Image Detection By Yolo v8

In [1]:
```
!pip install ultralytics # install package for import YOLO v8
```

```
Collecting ultralytics
  Downloading ultralytics-8.0.203-py3-none-any.whl (644 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 644.8/644.8 kB 9.6 MB/s eta 0:00:00a 0:00:01
Requirement already satisfied: matplotlib>=3.3.0 in /opt/conda/lib/python3.10/site-packages (from ultralytics) (3.6.3)
Requirement already satisfied: numpy>=1.22.2 in /opt/conda/lib/python3.10/site-packages (from ultralytics) (1.23.5)
Requirement already satisfied: opencv-python>=4.6.0 in /opt/conda/lib/python3.10/site-packages (from ultralytics) (4.7.
0.72)
Requirement already satisfied: pillow>=7.1.2 in /opt/conda/lib/python3.10/site-packages (from ultralytics) (9.5.0)
Requirement already satisfied: pyyaml>=5.3.1 in /opt/conda/lib/python3.10/site-packages (from ultralytics) (5.4.1)
Requirement already satisfied: requests>=2.23.0 in /opt/conda/lib/python3.10/site-packages (from ultralytics) (2.28.2)
Requirement already satisfied: scipy>=1.4.1 in /opt/conda/lib/python3.10/site-packages (from ultralytics) (1.10.1)
Requirement already satisfied: torch>=1.8.0 in /opt/conda/lib/python3.10/site-packages (from ultralytics) (2.0.0)
Requirement already satisfied: torchvision>=0.9.0 in /opt/conda/lib/python3.10/site-packages (from ultralytics) (0.15.
1)
Requirement already satisfied: tqdm>=4.64.0 in /opt/conda/lib/python3.10/site-packages (from ultralytics) (4.64.1)
Requirement already satisfied: pandas>=1.1.4 in /opt/conda/lib/python3.10/site-packages (from ultralytics) (1.5.3)
Requirement already satisfied: seaborn>=0.11.0 in /opt/conda/lib/python3.10/site-packages (from ultralytics) (0.12.2)
Requirement already satisfied: psutil in /opt/conda/lib/python3.10/site-packages (from ultralytics) (5.9.3)
Requirement already satisfied: py-cpuinfo in /opt/conda/lib/python3.10/site-packages (from ultralytics) (9.0.0)
Collecting thop>=0.1.1 (from ultralytics)
  Downloading thop-0.1.1.post2209072238-py3-none-any.whl (15 kB)
Requirement already satisfied: contourpy>=1.0.1 in /opt/conda/lib/python3.10/site-packages (from matplotlib>=3.3.0->ult
ralytics) (1.0.7)
Requirement already satisfied: cycler>=0.10 in /opt/conda/lib/python3.10/site-packages (from matplotlib>=3.3.0->ultraly
tics) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in /opt/conda/lib/python3.10/site-packages (from matplotlib>=3.3.0->ul
tralytics) (4.39.3)
Requirement already satisfied: kiwisolver>=1.0.1 in /opt/conda/lib/python3.10/site-packages (from matplotlib>=3.3.0->ul
tralytics) (1.4.4)
Requirement already satisfied: packaging>=20.0 in /opt/conda/lib/python3.10/site-packages (from matplotlib>=3.3.0->ultr
alytics) (21.3)
Requirement already satisfied: pyparsing>=2.2.1 in /opt/conda/lib/python3.10/site-packages (from matplotlib>=3.3.0->ult
ralytics) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in /opt/conda/lib/python3.10/site-packages (from matplotlib>=3.3.0-
>ultralytics) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /opt/conda/lib/python3.10/site-packages (from pandas>=1.1.4->ultralytic
s) (2023.3)
Requirement already satisfied: charset-normalizer<4,>=2 in /opt/conda/lib/python3.10/site-packages (from requests>=2.2
3.0->ultralytics) (2.1.1)
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/lib/python3.10/site-packages (from requests>=2.23.0->ultralyt
ics) (3.4)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /opt/conda/lib/python3.10/site-packages (from requests>=2.23.0-
>ultralytics) (1.26.15)
Requirement already satisfied: certifi>=2017.4.17 in /opt/conda/lib/python3.10/site-packages (from requests>=2.23.0->ul
tralytics) (2023.5.7)
Requirement already satisfied: filelock in /opt/conda/lib/python3.10/site-packages (from torch>=1.8.0->ultralytics) (3.
12.0)
Requirement already satisfied: typing-extensions in /opt/conda/lib/python3.10/site-packages (from torch>=1.8.0->ultraly
tics) (4.5.0)
Requirement already satisfied: sympy in /opt/conda/lib/python3.10/site-packages (from torch>=1.8.0->ultralytics) (1.12)
Requirement already satisfied: networkx in /opt/conda/lib/python3.10/site-packages (from torch>=1.8.0->ultralytics) (3.
1)
Requirement already satisfied: jinja2 in /opt/conda/lib/python3.10/site-packages (from torch>=1.8.0->ultralytics) (3.1.
2)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.10/site-packages (from python-dateutil>=2.7->matplotl
ib>=3.3.0->ultralytics) (1.16.0)
Requirement already satisfied: MarkupSafe>=2.0 in /opt/conda/lib/python3.10/site-packages (from jinja2->torch>=1.8.0->u
ltralytics) (2.1.2)
Requirement already satisfied: mpmath>=0.19 in /opt/conda/lib/python3.10/site-packages (from sympy->torch>=1.8.0->ultra
lytics) (1.3.0)
Installing collected packages: thop, ultralytics
Successfully installed thop-0.1.1.post2209072238 ultralytics-8.0.203
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system pack
age manager. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv
```

