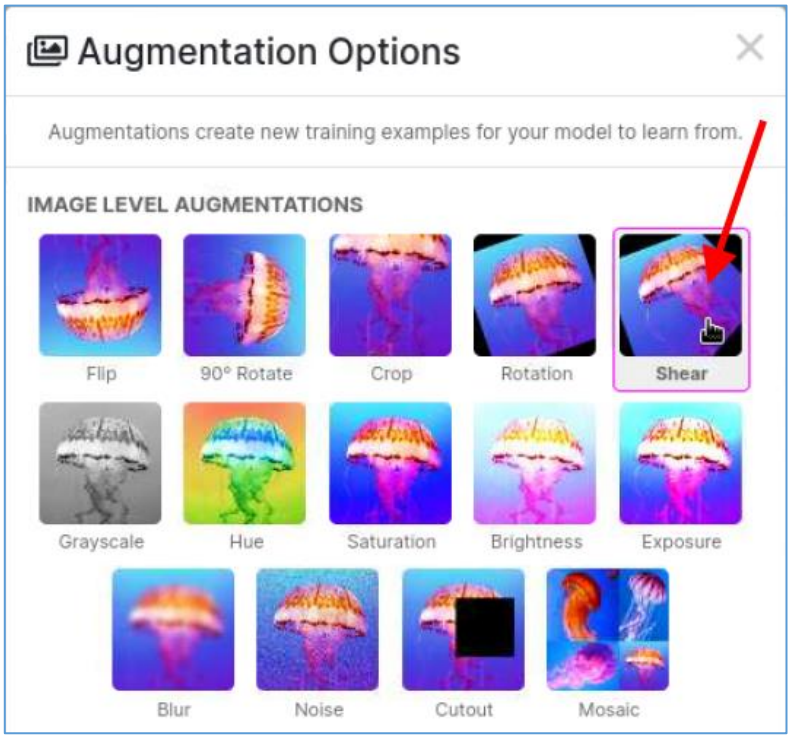
2.39 ลบรูปแบบ Augmentation เดิมทิ้งไป



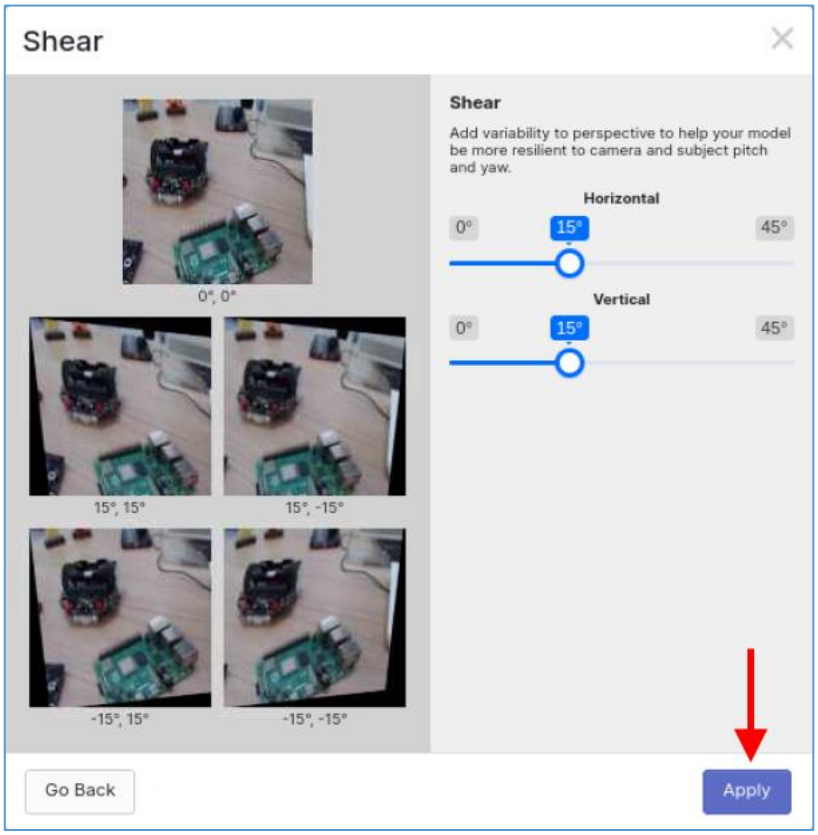
2.40 คลิก Add Augmentation Step



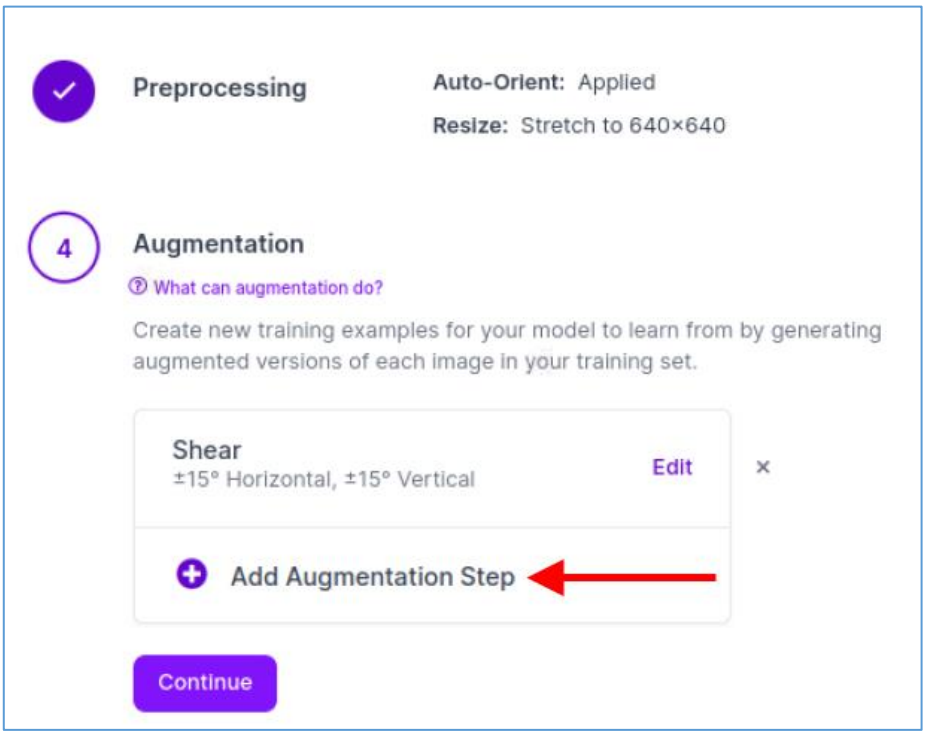
2.41 คลิกเลือกรูปแบบการเปลี่ยนแปลงรูปภาพ ในที่นี้เลือกเอียงรูปภาพ โดยเลือก Shear



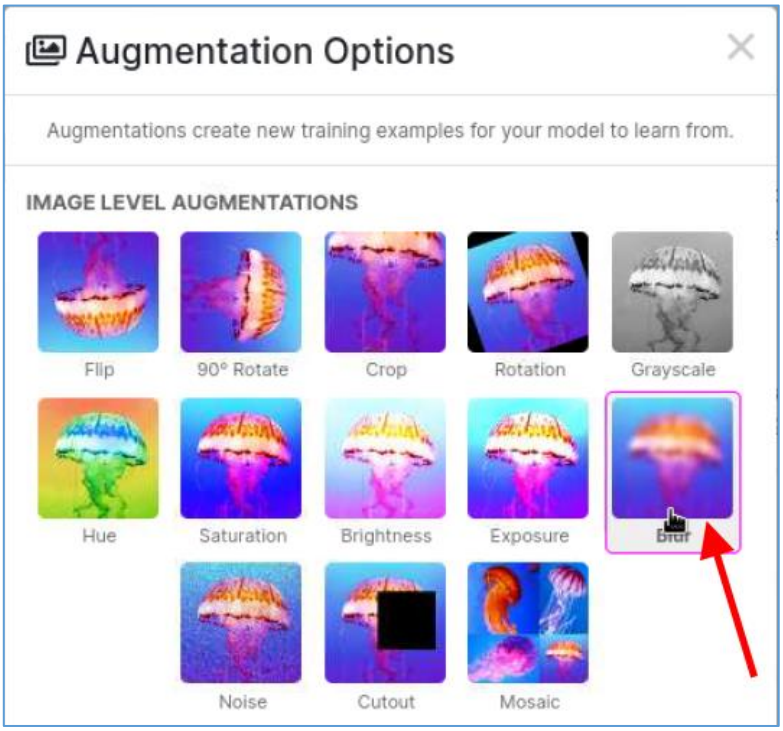
2.42 ปรับการเอียงรูปภาพในแนวตั้งและแนวนอน จากนั้นคลิกปุ่ม Apply



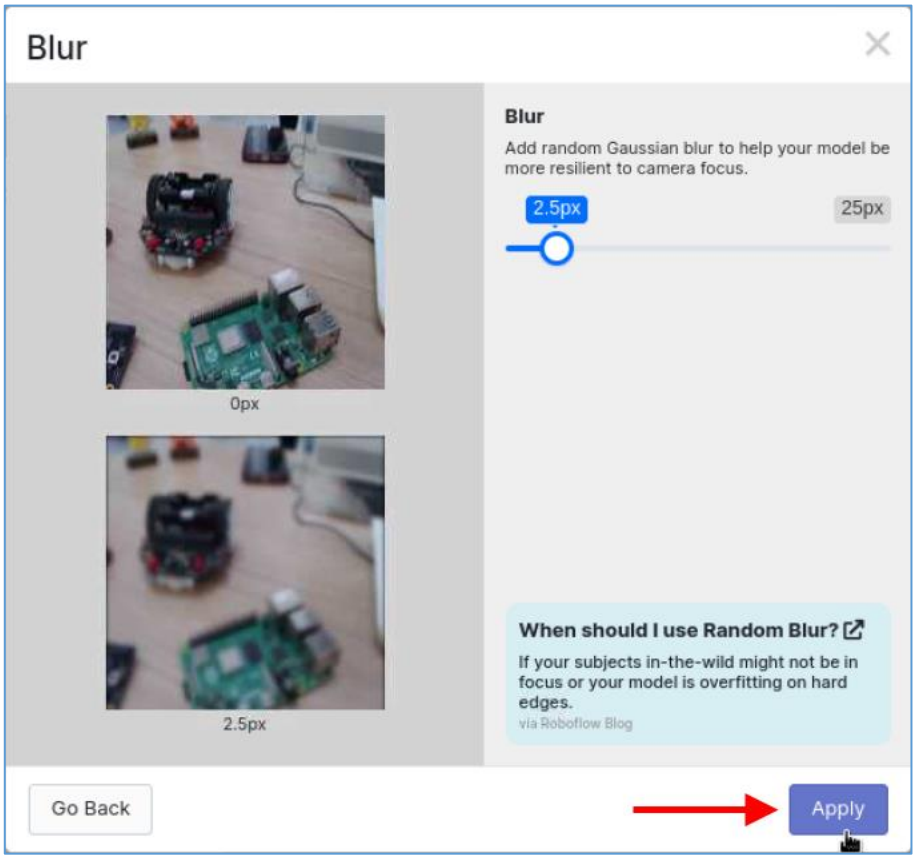
2.43 หากต้องการเพิ่ม Augmentation ให้คลิก Add Augmentation Step



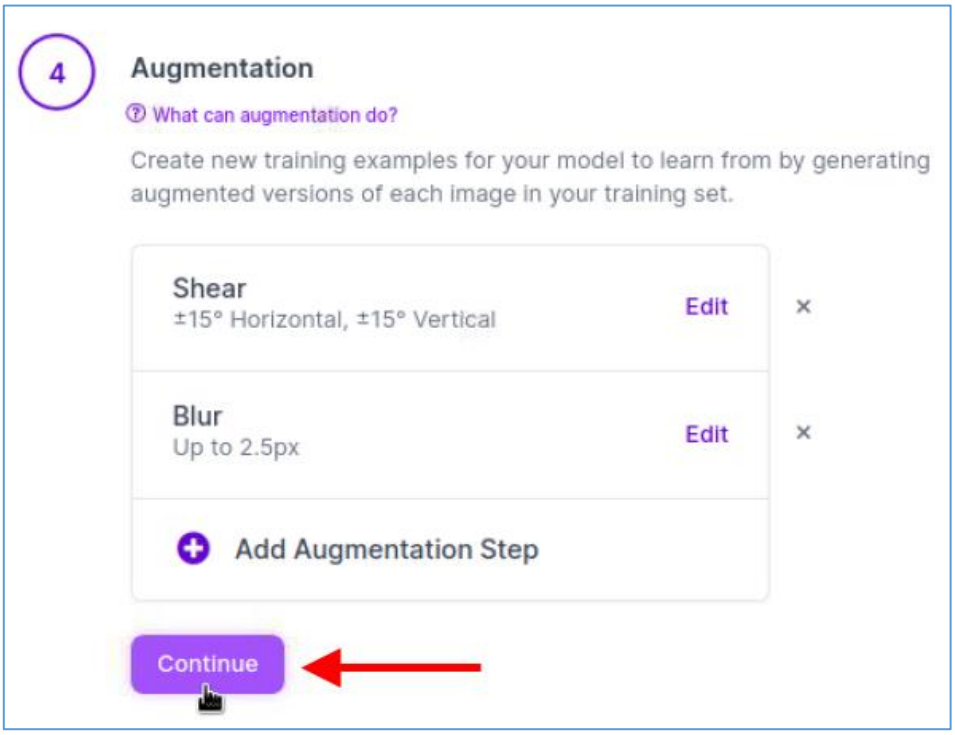
2.44 ในที่นี้จะเลือกเป็นแบบ Blur



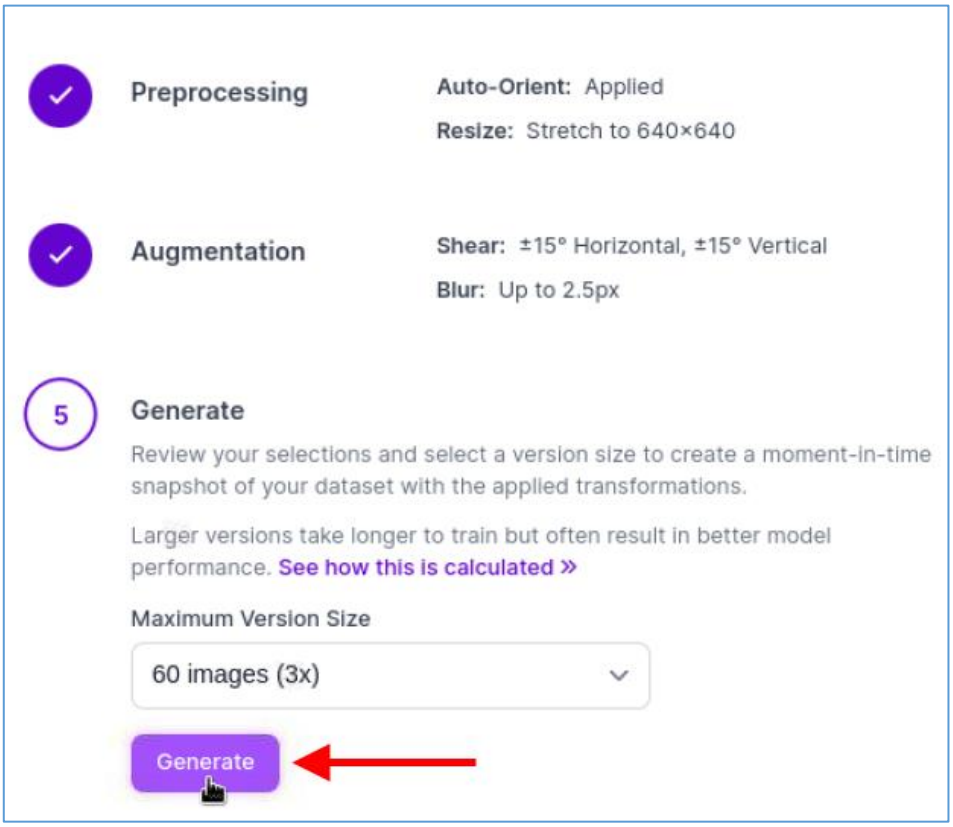
2.45 ปรับการเบลอของภาพ จากนั้นคลิกปุ่ม Apply



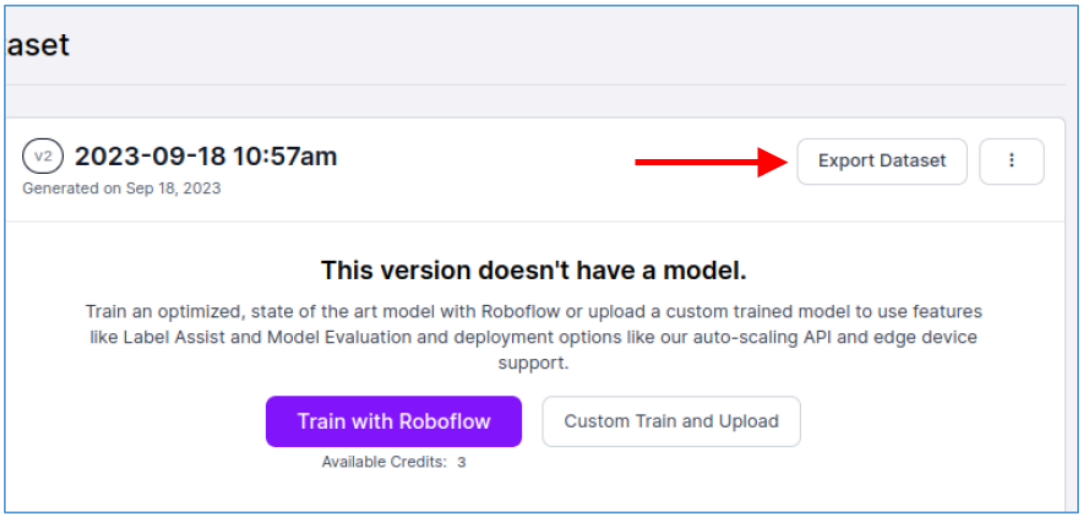
2.46 คลิกปุ่ม Continue



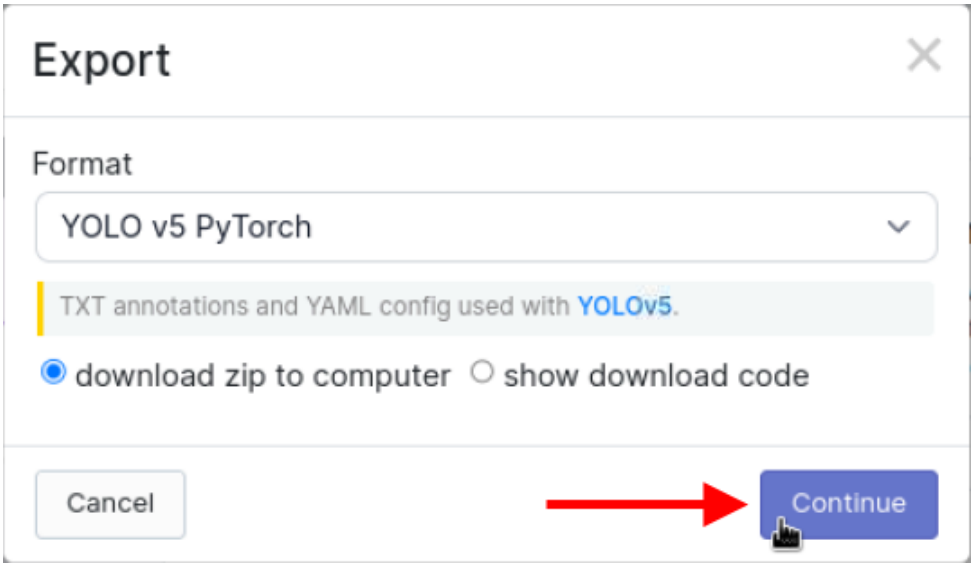
2.47 คลิกปุ่ม Generate



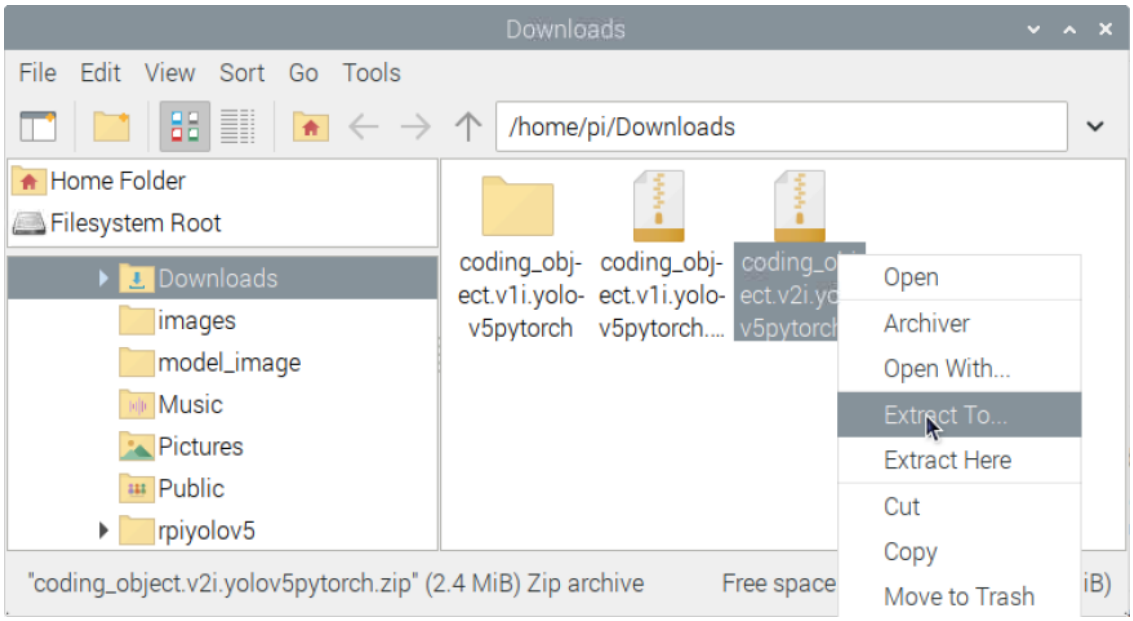
2.48 ส่งออก Dataset โดยคลิกปุ่ม Export Dataset