In [2]:
```
from ultralytics import YOLO

import warnings
warnings.filterwarnings('ignore')
```

Select type of model YOLO v8 for pre_train

In [3]:
```
model = YOLO("yolov8m.pt")
```

```
Downloading https://github.com/ultralytics/assets/releases/download/v0.0.0/yolov8m.pt to 'yolov8m.pt'...
100%|████████| 49.7M/49.7M [00:00<00:00, 75.2MB/s]
```

All YOLOv8 models for object detection ship already pre-trained on the COCO dataset

In this section we use that and in next part we create custome dataset

Types of model for pre_training in YOLO v8

Classification	Detection	Segmentation	Kind
yolov8n-cls.pt	yolov8n.pt	yolov8n-seg.pt	Nano
yolov8s-cls.pt	yolov8s.pt	yolov8s-seg.pt	Small
yolov8m-cls.pt	yolov8m.pt	yolov8m-seg.pt	Medium
yolov8l-cls.pt	yolov8l.pt	yolov8l-seg.pt	Large
yolov8x-cls.pt	yolov8x.pt	yolov8x-seg.pt	Huge

Fetch image

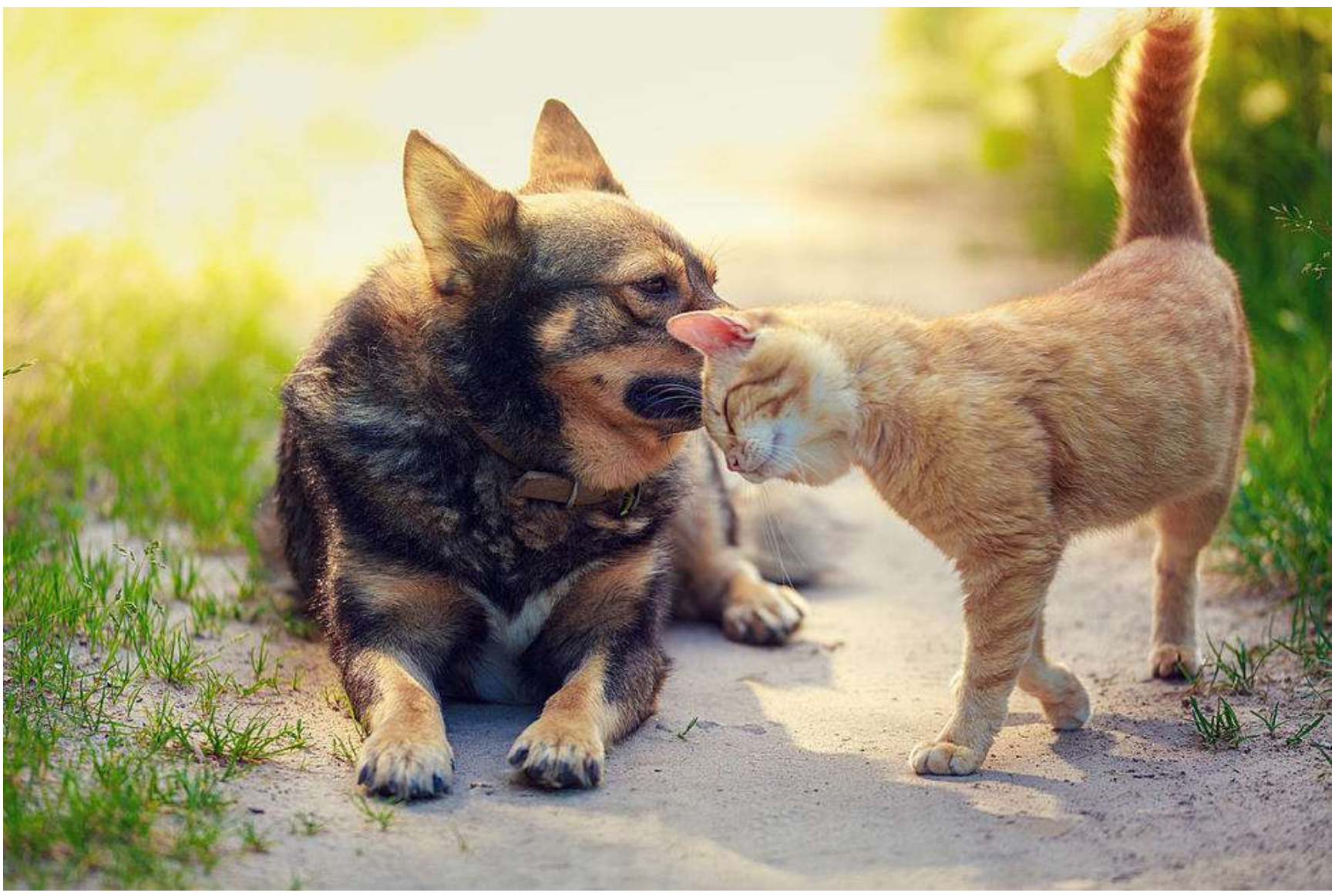
In [4]:
```
!pip install pillow
```

```
Requirement already satisfied: pillow in /opt/conda/lib/python3.10/site-packages (9.5.0)
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system pack
age manager. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv
```

In [5]:
```
from PIL import Image
```

In [6]:
```
url = "/input/images/pet.jpg"
img = Image.open(url)
img
```

Out[6]:



In [7]:
```
width, height = img.size
width, height
```

Out[7]:
```
(980, 653)
```

Predict Image

In [8]:
```
results = model.predict(url)
```

```
image 1/1 /kaggle/input/images/pet.jpg: 448x640 1 cat, 1 dog, 64.9ms
Speed: 14.9ms preprocess, 64.9ms inference, 23.2ms postprocess per image at shape (1, 3, 448, 640)
```

In [9]:
```
result = results[0]
```

The result contains detected objects and convenient properties to work with them. The most important one is the boxes array with information about detected bounding boxes on the image. You can determine how many objects it detected by running the len function

In [10]:
```
len(result.boxes)
```

Out[10]:
```
2
```

In [11]:
```
result.boxes
```

Out[11]:
```
ultralytics.engine.results.Boxes object with attributes:

cls: tensor([15., 16.], device='cuda:0')
conf: tensor([0.9443, 0.5302], device='cuda:0')
data: tensor([[4.8877e+02, 9.7979e-01, 9.2547e+02, 5.7025e+02, 9.4435e-01, 1.5000e+01],
        [1.9688e+02, 6.4756e+01, 5.7043e+02, 5.8792e+02, 5.3020e-01, 1.6000e+01]], device='cuda:0')
id: None
is_track: False
orig_shape: (653, 980)
shape: torch.Size([2, 6])
xywh: tensor([[707.1191, 285.6148, 436.7036, 569.2700],
        [383.6522, 326.3361, 373.5458, 523.1594]], device='cuda:0')
xywhn: tensor([[0.7216, 0.4374, 0.4456, 0.8718],
        [0.3915, 0.4997, 0.3812, 0.8012]], device='cuda:0')
xyxy: tensor([[488.7673,   0.9798, 925.4709, 570.2498],
        [196.8793,  64.7564, 570.4250, 587.9158]], device='cuda:0')
xyxyn: tensor([[0.4987, 0.0015, 0.9444, 0.8733],
        [0.2009, 0.0992, 0.5821, 0.9003]], device='cuda:0')
```