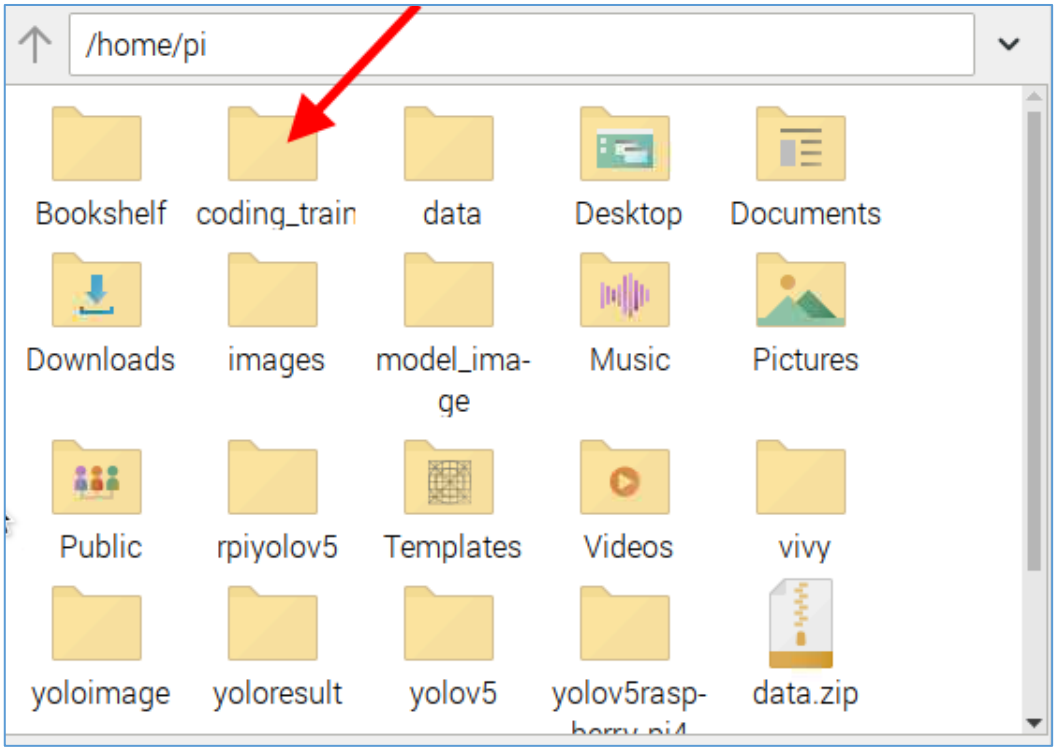
2.49 เลือกรูปแบบการส่งออกเหมือนครั้งแรก จากนั้นคลิกปุ่ม Continue



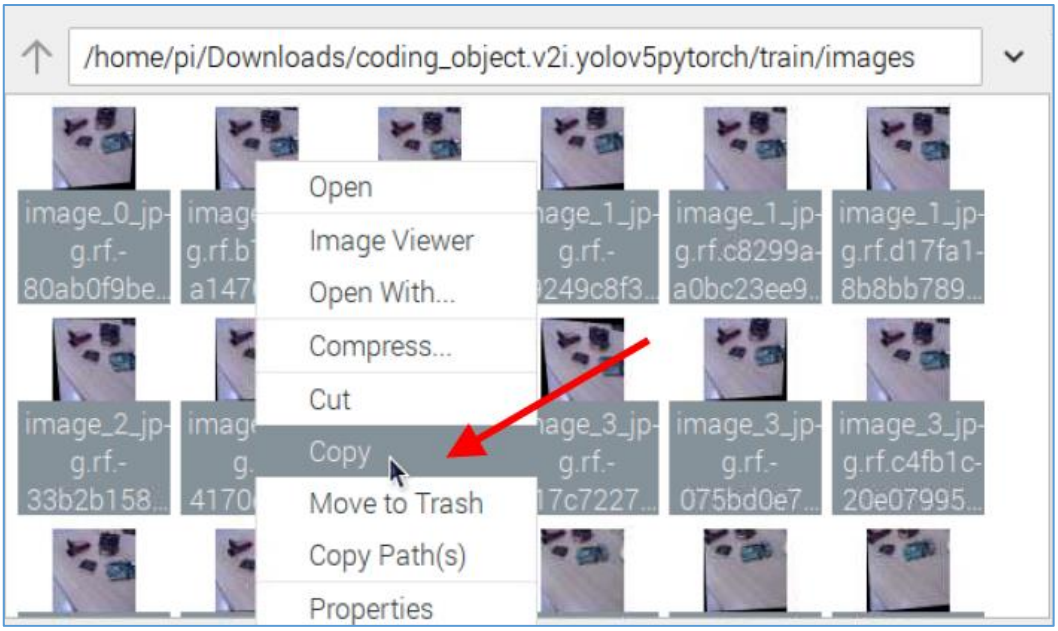
2.50 ไฟล์ที่สองจะถูกดาวน์โหลดมาไว้ในโฟลเดอร์ Downloads เช่นเดียวกับไฟล์แรก ให้ทำการแตกไฟล์เช่นเดียวกัน โดยคลิกขวาเลือก Extract To…



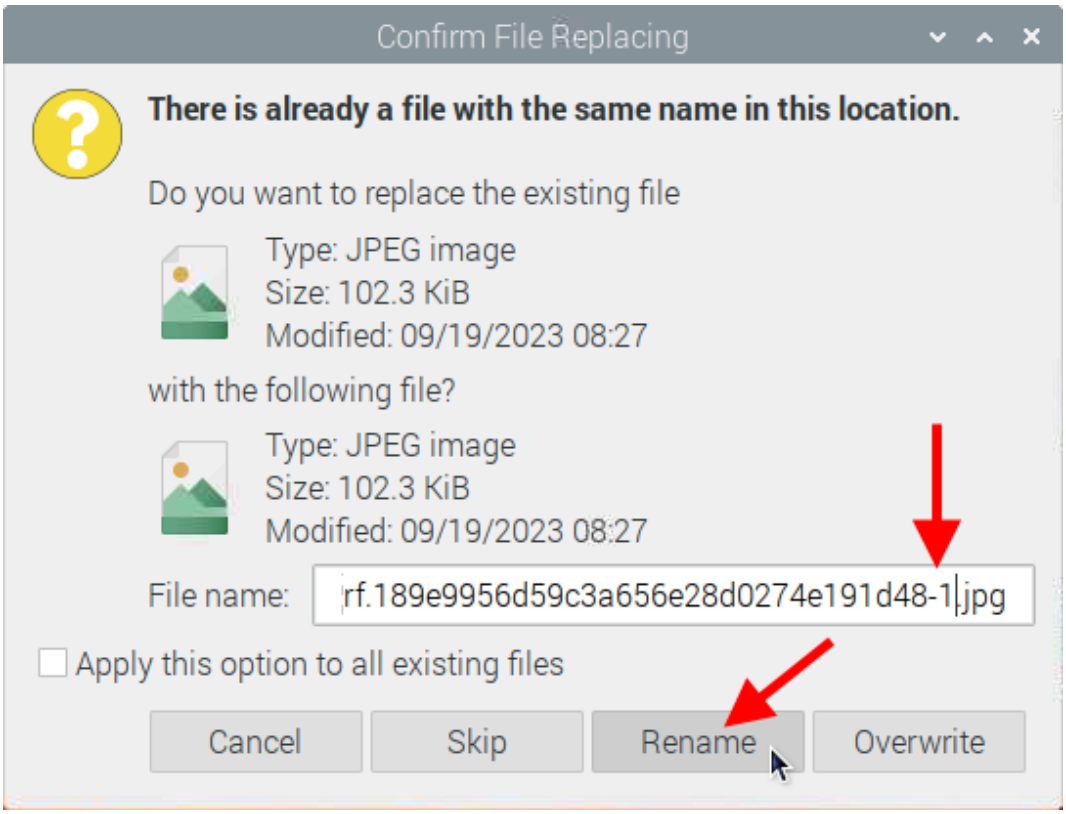
2.51 คัดลอกโฟลเดอร์ train ที่อยู่ใน /home/pi/Downloads/coding_object.v1i.yolov5pytorch ไปวางที่ /home/pi จากนั้นเปลี่ยนชื่อโฟลเดอร์เป็น coding_train



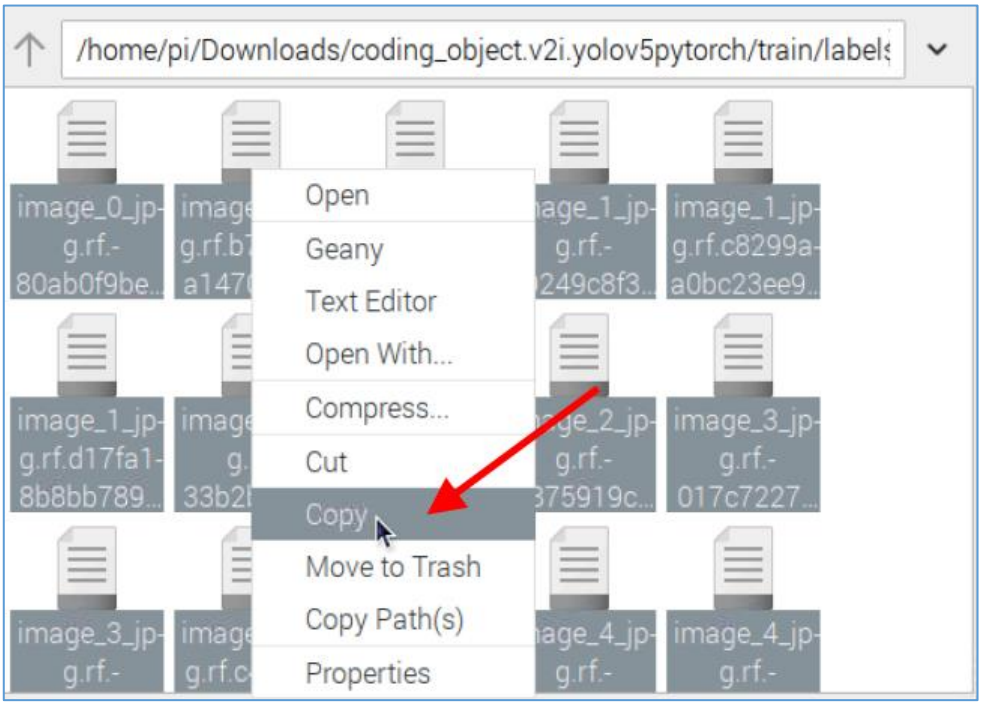
2.52 คัดลอกรูปภาพใน /home/pi/Downloads/coding_object.v2i.yolov5pytorch/train/images



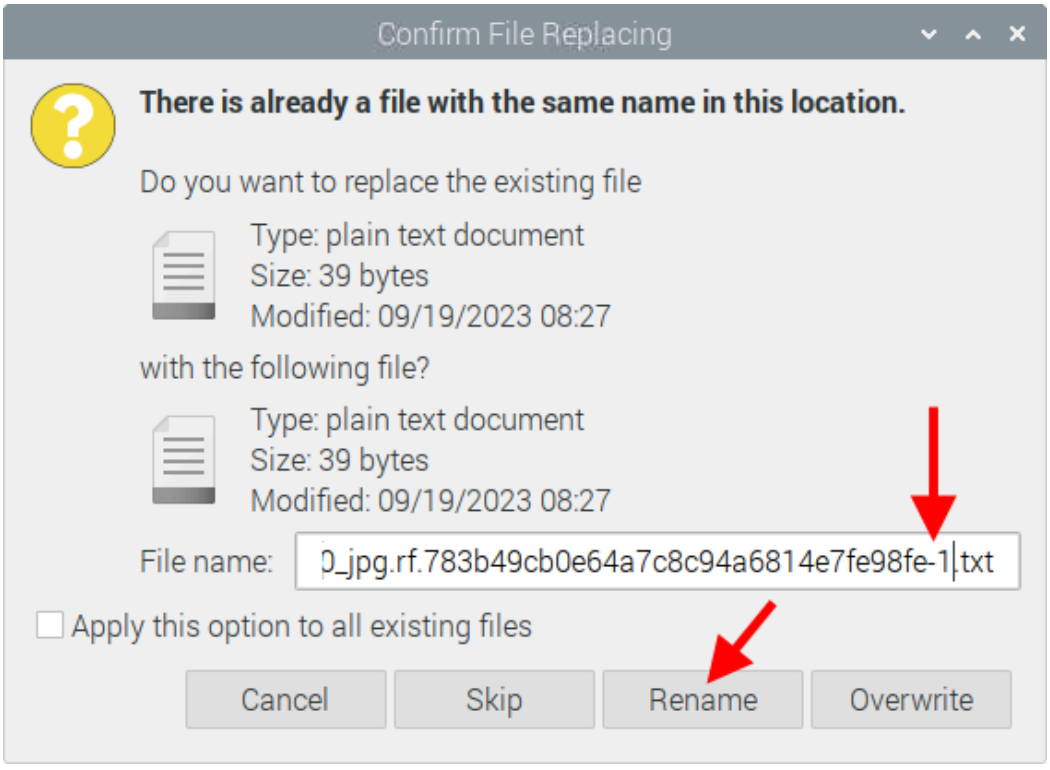
2.53 วางรูปภาพในโฟลเดอร์ /home/pi/coding_train/images หากมีชื่อไฟล์ซ้ำกับชื่อไฟล์เดิม ให้เปลี่ยนชื่อโดยใส่ “-1” ไว้ด้านหลังชื่อไฟล์ แล้วคลิก Rename



2.54 คัดลอกไฟล์เลเบล .txt ใน /home/pi/Downloads/coding_object.v2i.yolov5pytorch/train/labels

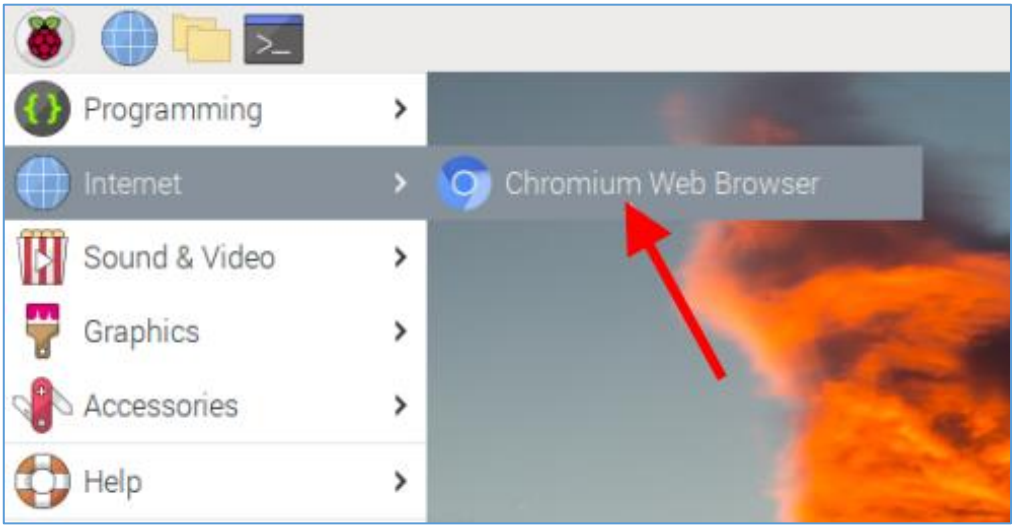


2.55 วางไฟล์ .txt ไว้ในโฟลเดอร์ /home/pi/coding_train/labels หากมีชื่อไฟล์ซ้ำกับชื่อไฟล์เดิม ให้เปลี่ยนชื่อโดยใส่ “-1” ไว้ด้านหลังชื่อไฟล์ แล้วคลิก Rename

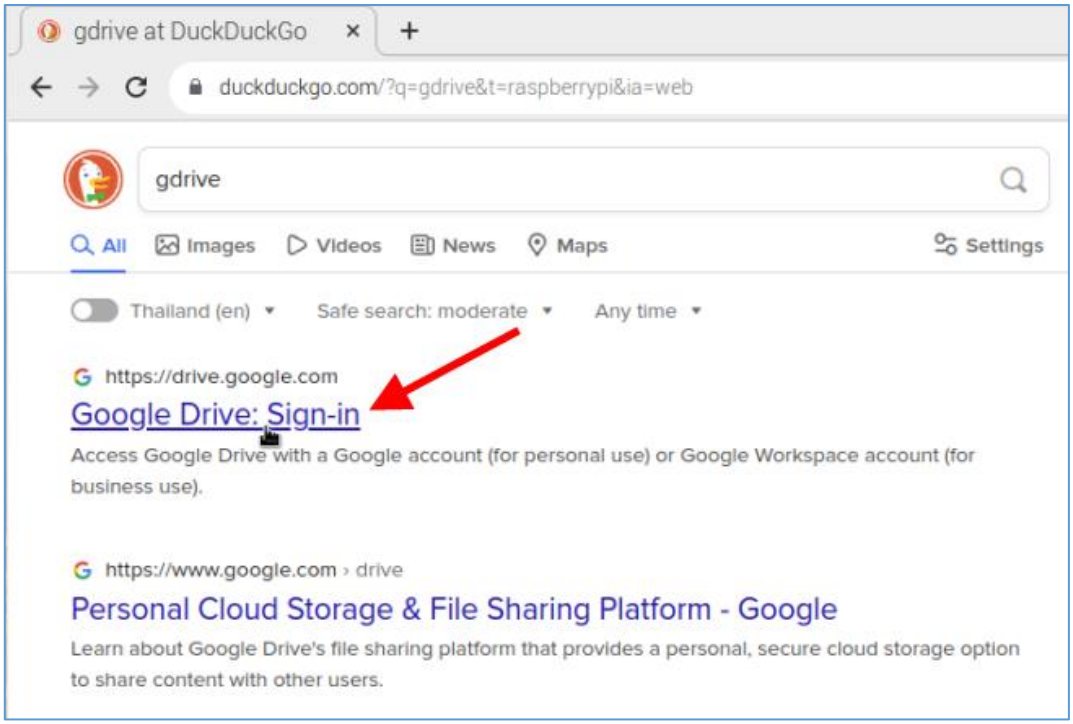


ขั้นตอนที่ 3 อัพโหลดชุดข้อมูล (Dataset) ขึ้น Google Drive

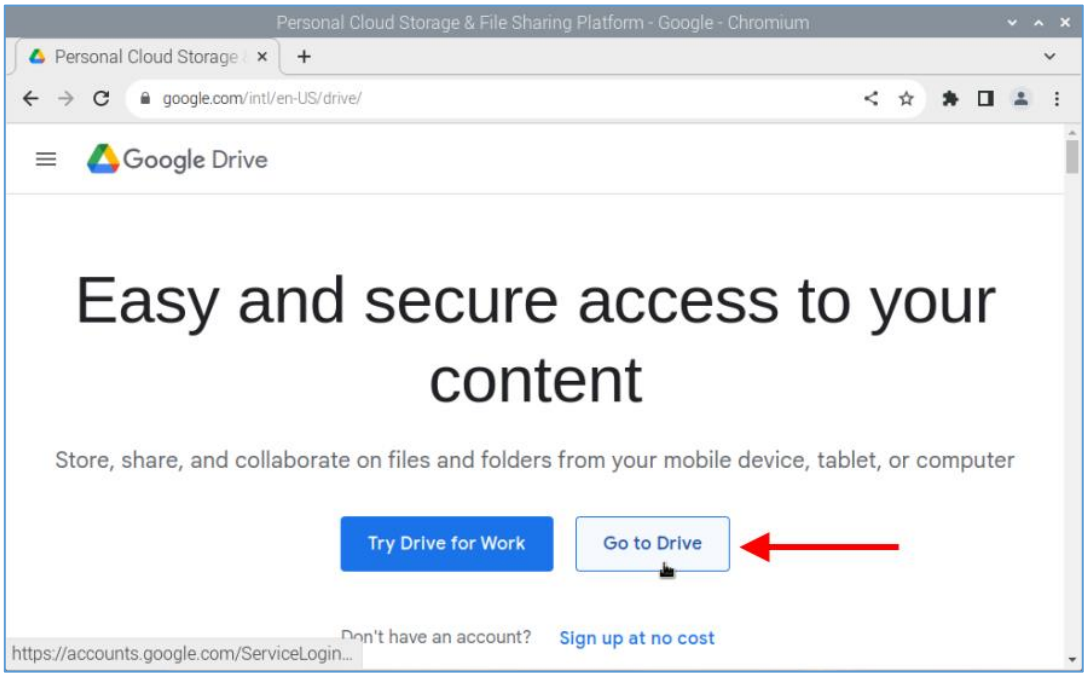
3.1 เปิดโปรแกรม Chromium Web Browser



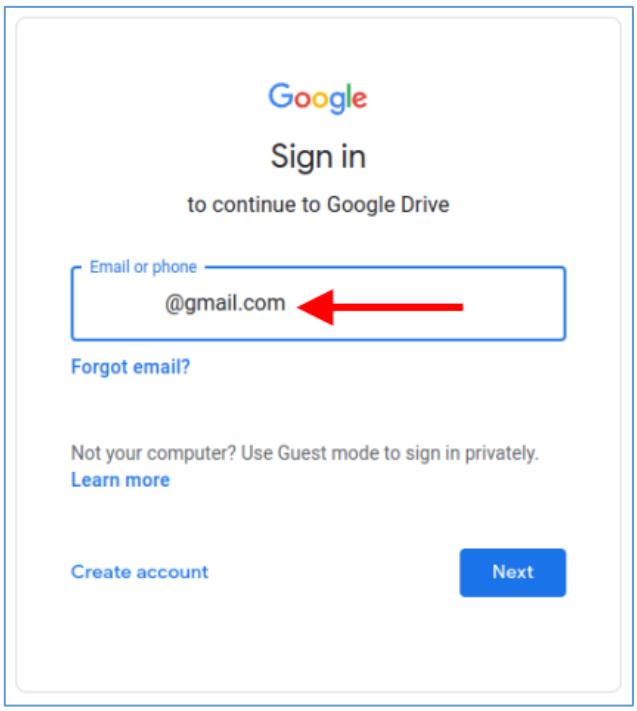
3.2 พิมพ์คำค้นหาว่า “gdrive” จากนั้นคลิกเข้าไปยังเว็บไซต์ Google Drive



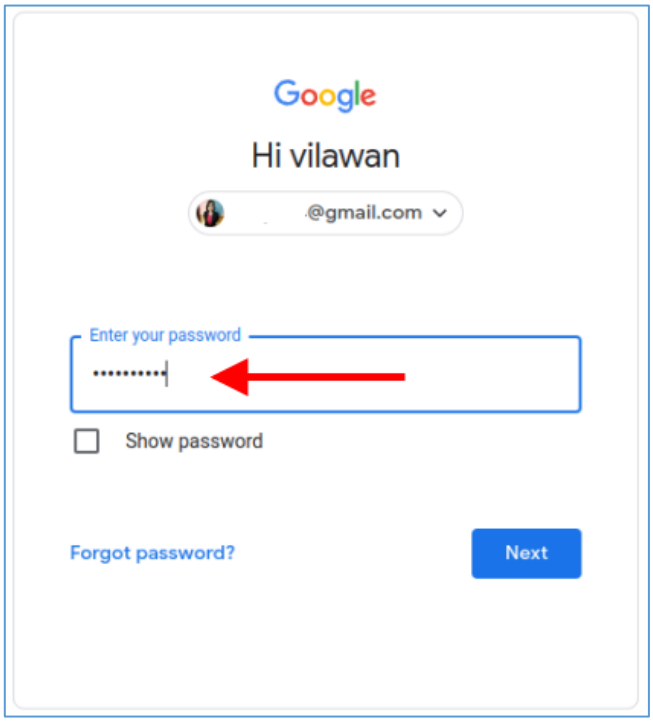
3.3 คลิกปุ่ม Go to Drive



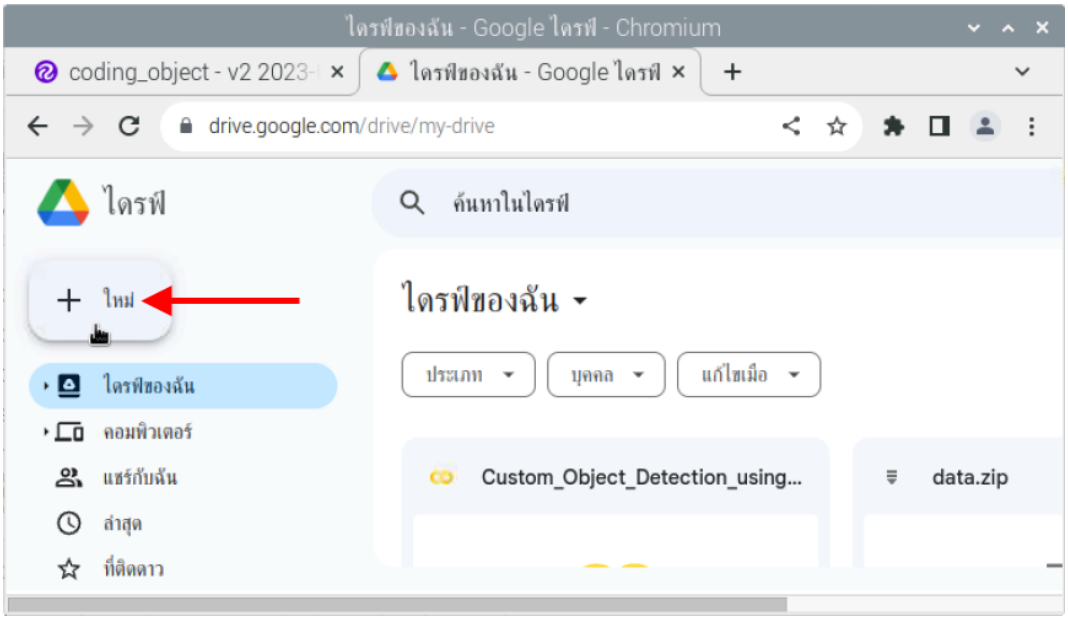
3.4 กรอกอีเมล จากนั้นคลิกปุ่ม Next



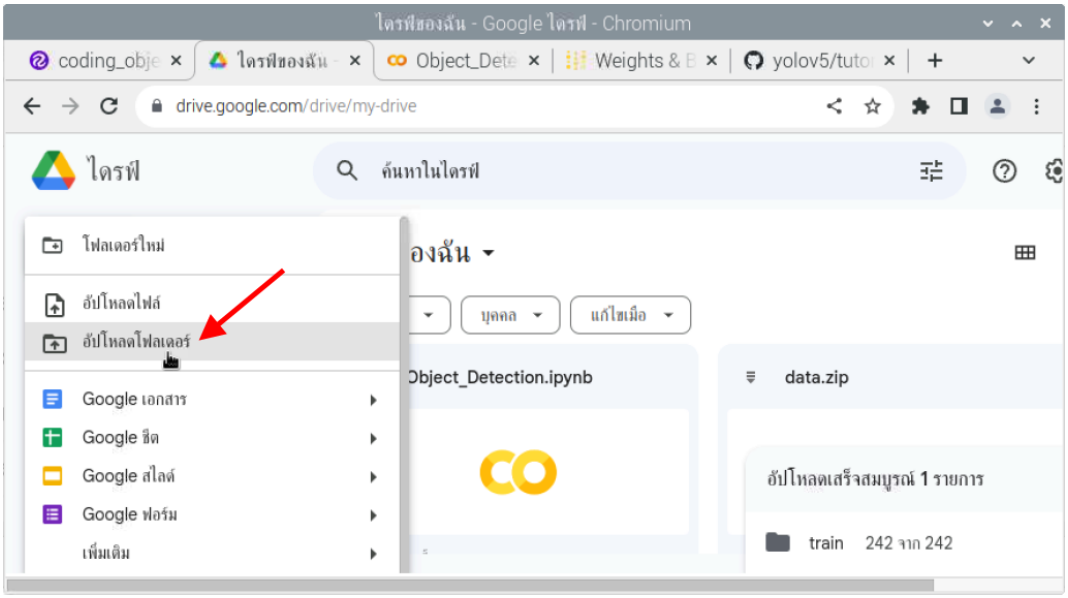
3.5 กรอกรหัสผ่าน จากนั้นคลิกปุ่ม Next



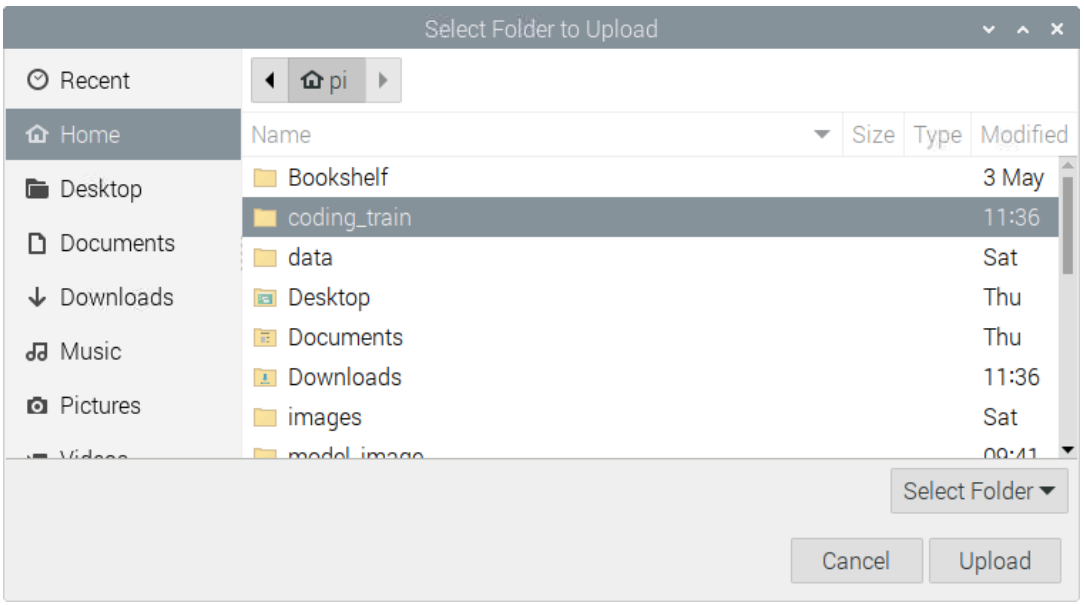
3.6 อัพโหลดโฟลเดอร์ Dataset ขึ้น Google Drive โดยคลิกปุ่ม + ใหม่



3.7 คลิก อัปโหลดโฟลเดอร์

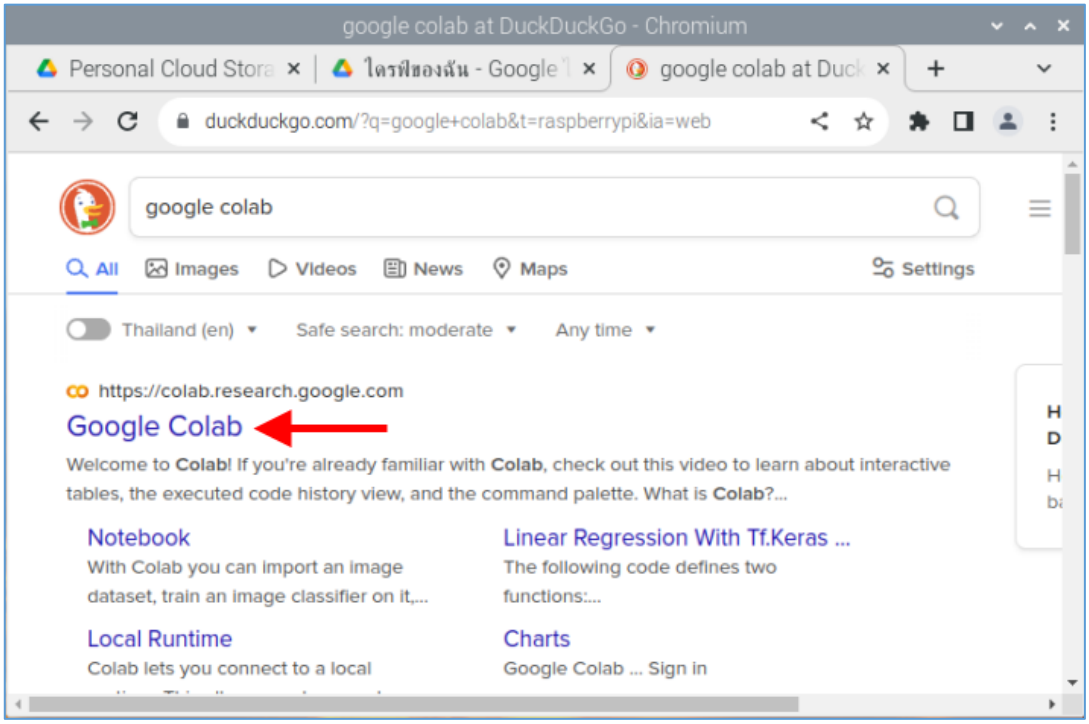


3.8 เลือกโฟลเดอร์ Dataset ในที่นี้คือโฟลเดอร์ชื่อ coding_train จากนั้นคลิกปุ่ม Upload

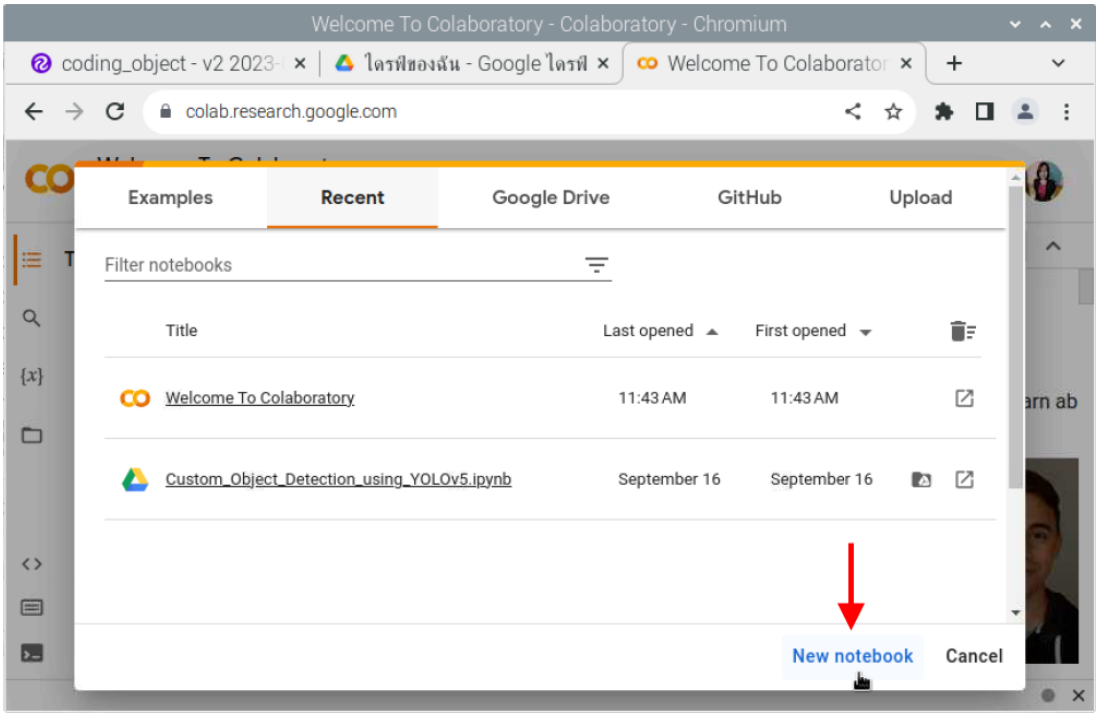


ขั้นตอนที่ 4 ฝึกฝนโมเดล (Train) ด้วย Google Colab

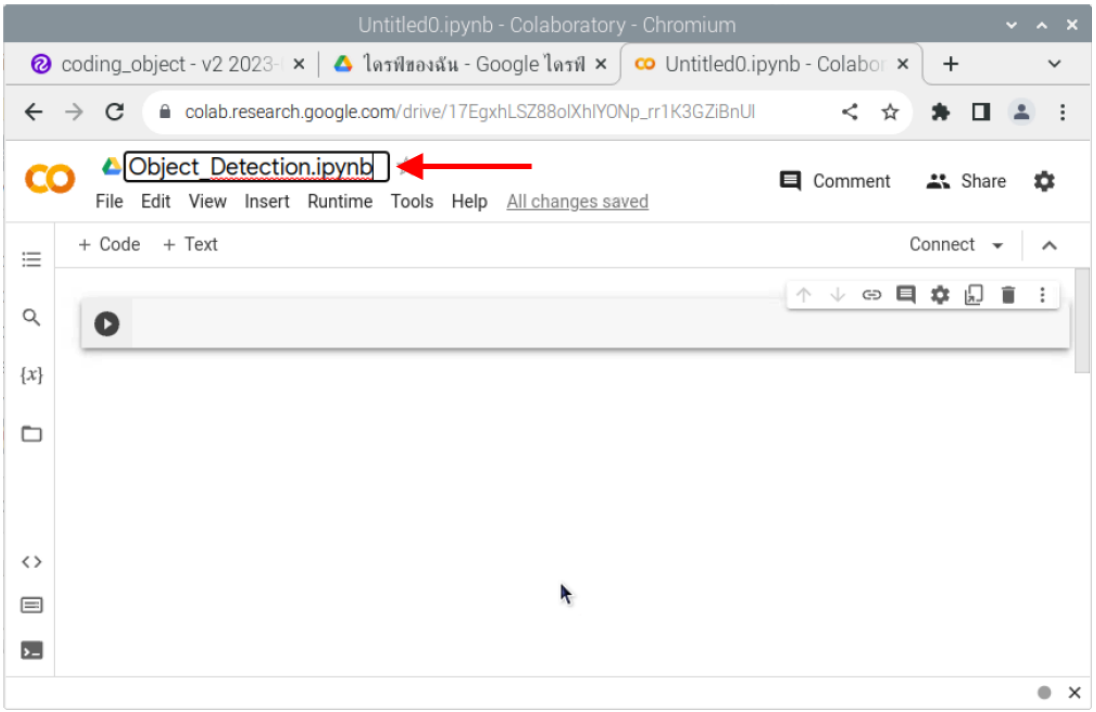
4.1 คลิกเปิดแท็บใหม่ และพิมพ์คำค้นหาว่า “google colab” จากนั้นคลิกเข้าเว็บไซต์ Google Colab



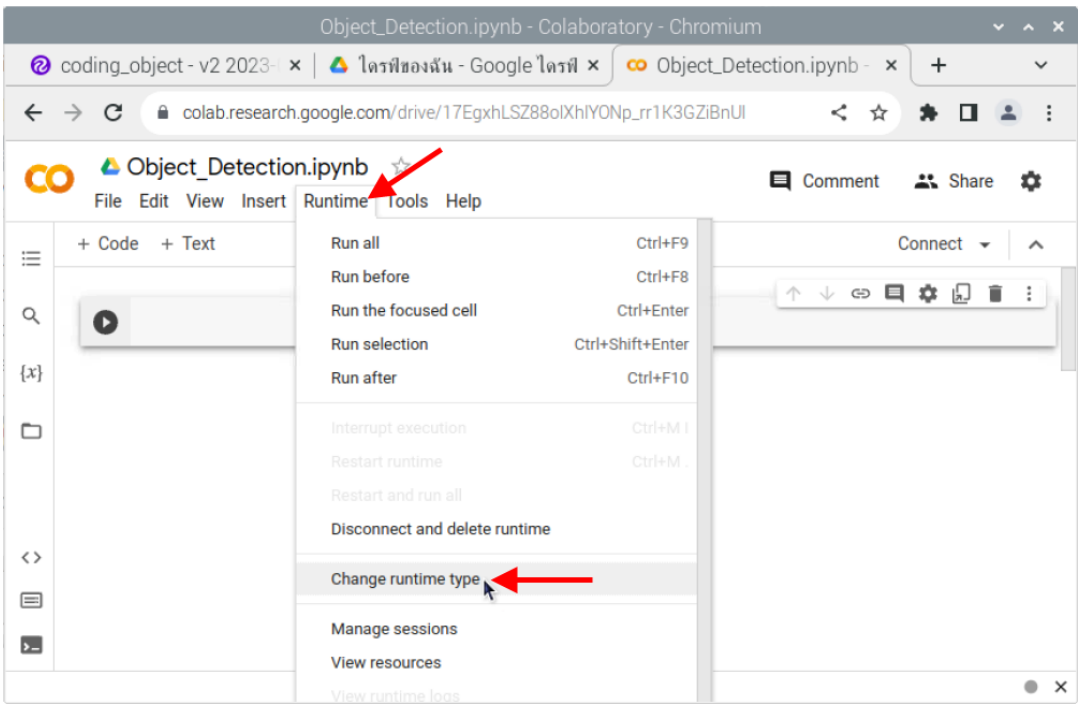
4.2 สร้างพื้นที่การทำงานใหม่ โดยคลิกปุ่ม New notebook หรือคลิกแท็บ File --> New notebook



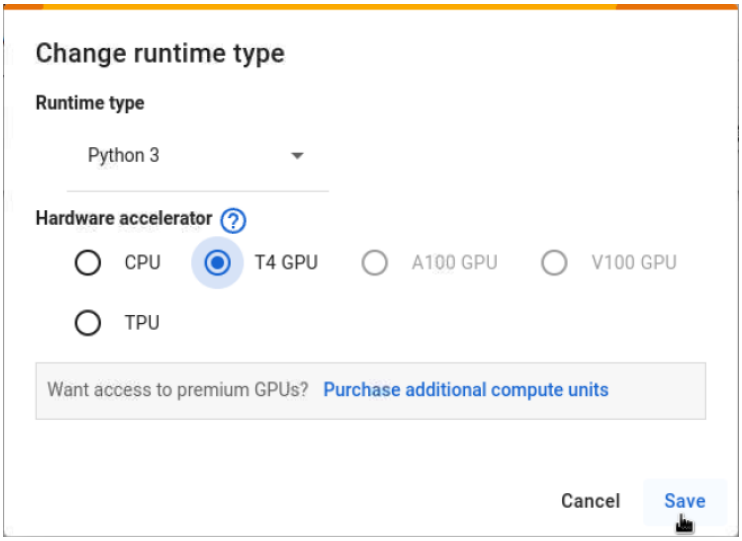
4.3 ตั้งชื่อโมเดล



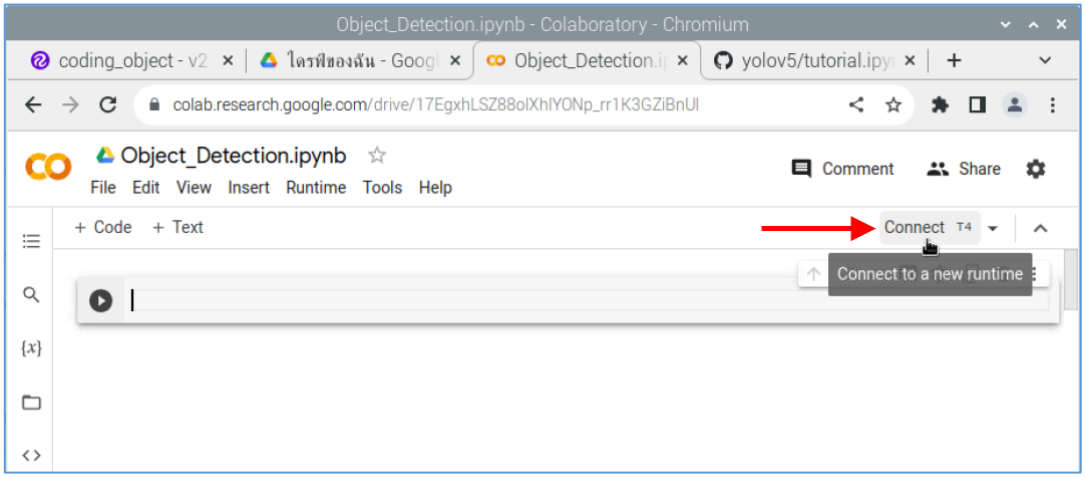
4.4 เลือกรันไทม์ โดยคลิกแท็บ Runtime แล้วคลิกเลือก Change runtime type



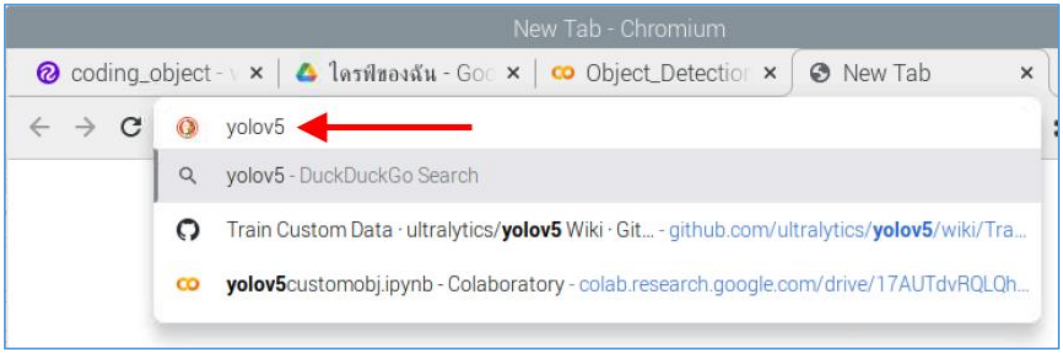
4.5 เลือกรันไทม์แบบ GPU เพื่อให้ประมวลผลเร็วขึ้น จากนั้นคลิกปุ่ม Save



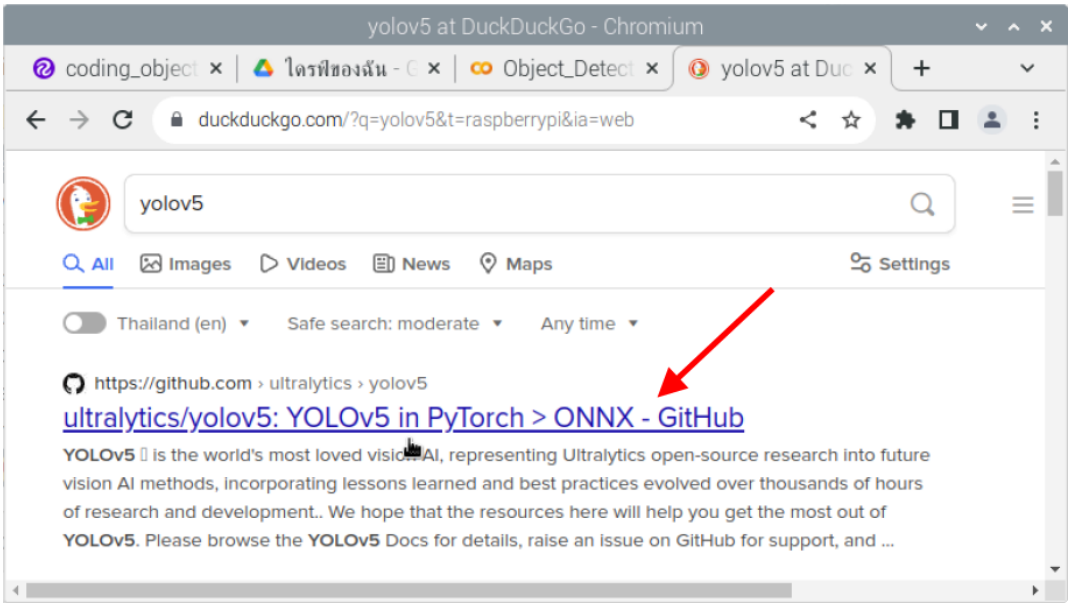
4.6 คลิกปุ่ม Connect เพื่อเชื่อมต่อกับบัญชี Gmail



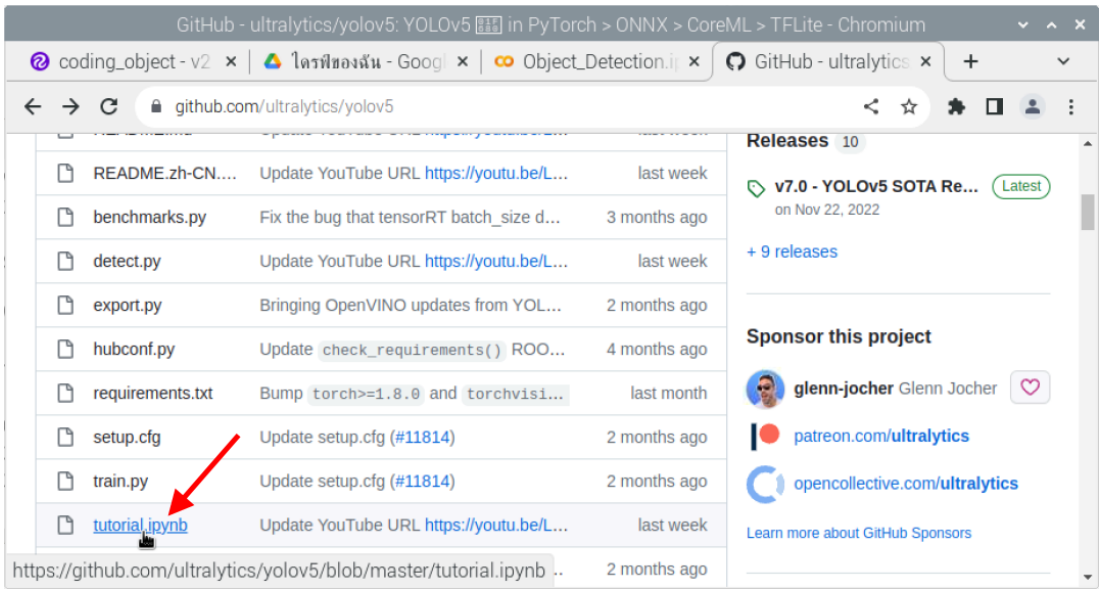
4.7 ค้นหาตัวอย่างโค้ดในการรันโปรแกรม โดยเปิดแท็บใหม่ และพิมพ์คำค้นหาว่า “yolov5”



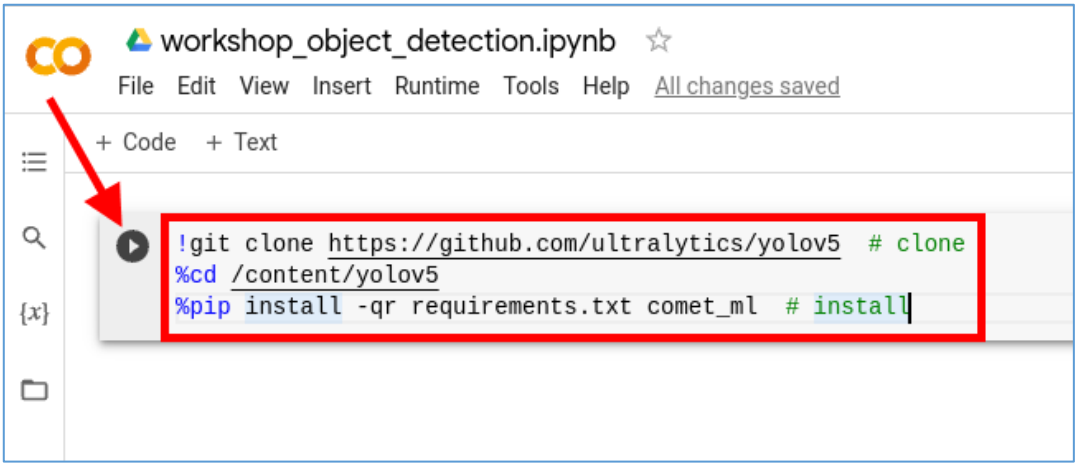
4.8 คลิกเข้าเว็บไซต์ https://github.com/ultralytics/yolov5



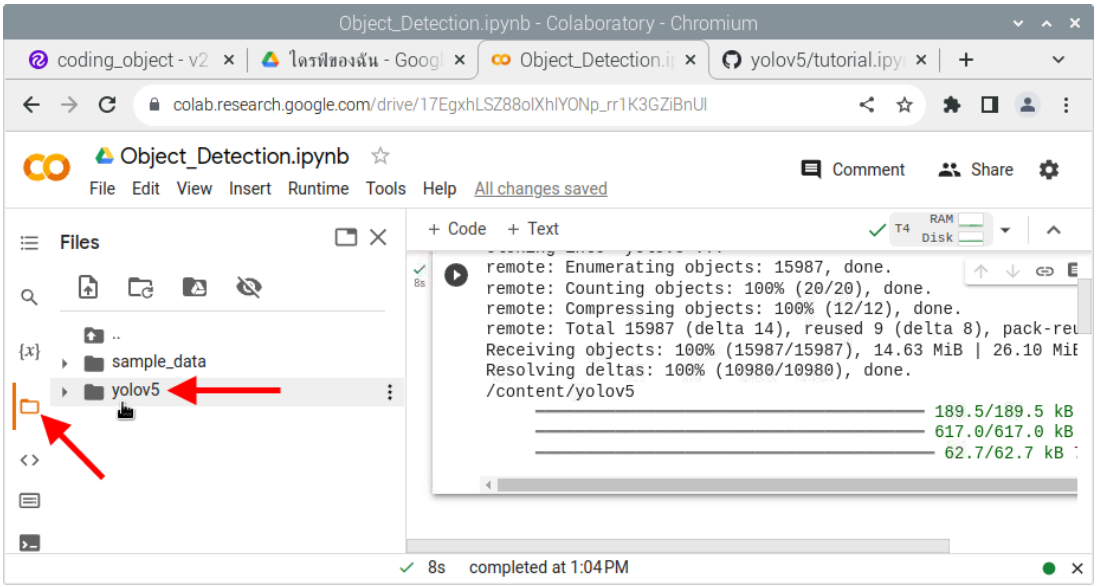
4.9 คลิกไฟล์ tutorial.ipnb



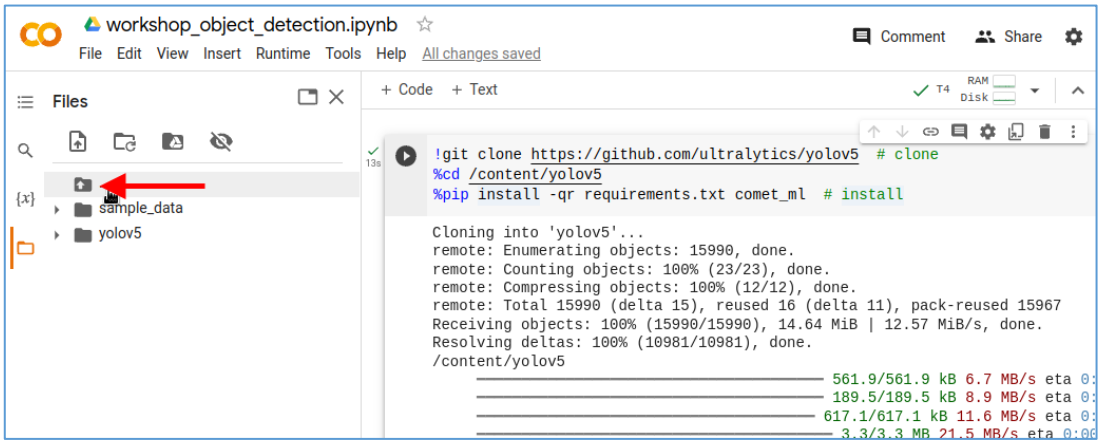
4.10 คัดลอกโค้ดคำสั่งเพื่อดาวน์โหลดไฟล์ Source Code ของ yolov5 และติดตั้งไลบรารี่ต่าง ๆ ที่จำเป็นต้องใช้ จากนั้นคลิกปุ่มรัน ดังรูป

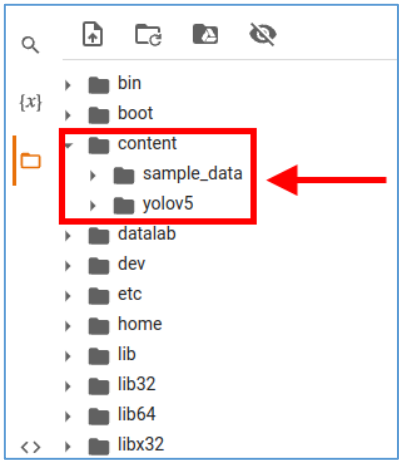


4.11 คลิกที่ Files (รูปโฟลเดอร์ด้านซ้ายมือ) จะปรากฏโฟลเดอร์ yolov5 ขึ้นมา

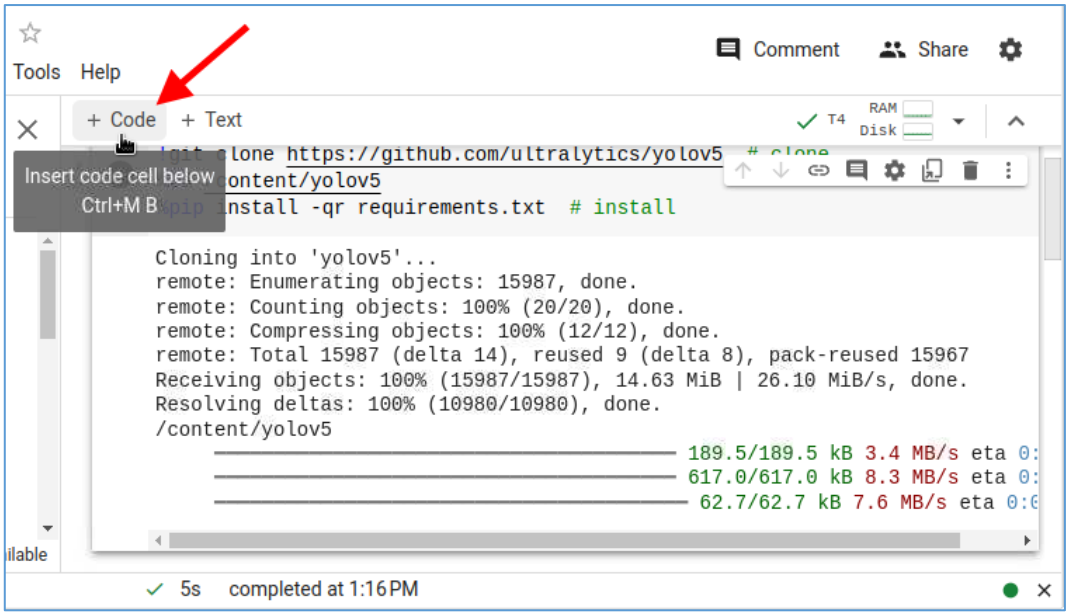


4.12 คลิก จะเห็นว่าโฟลเดอร์ yolov5 อยู่ในโฟลเดอร์ content

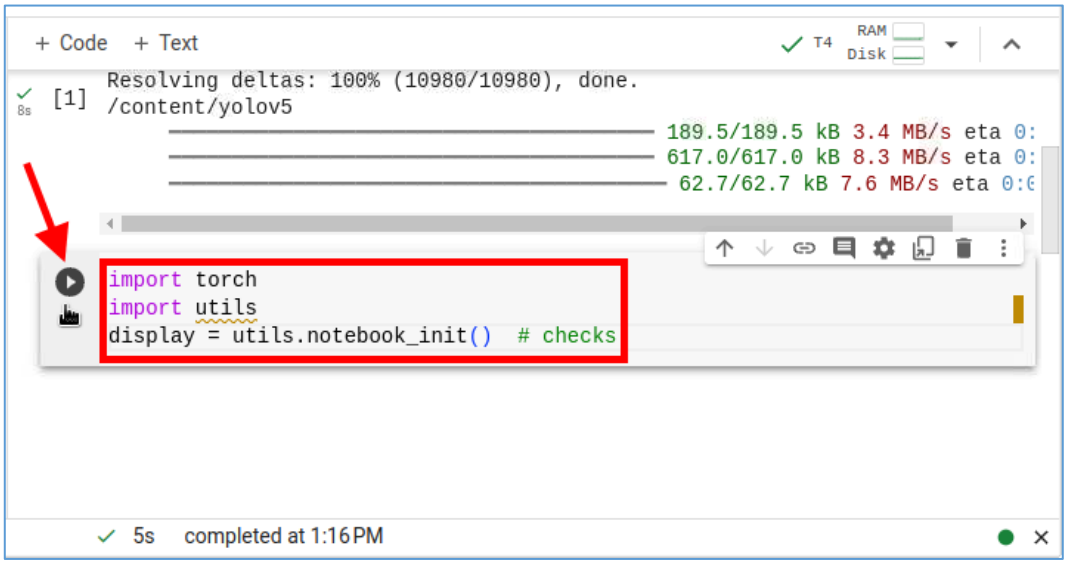




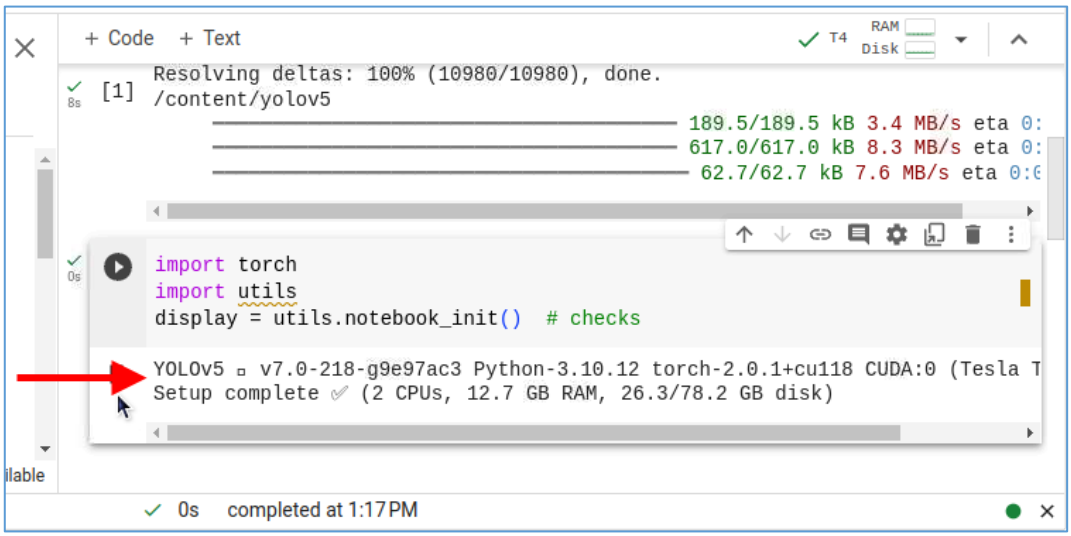
4.13 ทดสอบว่าการติดตั้ง PyTorch สมบูรณ์หรือไม่ โดยคลิก + Code เพื่อเพิ่มโค้ดบรรทัดใหม่



4.14 คัดลอกคำสั่งจากไฟล์ tutorial.ipnb มาวาง จากนั้นคลิกปุ่มรัน ดังรูป

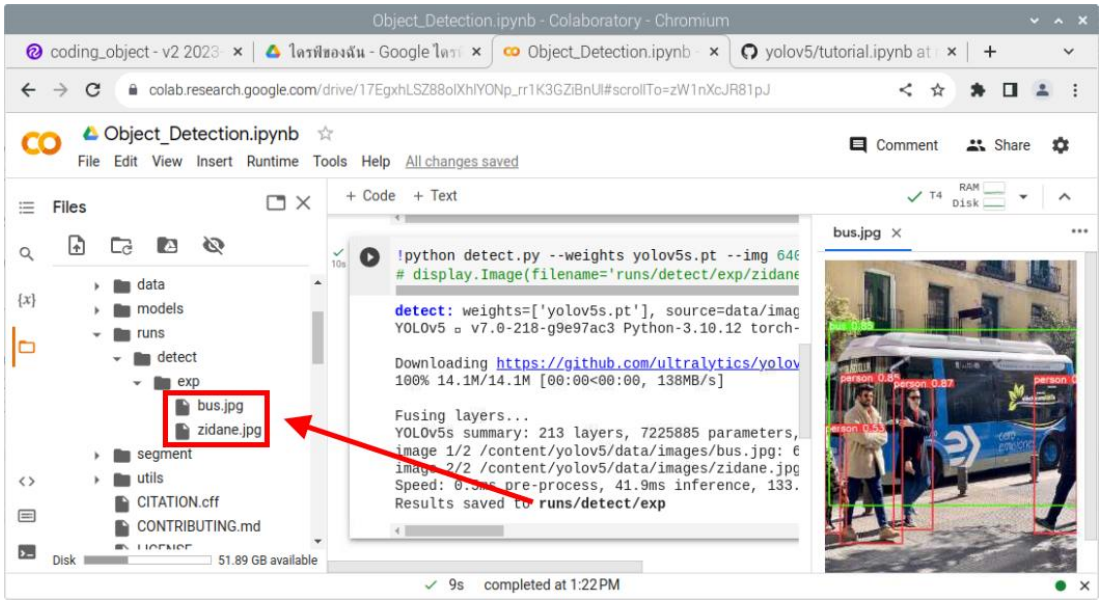


4.15 จะได้ผลลัพธ์ดังรูป แสดงว่าติดตั้ง PyTorch ผ่าน

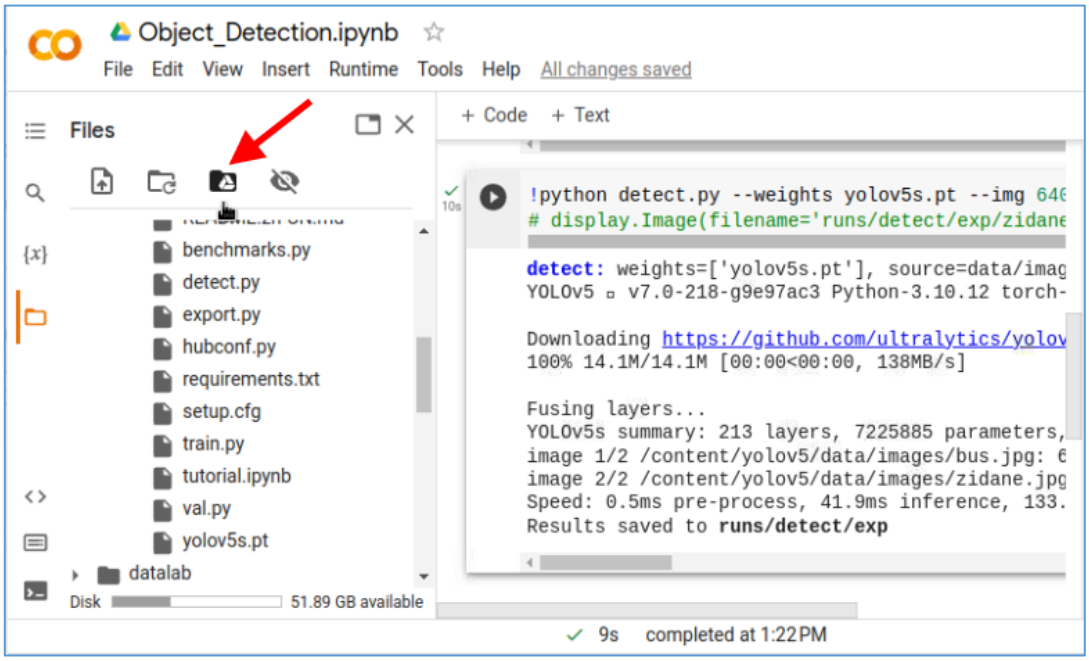


4.16 ทดสอบการใช้ Yolov5 ในการตรวจจับวัตถุของรูปภาพตัวอย่างที่มาพร้อมกับ yolov5 โดยรันไฟล์ detect.py และแสดงผลลัพธ์ออกมาด้านล่าง โดยคลิก + Code แล้วใส่คำสั่งดังรูป (คัดลอกโค้ดได้จาก tutorial.ipynb)

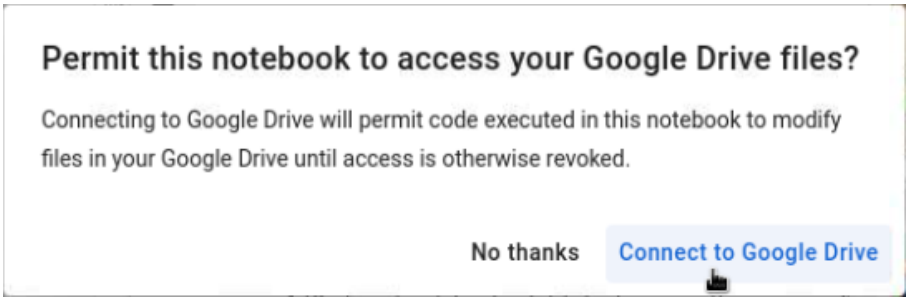
4.17 ผลลัพธ์จะถูกบันทึกไว้ในโฟลเดอร์ runs/detect/exp หากมีการรันไฟล์ detect อีกครั้ง จะได้โฟลเดอร์ exp1, exp2 ไปเรื่อย ๆ

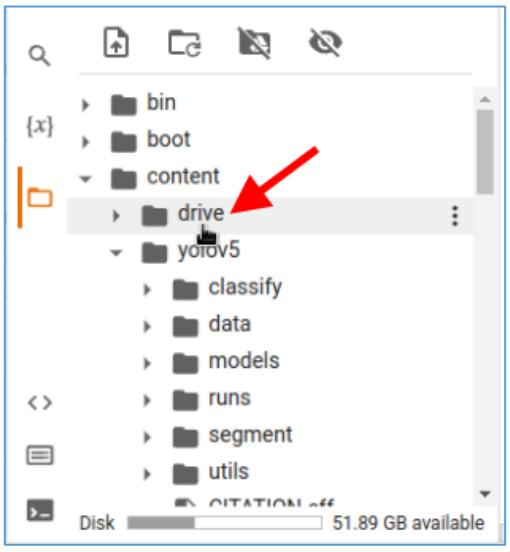


4.18 เชื่อมต่อกับ Google Drive เพื่อเรียกใช้งานรูปภาพ Dataset โดยคลิกที่ Mount Google Drive

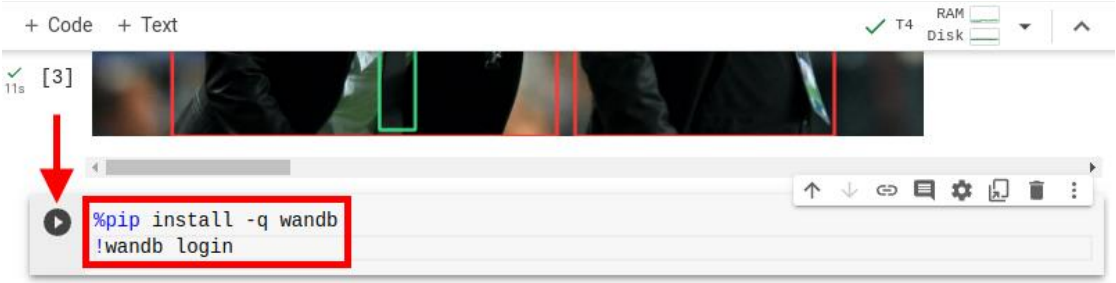


4.19 คลิก Connect to Google Drive จะปรากฏโฟลเดอร์ drive ขึ้นมาในโฟลเดอร์ Content

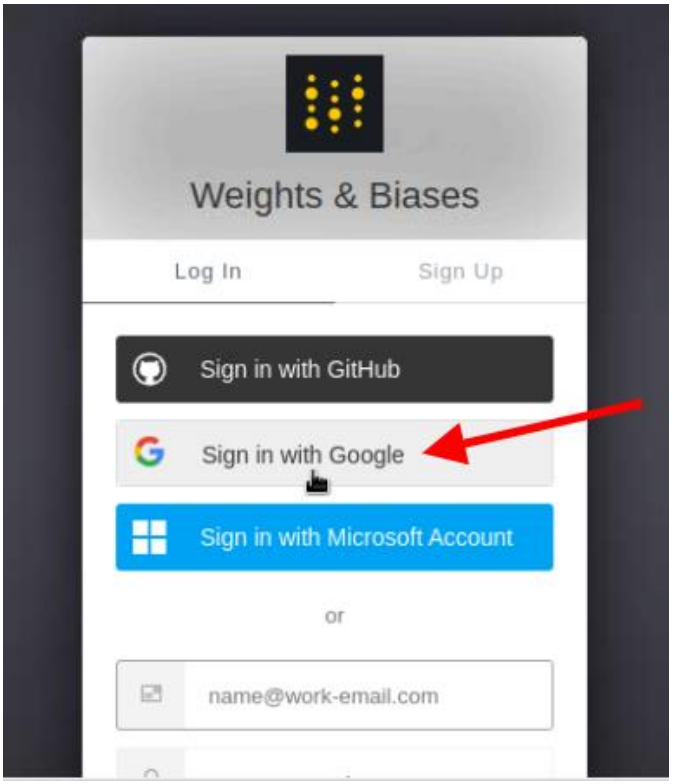




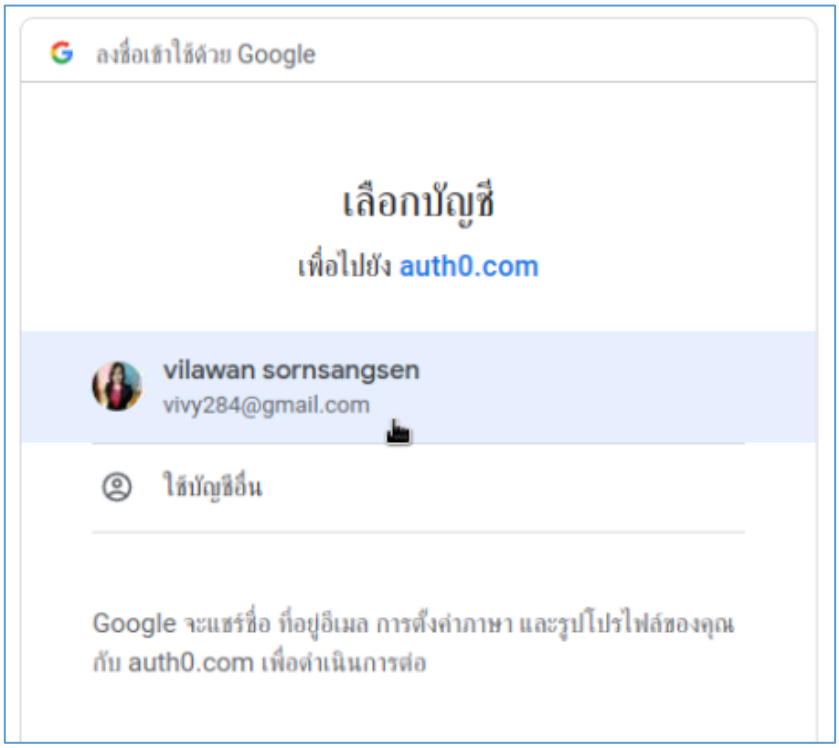
4.20 ติดตั้ง wandb (Weight & Biases) สำหรับใช้เป็น Dashboard ในการติดตามผลการสร้างแบบจำลอง โดยคลิก + Code แล้วพิมพ์คำสั่ง %pip install -q wandb และ !wandb login จากนั้นคลิกปุ่มรัน จะปรากฏลิงก์ขึ้นมาให้คลิกที่ https://wandb.ai/authorize



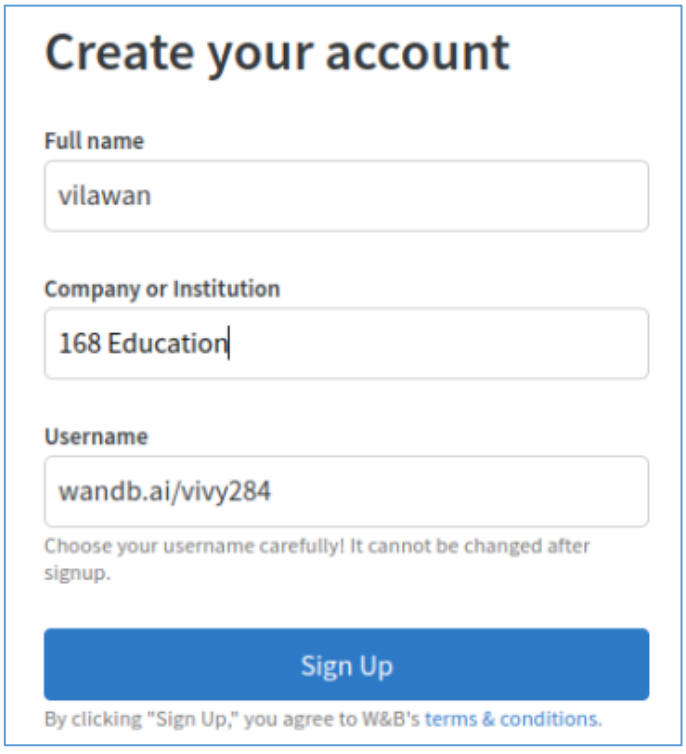
4.21 คลิกล็อกอินด้วย Gmail โดยคลิกปุ่ม Sign in with Google



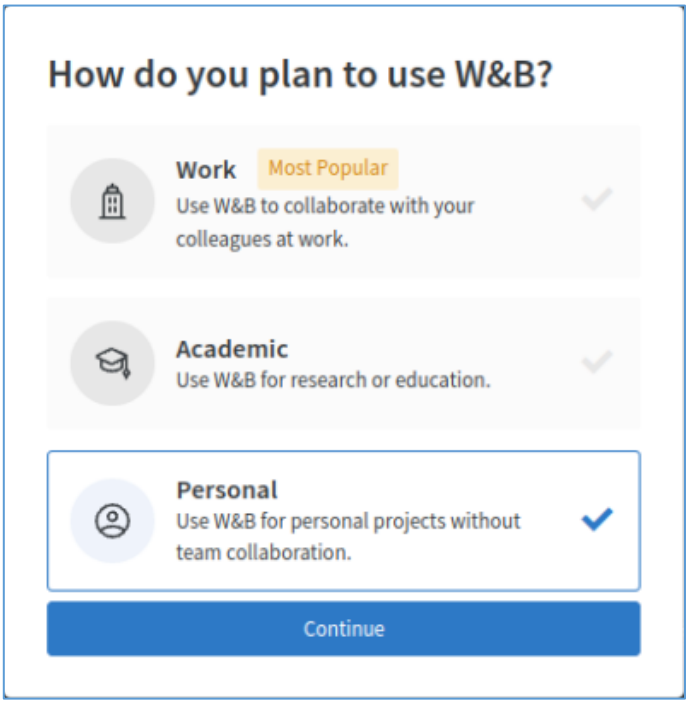
4.22 เลือกบัญชีที่จะล็อกอิน



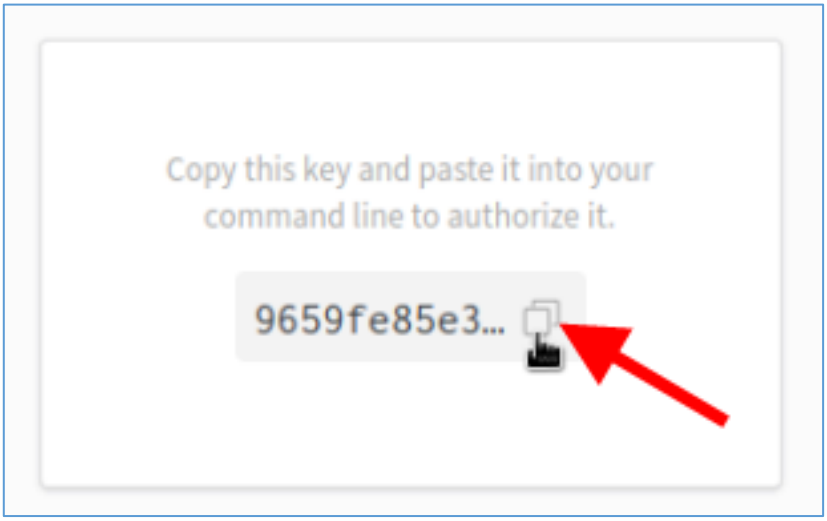
4.23 กรอกชื่อ / บริษัท จากนั้นคลิก Sign Up



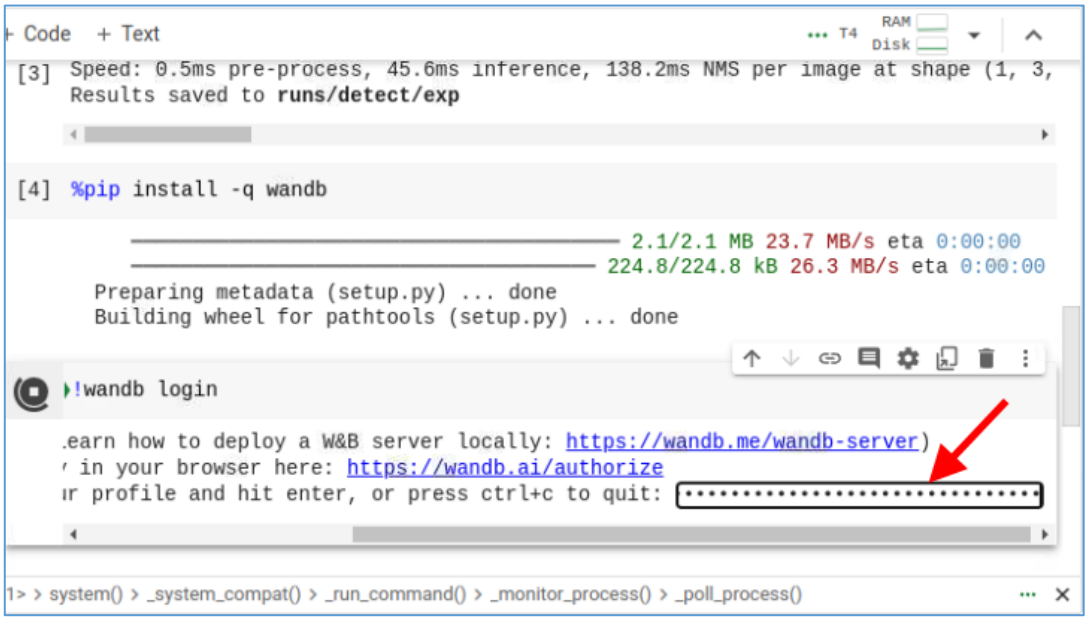
4.24 เลือกแผนการใช้งาน ให้คลิกเป็น Personal จากนั้นคลิกปุ่ม Continue



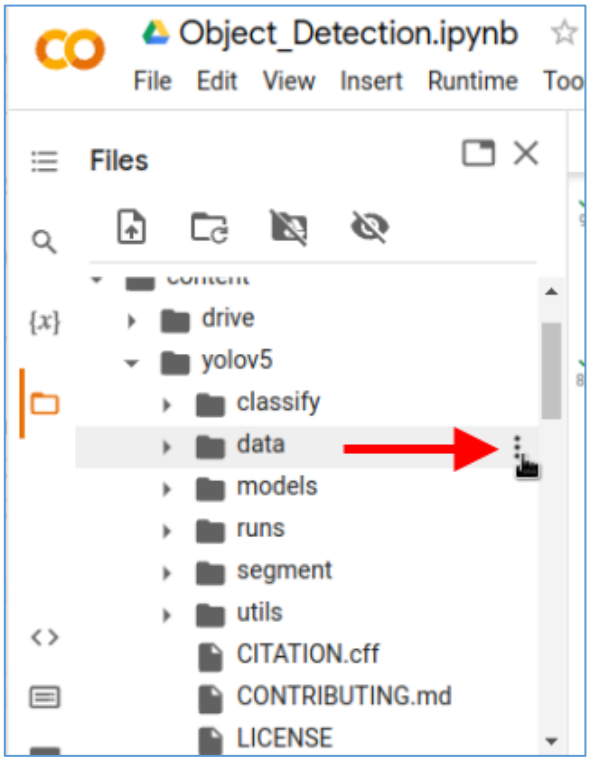
4.25 คลิกคัดลอกรหัส Authorize ซึ่งแต่ละคนจะได้รหัสไม่เหมือนกัน



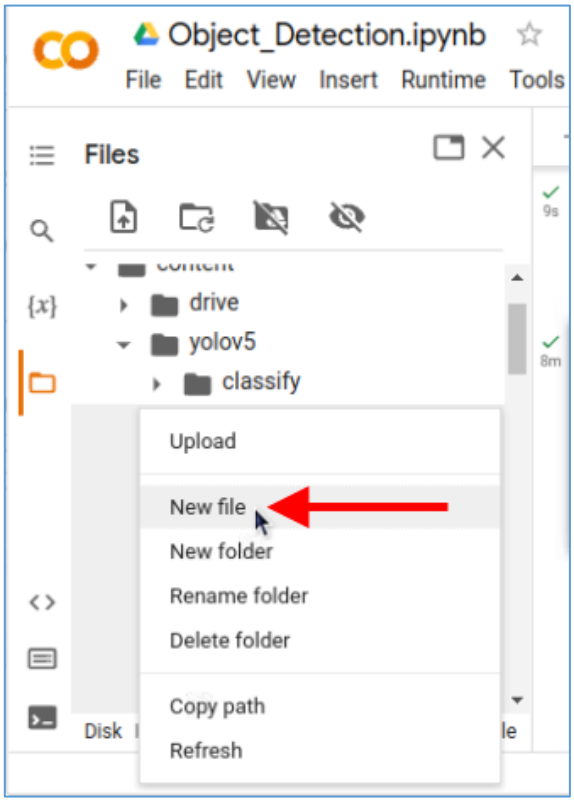
4.26 นำรหัส Authorize มาวาง ดังรูป จากนั้น Enter



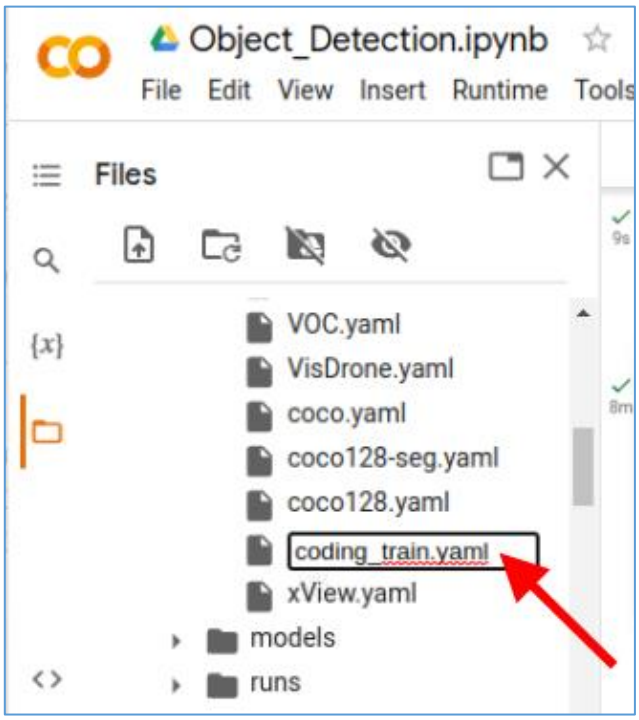
4.27 สร้างไฟล์ใหม่ในโฟลเดอร์ data โดยคลิกที่จุด 3 จุดด้านหลังโฟลเดอร์ data



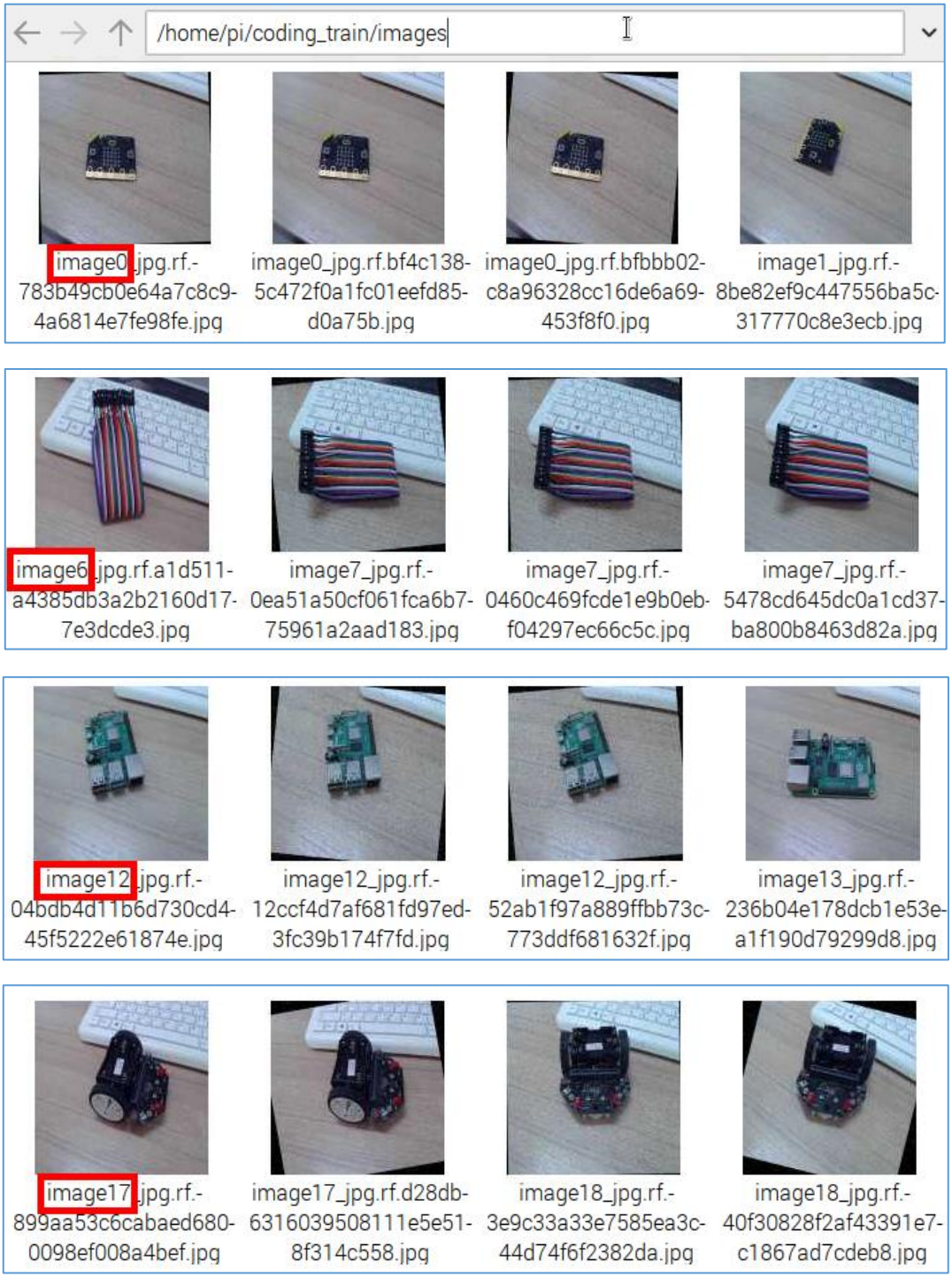
4.28 คลิกเลือก New file



4.29 ตั้งชื่อไฟล์เป็น coding_train.yaml จากนั้นดับเบิ้ลคลิกที่ไฟล์เพื่อเขียนโค้ดคำสั่ง



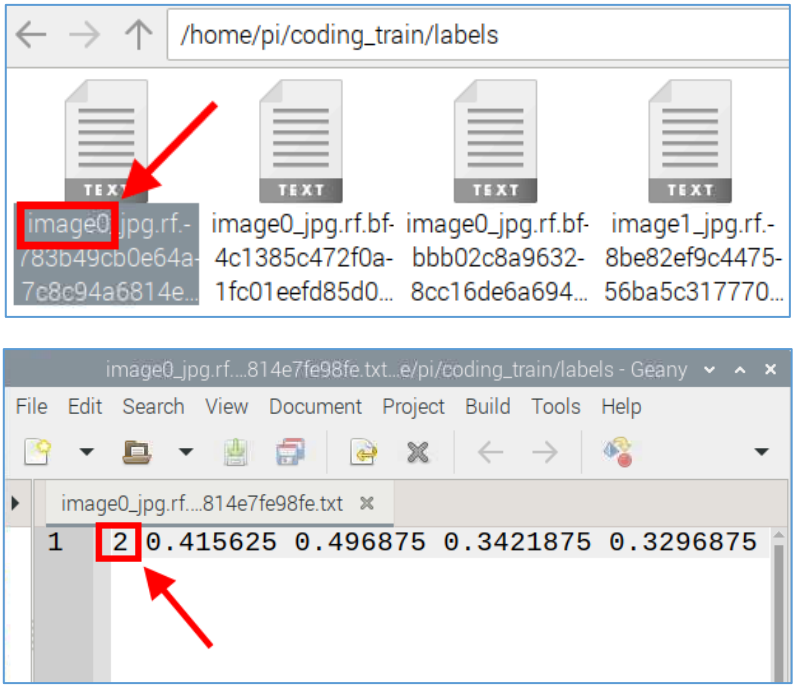
4.30 ตรวจสอบ id ของวัตถุทุกชิ้นที่จะทำการตรวจจับ โดยเปิดโฟลเดอร์ /home/pi/coding_train/images จากนั้นจดชื่อไฟล์รูปภาพทุกชิ้น



ตัวอย่างเช่น รูปภาพ Microbit = image0, สายแพร์ = image 6, Raspberry Pi = image12 และ Maqueen = image17

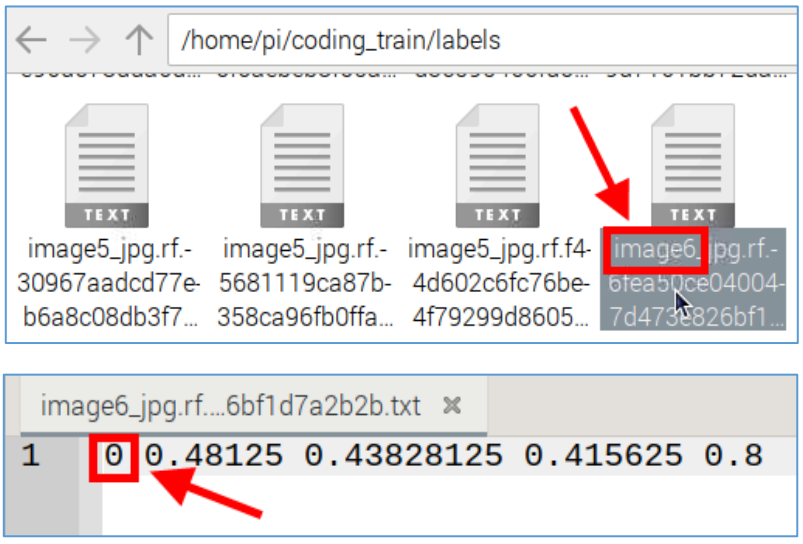
4.31 เข้าโฟลเดอร์ /home/pi/coding_train/labels จากนั้นเปิดไฟล์ลาเบล .txt ตามชื่อไฟล์รูปภาพที่ได้ในข้อ 4.30 ตัวอย่างเช่น

- รูปภาพ Microbit = image0



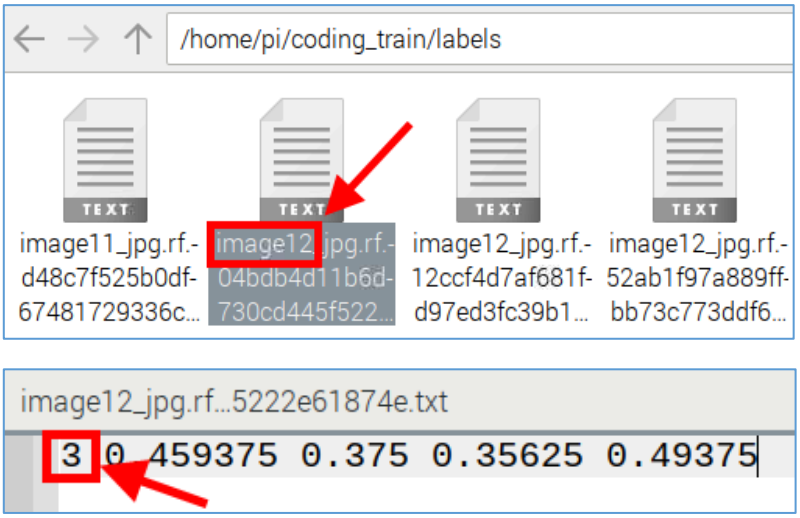
จะได้ว่า id ของ Microbit คือ 2

- รูปภาพสายแพร์ = image 6



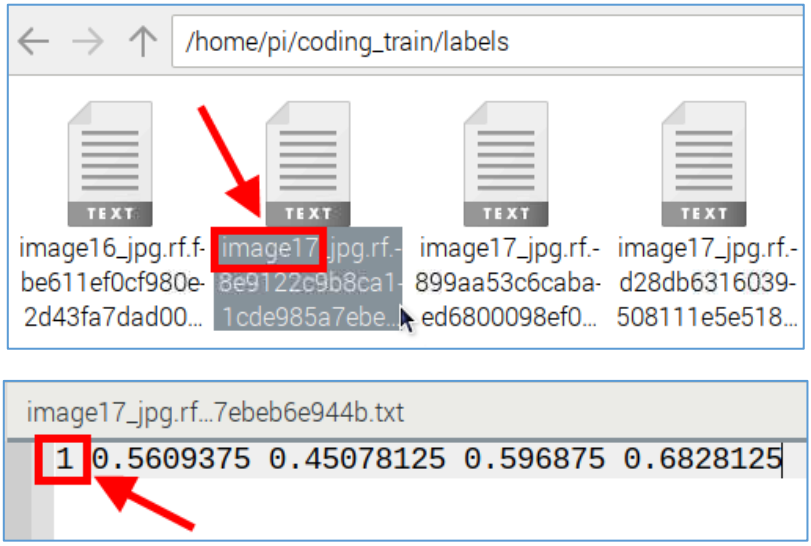
จะได้ว่า id ของสายแพร์ คือ 0

- รูปภาพ Raspberry Pi = image12



จะได้ว่า id ของ Raspberry Pi คือ 3

- รูปภาพ Maqueen = image17



จะได้ว่า id ของ Maqueen คือ 1

สรุปว่า id = 0 คือ สายแพร์

id = 1 คือ Maqueen

id = 2 คือ Microbit

id = 3 คือ Raspberry Pi

หมายเหตุ : แต่ละคนจะได้ id แตกต่างกันออกไป

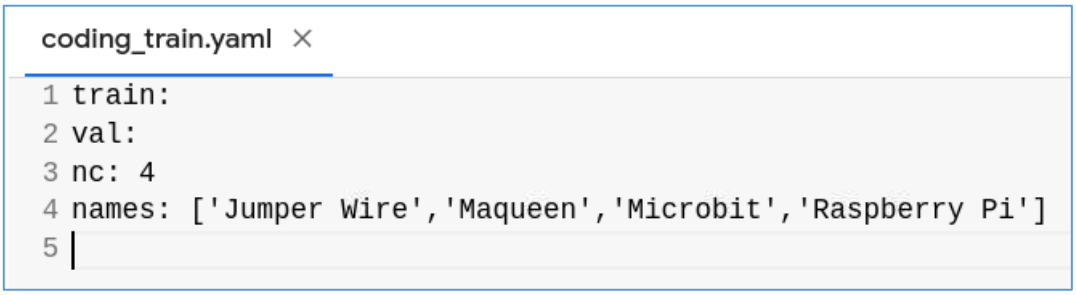
4.32 พิมพ์โค้ดคำสั่งใน coding_train.yaml ดังรูป

train: ตำแหน่งของโฟลเดอร์ Dataset ที่อยู่ใน Google Drive

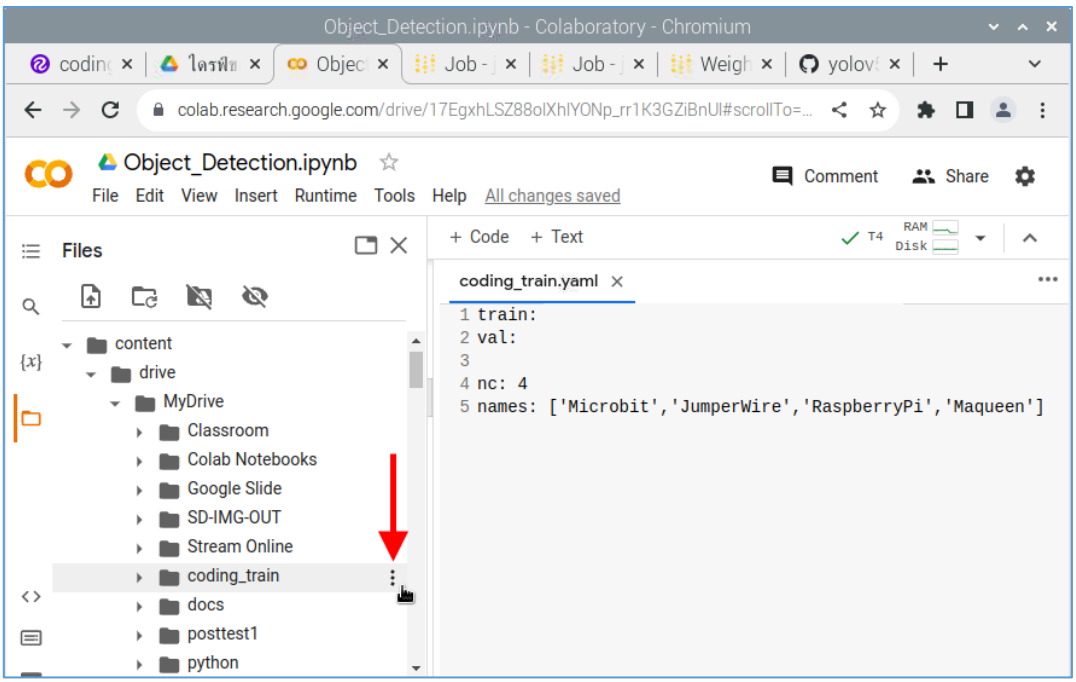
val: ตำแหน่งของโฟลเดอร์ Dataset ที่อยู่ใน Google Drive

nc: จำนวนวัตถุที่ทำการตรวจจับ

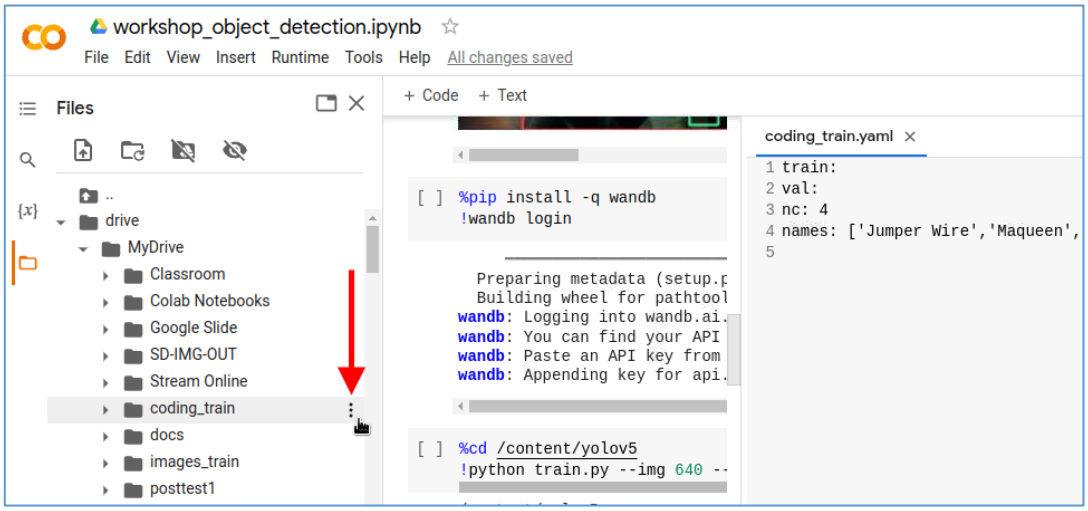
names: [‘ชื่อวัตถุที่มี id = 0’, ‘ชื่อวัตถุที่มี id = 1’, ‘ชื่อวัตถุที่มี id = 2’, ‘ชื่อวัตถุที่มี id = 3’]



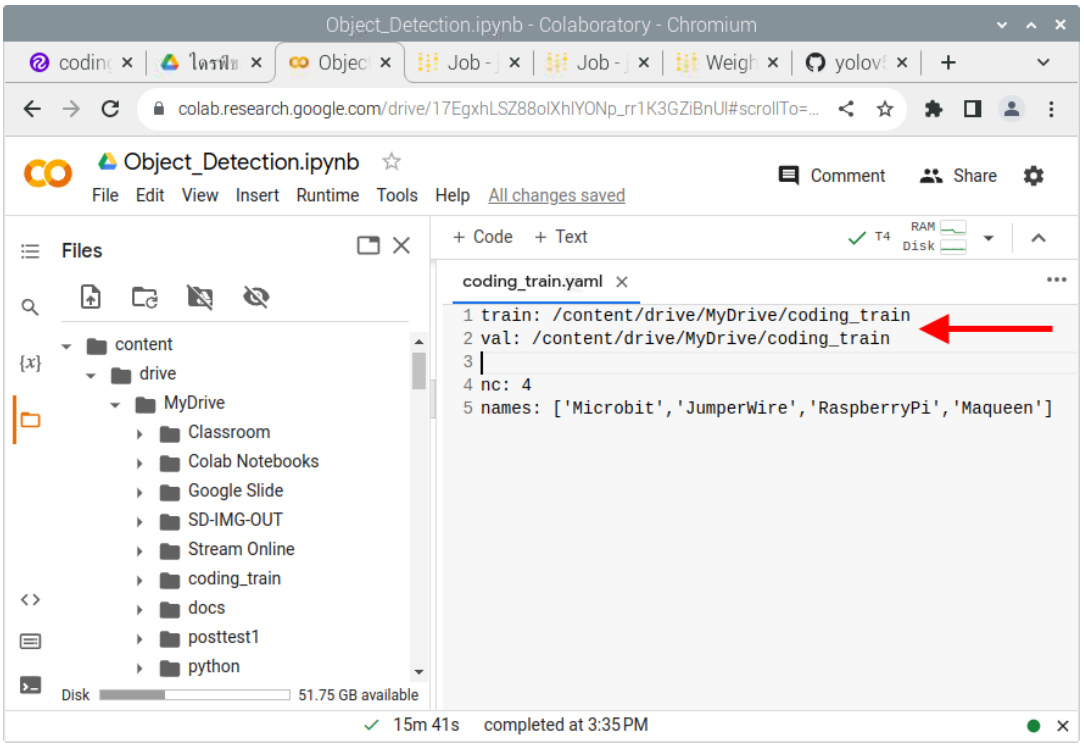
4.33 คลิกที่จุด 3 จุดด้านหลังโฟลเดอร์ coding_train (อยู่ในโฟลเดอร์ drive/MyDrive)



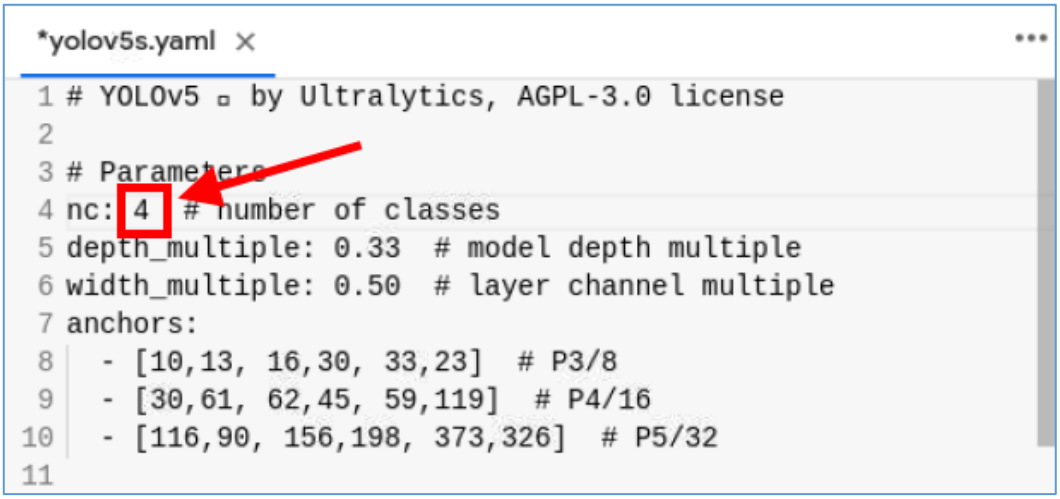
4.34 คลิก Copy path เพื่อคัดลอกตำแหน่งของชุดข้อมูล Dataset ที่อยู่ใน Google Drive



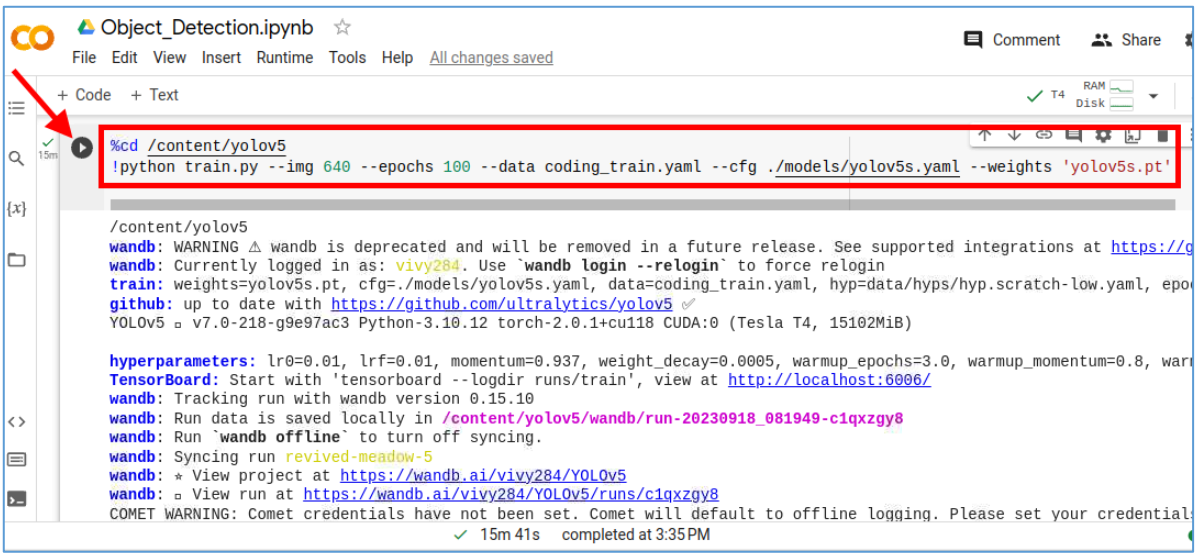
4.35 นำ Path ของโฟลเดอร์ coding_train มาวางด้านหลังโค้ดคำสั่ง train: และ val: ดังรูป



4.36 เปิดไฟล์ yolov5s.yaml ซึ่งอยู่ในโฟลเดอร์ models มาแก้ไขโค้ดคำสั่ง nc: 80 ให้เป็น nc: จำนวนวัตถุที่จะตรวจจับ เหมือนไฟล์ coding_train.yaml

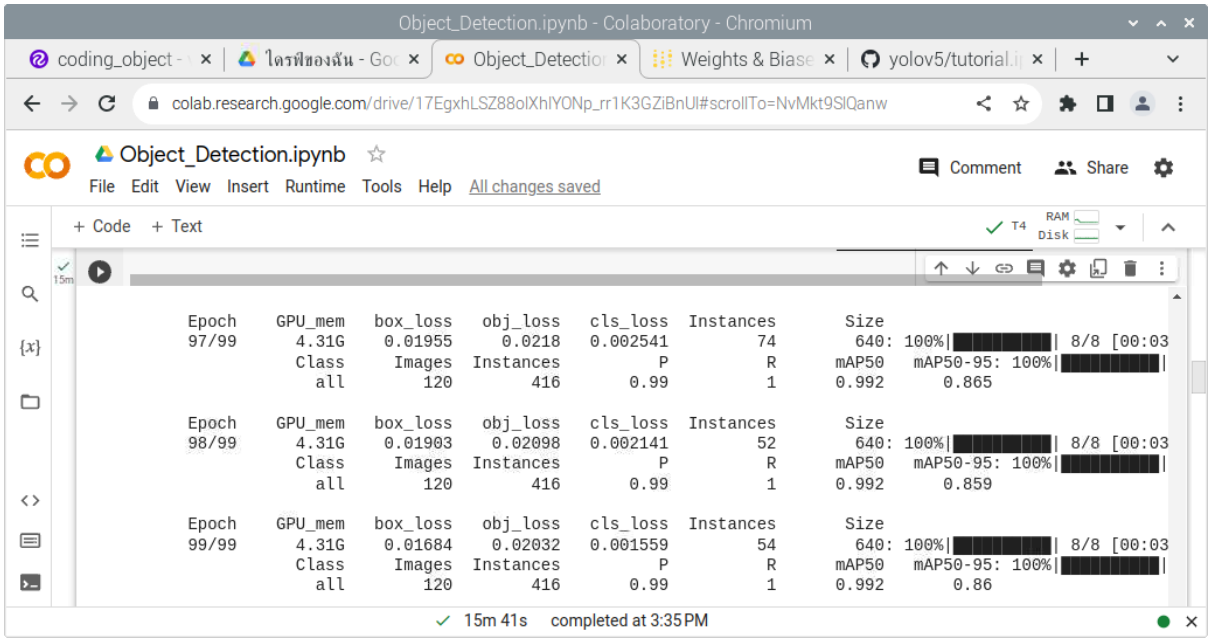


4.37 รันไฟล์ train.py ซึ่งเป็นโค้ดสำหรับการ Train โดยคลิก + Code แล้วพิมพ์คำสั่งดังรูป

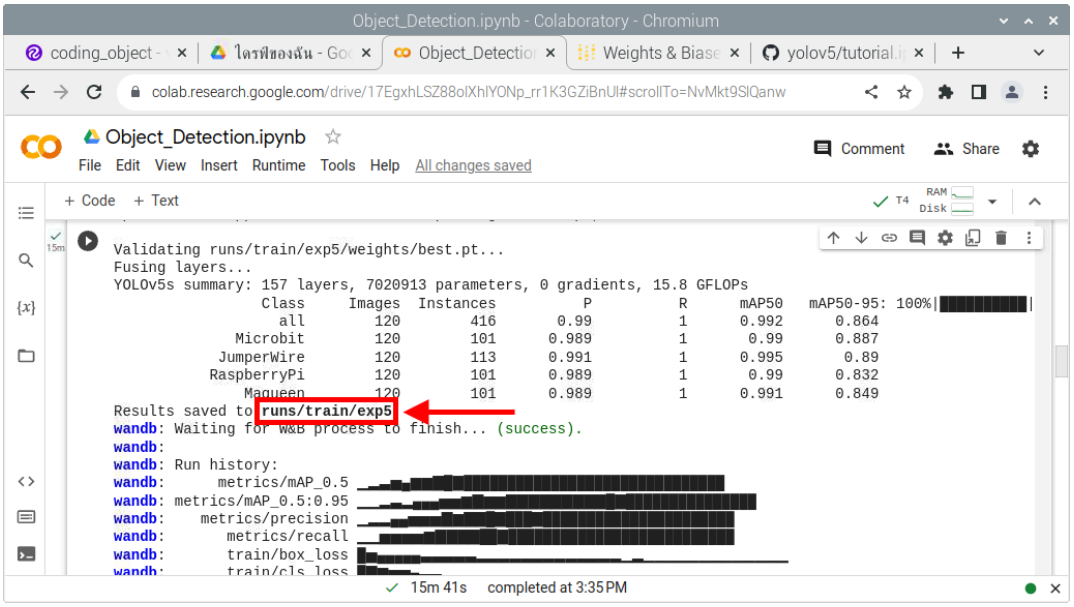


หมายเหตุ: (--mg 640 คือ ขนาดภาพ, --epochs 100 คือ จำนวนครั้งในการ Train, --data coding_train.yaml คือ ไฟล์ Dataset, --cfg ./models/yolov5s.yaml คือ ไฟล์ config, --weights ‘yolov5s.pt’ คือ โมเดลที่ใช้ Train)

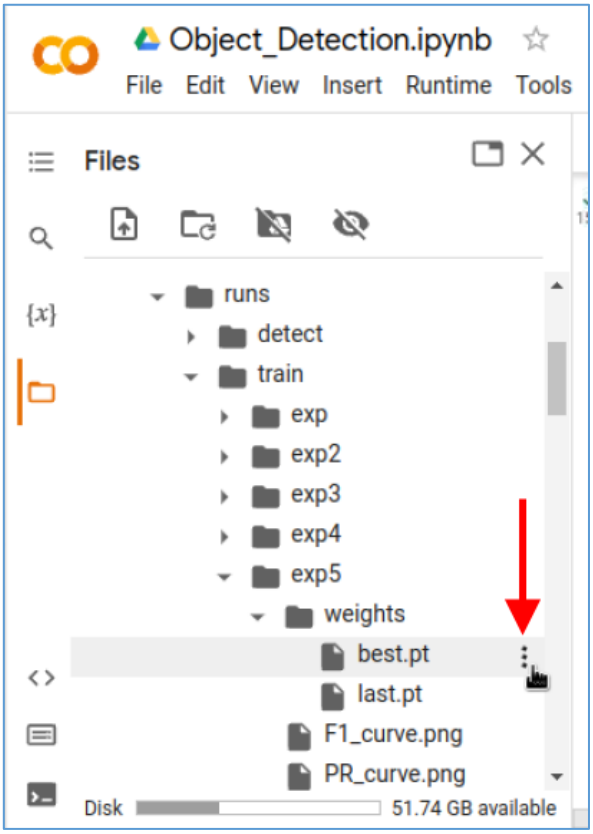
4.38 ระบบจะทำการ Train จำนวน 100 รอบ รอจนการ Train เสร็จสมบูรณ์



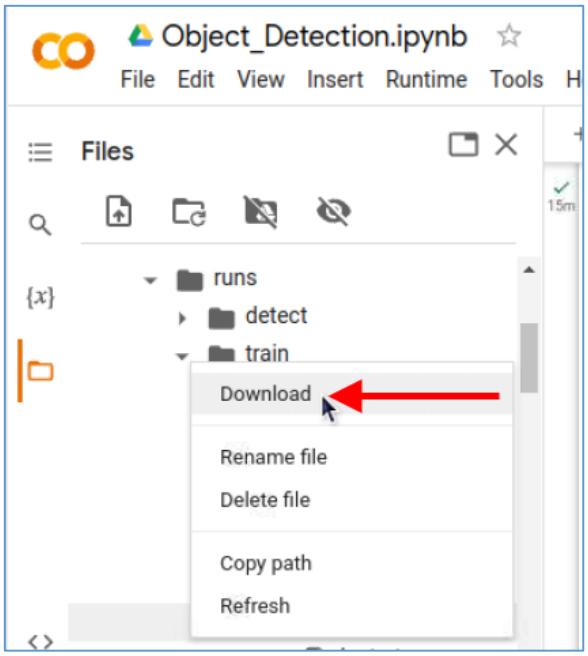
4.39 ผลลัพธ์ที่ได้จะอยู่ในโฟลเดอร์ runs/train/exp



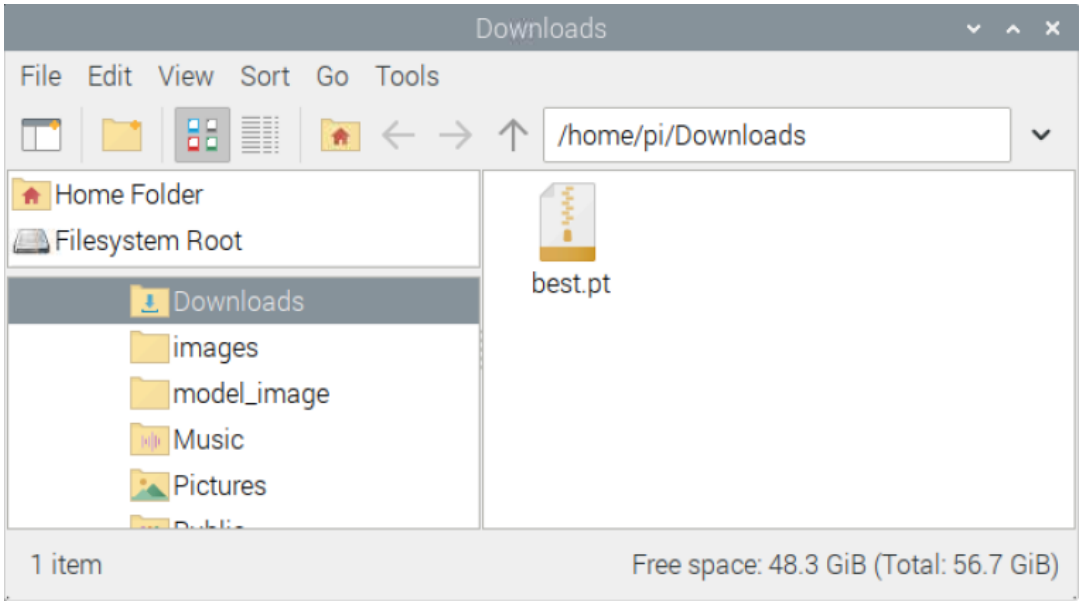
4.40 ดาวน์โหลดไฟล์ที่ทำการ Train สำเร็จ โดยคลิกไปที่โฟลเดอร์ runs/train/exp/weights จากนั้น คลิกจุด 3 จุดที่ไฟล์ best.pt



4.41 คลิก Download

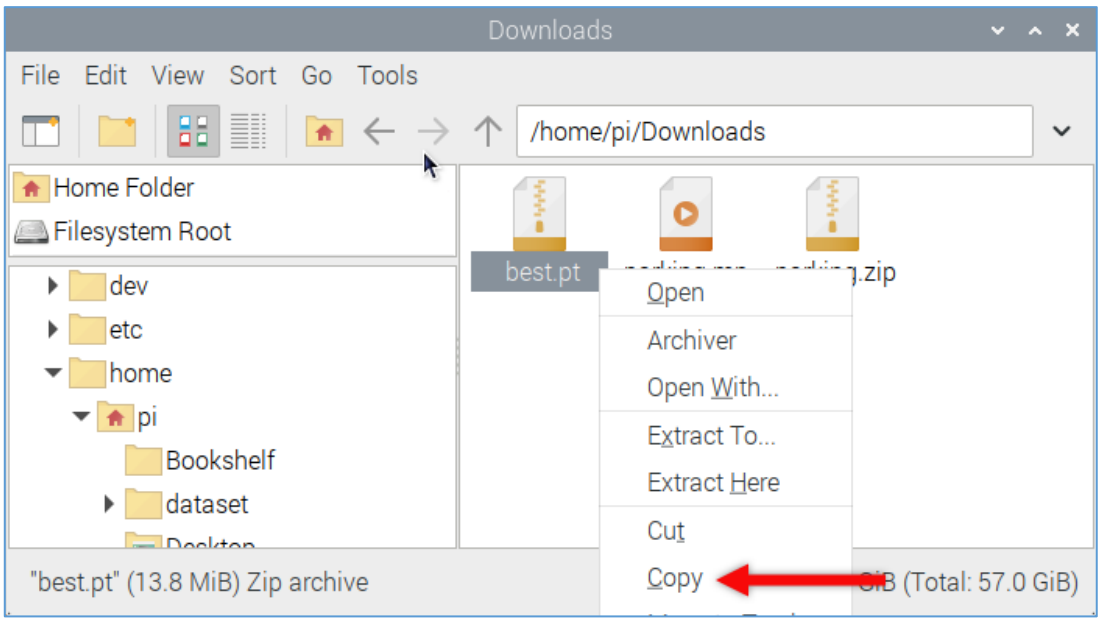


4.42 ไฟล์จะถูกดาวน์โหลดไว้ในโฟลเดอร์ /home/pi/Downloads

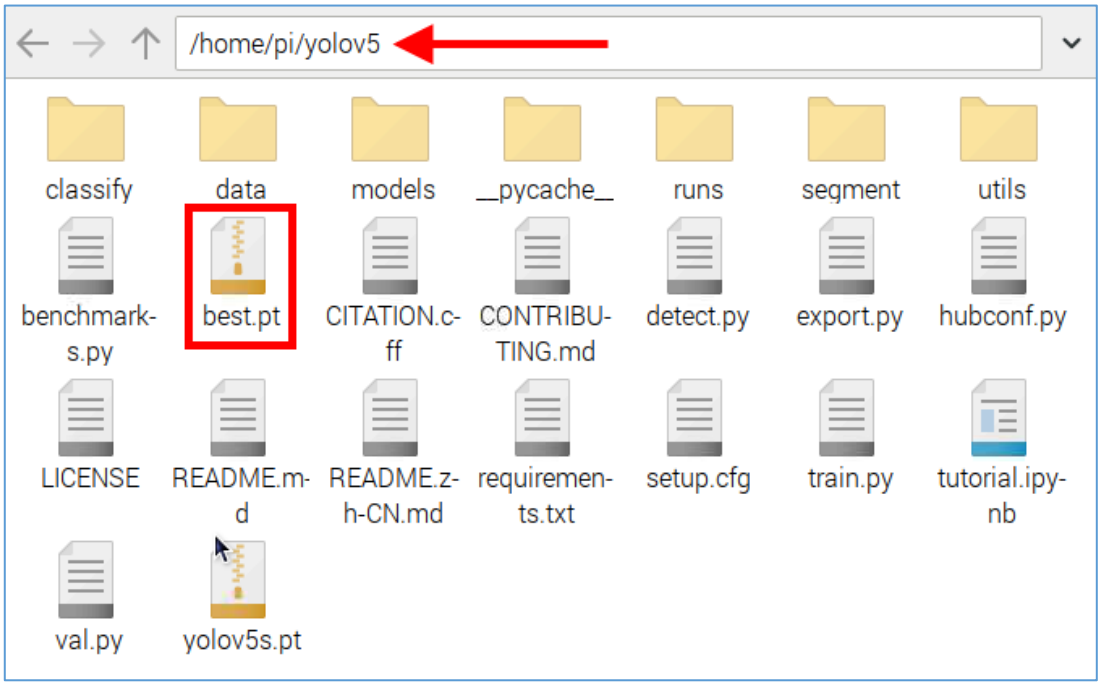


5. นำโมเดลที่ผ่านการ Train มาใช้งาน

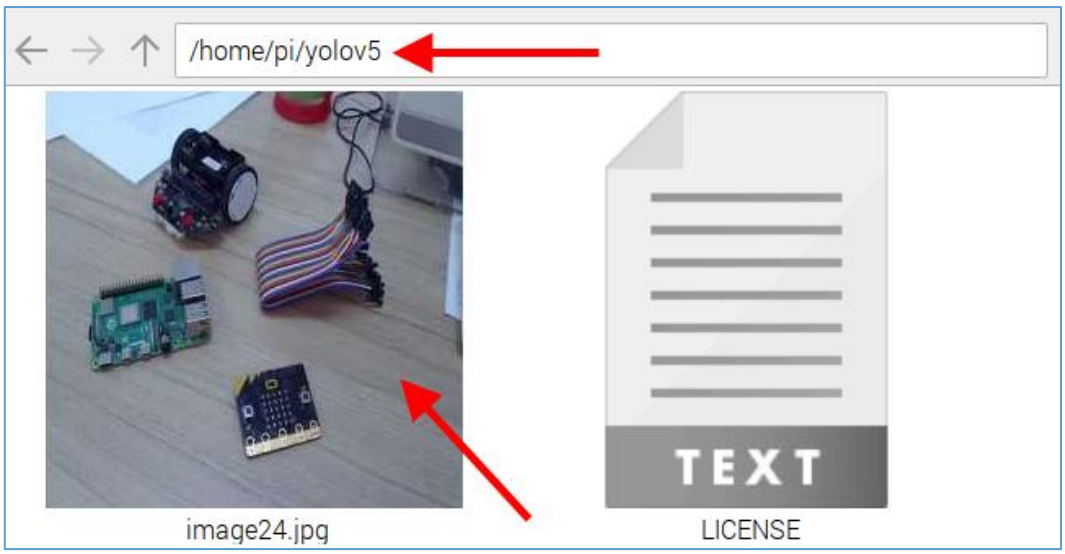
5.1 ไฟล์ที่ผ่านการ Train (best.pt) จะอยู่ในโฟลเดอร์ Downloads ให้คลิกขวาที่ไฟล์แล้วเลือกคำสั่ง Copy



5.2 คัดลอกไฟล์ best.pt มาไว้ในโฟลเดอร์ /home/pi/yolov5

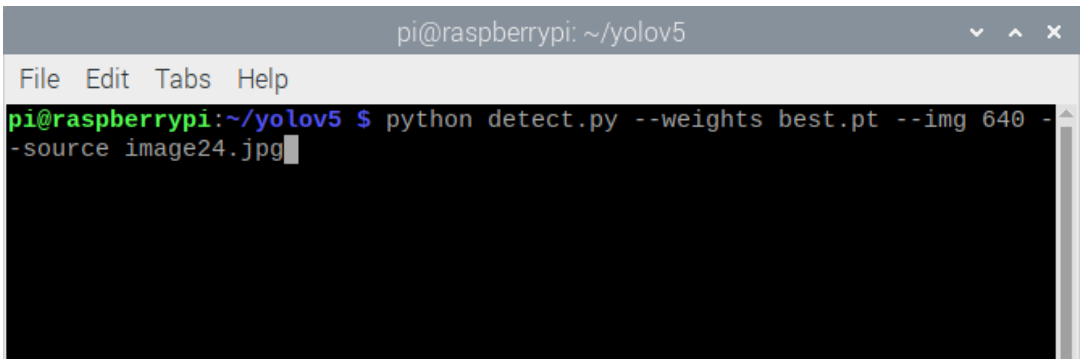


5.3 ทดสอบการตรวจจับวัตถุ โดยคัดลอกรูปภาพในโฟลเดอร์ images มาไว้ในโฟลเดอร์ yolov5 ยกตัวอย่างไฟล์ image24.jpg

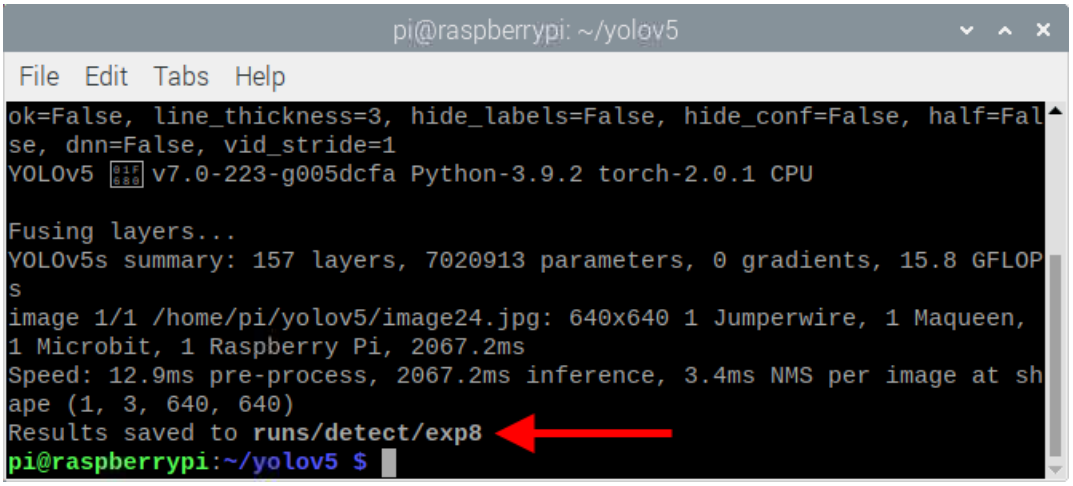


5.4 ทดลอง Classify Object ด้วยรูปภาพ โดยเปิด Terminal แล้วเข้าไปในโฟลเดอร์ yolov5 โดยพิมพ์คำสั่ง cd yolov5

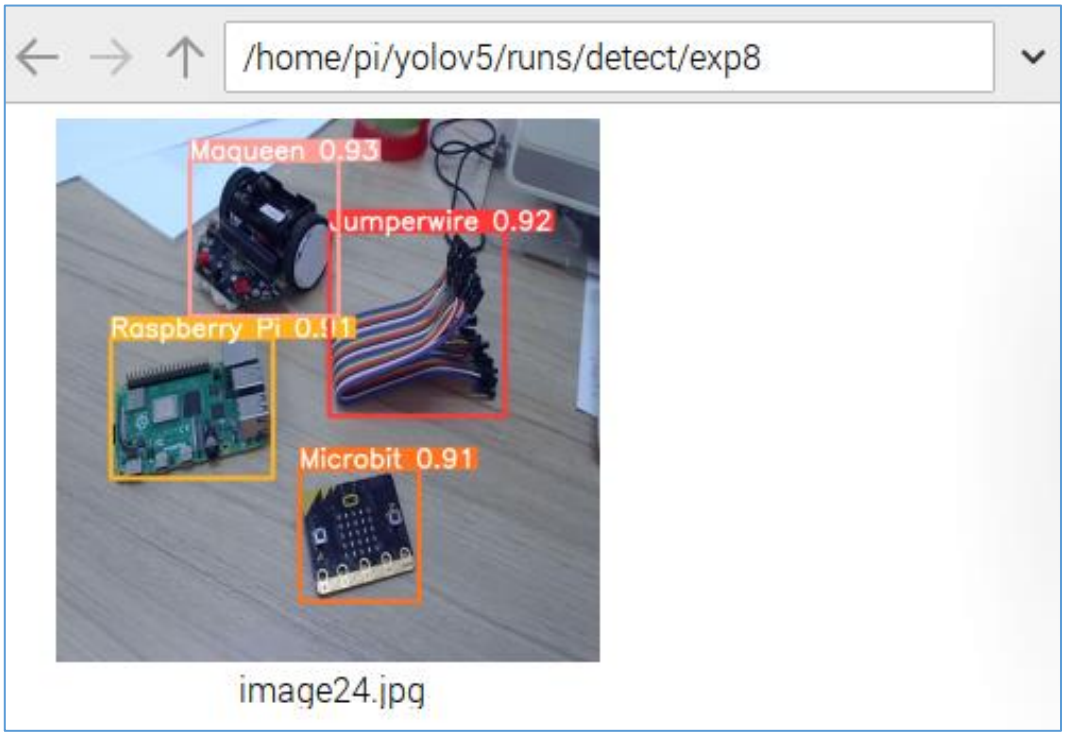
5.5 รันไฟล์ detect.py โดยพิมพ์คำสั่ง python detect.py --weights best.pt --img 640 --source image24.jpg



5.6 ผลลัพธ์จะถูกบันทึกไว้ในโฟลเดอร์ /home/pi/yolov5/runs/detect/exp ดังรูป

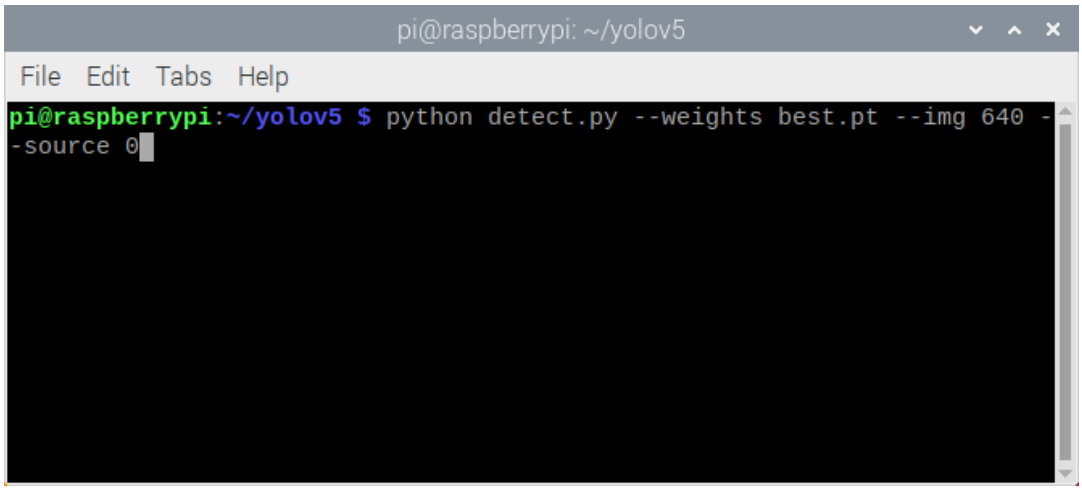


5.7 เข้าไปในโฟลเดอร์ /home/pi/yolov5/runs/detect/exp จะปรากฏรูปภาพผลลัพธ์ ดังรูป

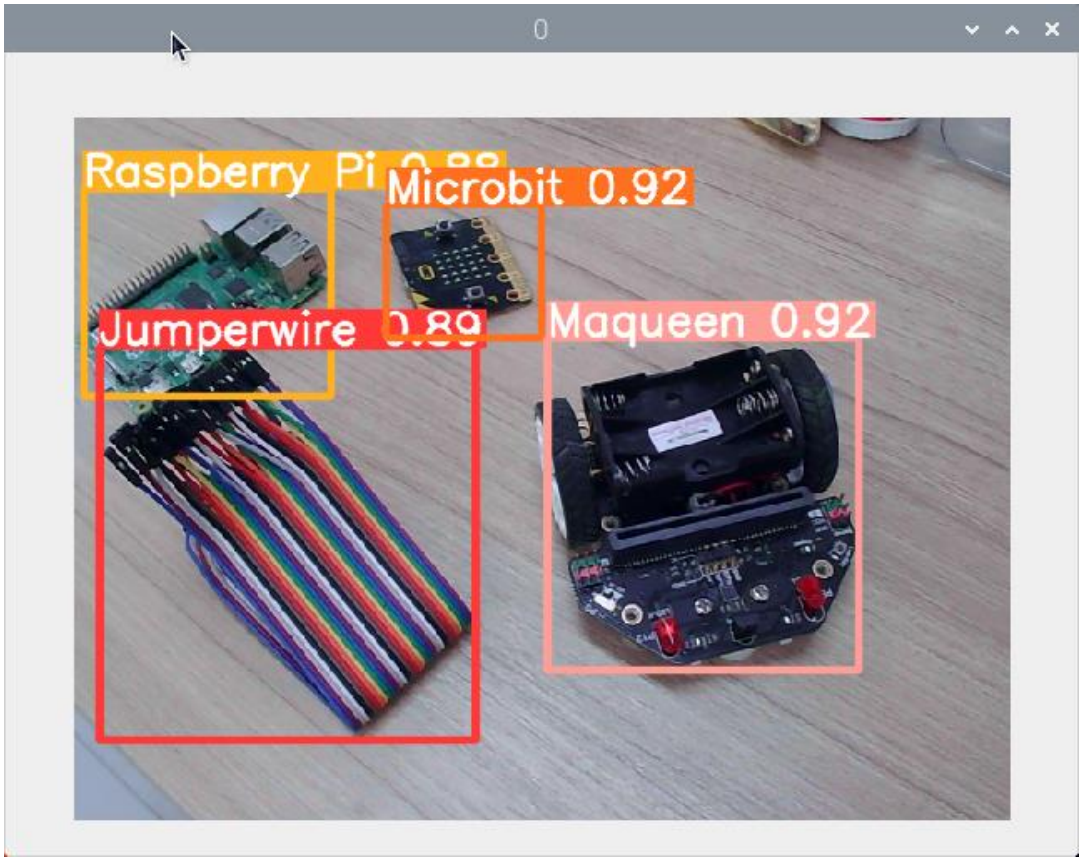


5.8 ทดลองรันโปรแกรมตรวจจับวัตถุ Object Detection กับกล้อง Webcam โดยพิมพ์คำสั่ง

python detect.py --weights best.pt --img 640 --source 0



5.9 จะปรากฏหน้าจอกล้องขึ้นมา ให้นำวัตถุไปส่องด้านหน้ากล้องเพื่อตรวจจับวัตถุ



5.10 เขียนโค้ดคำสั่งตรวจเช็คพื้นที่ว่างบนลานจอด

count_car_parking.py

```python
import cv2
import torch
import numpy as np
import time

points=[]
def POINTS(event, x, y, flags, param):
    if event == cv2.EVENT_MOUSEMOVE:
        colorsBGR = [x, y]
        print(colorsBGR)
cv2.namedWindow('FRAME')
cv2.setMouseCallback('FRAME', POINTS)

model = torch.hub.load('ultralytics/yolov5', 'yolov5s', pretrained=True)
cap = cv2.VideoCapture('parking.mp4')

area = [(4, 234), (4, 310), (480, 255), (426, 202)]
while True:
    ret,frame = cap.read()
    if not ret:
        break
    frame = cv2.resize(frame,(640,360))

    results = model(frame)
    list = []
    for index, row in results.pandas().xyxy[0].iterrows():
        x1 = int(row['xmin'])
        y1 = int(row['ymin'])
        x2 = int(row['xmax'])
        y2 = int(row['ymax'])
        d = (row['name'])
        cx = int(x1 + x2) // 2
        cy = int(y1 + y2) // 2

        if 'car' in d:
            results = cv2.pointPolygonTest(np.array(area, np.int32), ((cx, cy)), False)
            if results >= 0:
                #print(results)
                cv2.rectangle(frame,(x1, y1), (x2, y2), (0, 0, 255), 3)
                cv2.putText(frame, str(d), (x1, y1), cv2.FONT_HERSHEY_PLAIN, 2, (255, 0, 0), 2)

                list.append([cx])
    cv2.polylines(frame, [np.array(area, np.int32)], True, (0, 255, 0), 2)
    a = (len(list))
    print(a)
    cv2.putText(frame, "Car = " + str(a), (50, 50), cv2.FONT_HERSHEY_PLAIN, 2, (255, 0, 0), 2)
    cv2.putText(frame, "Empty Parking = " + str(12 - a), (50, 90), cv2.FONT_HERSHEY_PLAIN, 2, (0, 255, 0), 2)
    cv2.imshow("FRAME", frame)
    cv2.setMouseCallback("FRAME", POINTS)

    #time.sleep(0.1)
    if cv2.waitKey(1) & 0xFF == 27:
        break
cap.release()
cap.destroyAllWindows()
```

หมายเหตุ : หากใช้โมเดลที่เทรนเอง จะต้องเปลี่ยนคำสั่งในบรรทัดที่ 14 เป็น

model = torch.hub.load(‘ultralytics/yolov5’, ‘custom’, ‘best.pt’, force_reload=True)