- xyxy – the coordinates of the box as an array [x0,y0,x1,y1]
- cls – the ID of object type
- conf – the confidence level of the model about this object

In [12]:
```
print("Object type:", result.boxes.cls)
print("Coordinates:", result.boxes.xyxy)
print("Probability:", result.boxes.conf)
```

```
Object type: tensor([15., 16.], device='cuda:0')
Coordinates: tensor([[488.7673,   0.9798, 925.4709, 570.2498],
        [196.8793,  64.7564, 570.4250, 587.9158]], device='cuda:0')
Probability: tensor([0.9443, 0.5302], device='cuda:0')
```

the PyTorch models are encoded as an array of PyTorch Tensor objects, so you need to extract the first item from each of these arrays

Example: first class from Object type

In [13]:
```
print("Object type:", result.boxes.cls[0])
print("Coordinates:", result.boxes.xyxy[0])
print("Probability:", result.boxes.conf[0])
```

```
Object type: tensor(15., device='cuda:0')
Coordinates: tensor([488.7673,   0.9798, 925.4709, 570.2498], device='cuda:0')
Probability: tensor(0.9443, device='cuda:0')
```

- To unpack actual values from Tensor, you need to use the .tolist() method for tensors with array
- o unpack actual values from Tensor, you need to use the .item() method

In [14]:
```
cords =result.boxes.xyxy[0].tolist()
class_id =result.boxes.cls[0].item()
conf = result.boxes.conf[0].item()

print("Object type:", class_id)
print("Coordinates:", cords)
print("Probability:", conf)
```

```
Object type: 15.0
Coordinates: [488.76727294921875, 0.9797887802124023, 925.4708862304688, 570.249755859375]
Probability: 0.9443464875221252
```

The object type is 15 here. What does this mean?

you can find that 15 is "cat"

In [15]:
```
result.names[15]
```

Out[15]:
```
'cat'
```

In [16]:
```
cords =result.boxes.xyxy[0].tolist()
class_id =result.boxes.cls[0].item()
conf = result.boxes.conf[0].item()

print("Object type:", result.names[class_id]) # Change Index to class_id istead of 15
print("Coordinates:", cords)

print("Probability:", conf)
```

```
Object type: cat
Coordinates: [488.76727294921875, 0.9797887802124023, 925.4708862304688, 570.249755859375]
Probability: 0.9443464875221252
```

Show All of images that detected

In [17]:
```
for box in result.boxes:
    cords =box.xyxy[0].tolist()
    class_id =box.cls[0].item()
    conf = box.conf[0].item()

    print("Object type:",result.names[class_id],"(",class_id,")")
    print("Coordinates:", [round(cord,2) for cord in cords])
    print("Probability:", round(conf,2))
    print("------------------")
```

```
Object type: cat ( 15.0 )
Coordinates: [488.77, 0.98, 925.47, 570.25]
Probability: 0.94
------------------
Object type: dog ( 16.0 )
Coordinates: [196.88, 64.76, 570.43, 587.92]
Probability: 0.53
------------------
```

In [18]:
```
import cv2
import numpy as np
```

In [19]:
```
image = np.array(img) # convert image to array
for box in result.boxes:
    cords =box.xyxy[0].tolist()
    class_id =box.cls[0].item()
    conf = box.conf[0].item()

    start = (int(cords[0]),int(cords[1])) # x0, y0
    end   = (int(cords[2]),int(cords[3])) # x1, y1

    cv2.rectangle(image,start,end,(0,200,0),thickness=2)
    cv2.putText(image, result.names[class_id], (start[0]+15,start[1]+30), cv2.FONT_HERSHEY_SIMPLEX,0.9, (10,0,10), 2)#
```

In [20]:
```
Image.fromarray(image)
```

Out[20]: