Mini Project Report

Green Flash

Author:

Saleh Mohammed Alanazi

Anas Ahmed Alwohaib

Khuzama Khaled Alsnan

Alyaa Ali Bajaber

Supervisor: **Hani Elshafey**

1- YOLO.ipynb

Import Required Libraries

```
#import tensorflow and keras
import tensorflow as tf
from keras import layers, models, datasets

from ultralytics import YOLO
import os
from IPython import display
from roboflow import Roboflow
```

+ Code + Text

```
%pip install roboflow
%pip install ultralytics
%pip install ipython
```

```
Requirement already satisfied: roboflow in /usr/local/lib/python3.10/dist-packages (1.1.39)
Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-packages (from roboflow) (2024.7.4)
Requirement already satisfied: idna==3.7 in /usr/local/lib/python3.10/dist-packages (from roboflow) (3.7)
Requirement already satisfied: cycler in /usr/local/lib/python3.10/dist-packages (from roboflow) (0.12.1)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.10/dist-packages (from roboflow) (1.4.5)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from roboflow) (3.7.1)
Requirement already satisfied: numpy>=1.18.5 in /usr/local/lib/python3.10/dist-packages (from roboflow) (1.26.4)
Requirement already satisfied: opencv-python-headless==4.10.0.84 in /usr/local/lib/python3.10/dist-packages (from roboflow) (4.10.0.84)
Requirement already satisfied: Pillow>=7.1.2 in /usr/local/lib/python3.10/dist-packages (from roboflow) (9.4.0)
Requirement already satisfied: python-dateutil in /usr/local/lib/python3.10/dist-packages (from roboflow) (2.8.2)
Requirement already satisfied: python-dotenv in /usr/local/lib/python3.10/dist-packages (from roboflow) (1.0.1)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from roboflow) (2.32.3)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from roboflow) (1.16.0)
Requirement already satisfied: urllib3>=1.26.6 in /usr/local/lib/python3.10/dist-packages (from roboflow) (2.0.7)
Requirement already satisfied: tqdm>=4.41.0 in /usr/local/lib/python3.10/dist-packages (from roboflow) (4.66.5)
Requirement already satisfied: PyYAML>=5.3.1 in /usr/local/lib/python3.10/dist-packages (from roboflow) (6.0.2)
Requirement already satisfied: requests-toolbelt in /usr/local/lib/python3.10/dist-packages (from roboflow) (1.0.0)
Requirement already satisfied: filetype in /usr/local/lib/python3.10/dist-packages (from roboflow) (1.2.0)
```

Data Collection

```
rf = Roboflow(api_key="6Mtasz76njBtqTQgOjjr")
project = rf.workspace("cardetetction").project("car-jgm0v")
version = project.version(2)
dataset = version.download("yolov8")
```

```
loading Roboflow workspace...
loading Roboflow project...
Dependency ultralytics==8.0.196 is required but found version=8.2.77, to fix: `pip install ultralytics==8.0.196`
```

Data Preprocessing

```
display.clear_output()
display.clear_output()
```

Model selection, implementation, training

```
!yolo task =detect mode=train model=yolov8n.pt data={dataset.location}/data.yaml epochs=50 imgsz=640 optimizer=Adam lr0=0.001
```

```
Ultralytics YOLOv8.2.77 🚀 Python-3.10.12 torch-2.3.1+cu121 CUDA:0 (Tesla T4, 15102MiB)
engine/trainer: task=detect, mode=train, model=yolov8n.pt, data=/content/Car-2/data.yaml, epochs=50, time=None, patience=100, batch=16, imgsz=640, save=True, save_period
Downloading https://ultralytics.com/assets/Arial.ttf to '/root/.config/Ultralytics/Arial.ttf'...
100% 755k/755k [00:00<00:00, 26.2MB/s]
Overriding model.yaml nc=80 with nc=3

                   from  n    params  module                                       arguments
  0                  -1  1       464  ultralytics.nn.modules.conv.Conv             [3, 16, 3, 2]
  1                  -1  1      4672  ultralytics.nn.modules.conv.Conv             [16, 32, 3, 2]
  2                  -1  1      7360  ultralytics.nn.modules.block.C2f             [32, 32, 1, True]
  3                  -1  1     18560  ultralytics.nn.modules.conv.Conv             [32, 64, 3, 2]
  4                  -1  2     49664  ultralytics.nn.modules.block.C2f             [64, 64, 2, True]
```

Model selection, implementation, training

```
!yolo task =detect mode=train model=yolov8n.pt data={dataset.location}/data.yaml epochs=50 imgsz=640 optimizer=Adam lr0=0.001
```

```
                   all        280       3292      0.582      0.687      0.651      0.565

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      43/50       2.3G     0.5094     0.3913     0.8406        107        640: 100% 94/94 [00:31<00:00,  3.02it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 9/9 [00:02<00:00,  3.15it/s]
                   all        280       3292      0.577      0.655      0.633      0.545

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      44/50       2.3G      0.503     0.3853     0.8394        112        640: 100% 94/94 [00:30<00:00,  3.11it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 9/9 [00:02<00:00,  3.69it/s]
                   all        280       3292      0.621      0.658      0.648      0.561

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      45/50      2.16G     0.4974      0.377      0.837        100        640: 100% 94/94 [00:30<00:00,  3.07it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 9/9 [00:02<00:00,  3.60it/s]
                   all        280       3292      0.632      0.638      0.655      0.567

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      46/50      2.27G       0.49     0.3703      0.833        102        640: 100% 94/94 [00:31<00:00,  3.02it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 9/9 [00:02<00:00,  3.66it/s]
                   all        280       3292      0.612      0.692      0.658      0.566

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      47/50      2.16G     0.4904     0.3663     0.8354        128        640: 100% 94/94 [00:29<00:00,  3.21it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 9/9 [00:02<00:00,  3.25it/s]
                   all        280       3292      0.633      0.694      0.662      0.573

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      48/50      2.16G     0.4838       0.36      0.832        130        640: 100% 94/94 [00:26<00:00,  3.54it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 9/9 [00:04<00:00,  2.09it/s]
                   all        280       3292      0.637      0.659       0.65      0.562

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      49/50      2.16G     0.4802     0.3528     0.8306        125        640: 100% 94/94 [00:27<00:00,  3.43it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 9/9 [00:02<00:00,  3.82it/s]
                   all        280       3292      0.602      0.682      0.654       0.57
```

Model Testing

```
!yolo task =detect mode=val model=/content/runs/detect/train3/weights/best.pt source={dataset.location}/test/images
```

```
WARNING ⚠️ 'data' argument is missing. Using default 'data=coco8.yaml'.
Ultralytics YOLOv8.2.77 🚀 Python-3.10.12 torch-2.3.1+cu121 CUDA:0 (Tesla T4, 15102MiB)
Model summary (fused): 168 layers, 3,006,233 parameters, 0 gradients, 8.1 GFLOPs

Dataset 'coco8.yaml' images not found ⚠️, missing path '/content/datasets/coco8/images/val'
Downloading https://ultralytics.com/assets/coco8.zip to '/content/datasets/coco8.zip'...
100% 433k/433k [00:00<00:00, 16.2MB/s]
Unzipping /content/datasets/coco8.zip to /content/datasets/coco8...: 100% 25/25 [00:00<00:00, 4670.93file/s]
Dataset download success ✅ (0.6s), saved to /content/datasets
```

Model Evaluation

```
[ ] !yolo task =detect mode=predict model=/content/runs/detect/train3/weights/best.pt source={dataset.location}/test/images
```

```
image 166/220 /content/Car-2/test/images/pagi_16112021_mp4-197_jpg.rf.525c32236b5f4d51e6460ac7fab9b4f9.jpg: 640x640 3 Trucks, 9 carss, 8.9ms
image 167/220 /content/Car-2/test/images/pagi_16112021_mp4-210_jpg.rf.3b64be2863d4b80e9768ae9a465e1ba5.jpg: 640x640 1 Truck, 10 carss, 6.1ms
image 168/220 /content/Car-2/test/images/pagi_16112021_mp4-215_jpg.rf.7e2160d1bccefbe0e55953f08d3cd8b2.jpg: 640x640 1 Truck, 13 carss, 6.2ms
image 169/220 /content/Car-2/test/images/pagi_16112021_mp4-217_jpg.rf.adc043c888a9dbb3a2b3967fc998859b.jpg: 640x640 1 Truck, 12 carss, 8.0ms
image 170/220 /content/Car-2/test/images/pagi_16112021_mp4-221_jpg.rf.987a05a1fa84740b290f9dc606bc73a4.jpg: 640x640 2 Trucks, 14 carss, 6.0ms
image 171/220 /content/Car-2/test/images/pagi_16112021_mp4-226_jpg.rf.a547ec4f508eaaa3893e95e979dbca3f.jpg: 640x640 2 Trucks, 10 carss, 6.3ms
image 172/220 /content/Car-2/test/images/pagi_16112021_mp4-227_jpg.rf.b4542f77f3d93db9837d72d0db62d629.jpg: 640x640 2 Trucks, 10 carss, 6.6ms
image 173/220 /content/Car-2/test/images/pagi_16112021_mp4-228_jpg.rf.0fe354d651d05c1dd266ded001fcd0cd.jpg: 640x640 2 Trucks, 10 carss, 6.0ms
image 174/220 /content/Car-2/test/images/pagi_16112021_mp4-230_jpg.rf.2b0ed27be40717fc7318b933865deb8a.jpg: 640x640 3 Trucks, 11 carss, 6.0ms
image 175/220 /content/Car-2/test/images/pagi_16112021_mp4-232_jpg.rf.4030457ce39bb93769838177a7dc8d1f.jpg: 640x640 3 Trucks, 10 carss, 6.0ms
image 176/220 /content/Car-2/test/images/pagi_16112021_mp4-234_jpg.rf.8483006ec7c07a0b2a35243b3a2233e3.jpg: 640x640 3 Trucks, 11 carss, 7.2ms
image 177/220 /content/Car-2/test/images/pagi_16112021_mp4-237_jpg.rf.2f553ddaf6e1ee9de4205fa9c7cbf5ea.jpg: 640x640 3 Trucks, 10 carss, 6.0ms
image 178/220 /content/Car-2/test/images/pagi_16112021_mp4-241_jpg.rf.23992ee58bbc112060d301d5c05aff6a.jpg: 640x640 3 Trucks, 10 carss, 6.3ms
image 179/220 /content/Car-2/test/images/pagi_16112021_mp4-247_jpg.rf.1150b712b7e441ac9cf4cf152a23dc1f.jpg: 640x640 4 Trucks, 9 carss, 6.2ms
image 180/220 /content/Car-2/test/images/pagi_16112021_mp4-248_jpg.rf.dc36b337b4e60185fbe53bd7509f198b.jpg: 640x640 2 Trucks, 8 carss, 6.1ms
```

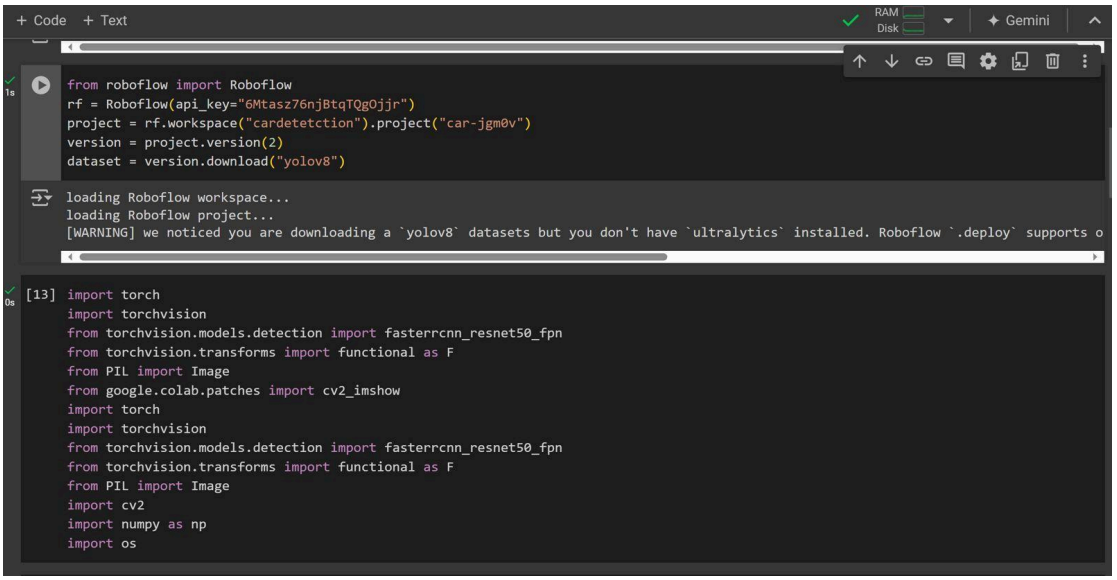
```
!pip install roboflow
```

```
Collecting roboflow
  Downloading roboflow-1.1.39-py3-none-any.whl.metadata (9.4 kB)
Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-packages (from roboflow) (2024.7.4)
Requirement already satisfied: idna==3.7 in /usr/local/lib/python3.10/dist-packages (from roboflow) (3.7)
```

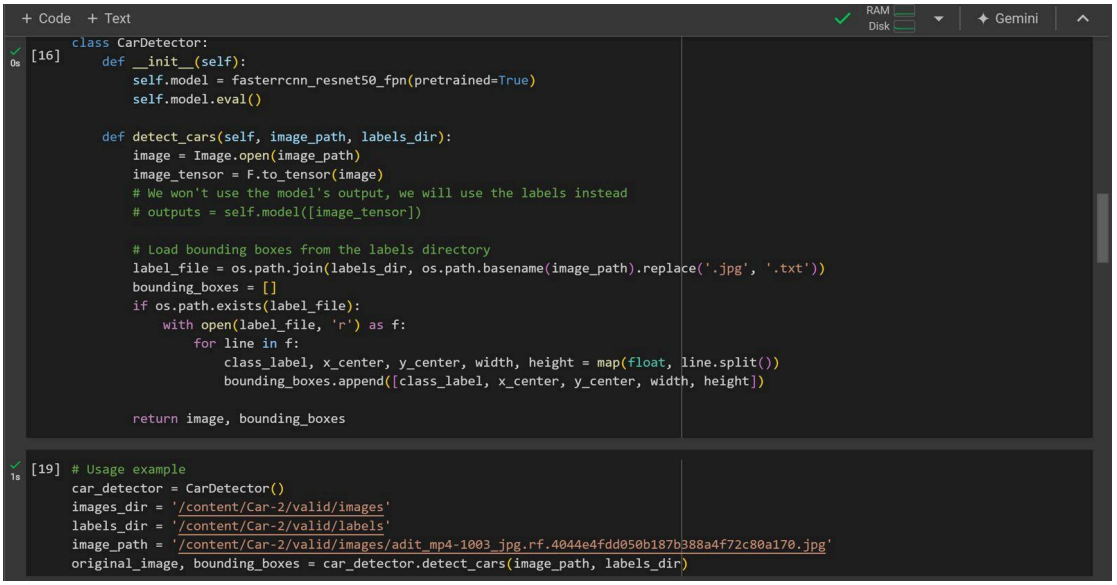
```
from roboflow import Roboflow
rf = Roboflow(api_key="6Mtasz76njBtqTQgOjjr")
project = rf.workspace("cardetetction").project("car-jgm0v")
version = project.version(2)
dataset = version.download("yolov8")
```

```
loading Roboflow workspace...
loading Roboflow project...
[WARNING] we noticed you are downloading a `yolov8` datasets but you don't have `ultralytics` installed. Roboflow `.deploy` supports only models trained with `ultralytics=
Downloading Dataset Version Zip in Car-2 to yolov8:: 100%|██████████| 121587/121587 [00:05<00:00, 20354.86it/s]

Extracting Dataset Version Zip to Car-2 in yolov8:: 100%|██████████| 4012/4012 [00:01<00:00, 2239.38it/s]
```

2- CarCounterDetection.ipynb



```
+ Code + Text                                                    RAM Disk   Gemini

1s
from roboflow import Roboflow
rf = Roboflow(api_key="6Mtasz76njBtqTQgOjjr")
project = rf.workspace("cardetetction").project("car-jgm0v")
version = project.version(2)
dataset = version.download("yolov8")

loading Roboflow workspace...
loading Roboflow project...
[WARNING] we noticed you are downloading a `yolov8` datasets but you don't have `ultralytics` installed. Roboflow `.deploy` supports o

0s
[13] import torch
     import torchvision
     from torchvision.models.detection import fasterrcnn_resnet50_fpn
     from torchvision.transforms import functional as F
     from PIL import Image
     from google.colab.patches import cv2_imshow
     import torch
     import torchvision
     from torchvision.models.detection import fasterrcnn_resnet50_fpn
     from torchvision.transforms import functional as F
     from PIL import Image
     import cv2
     import numpy as np
     import os
```



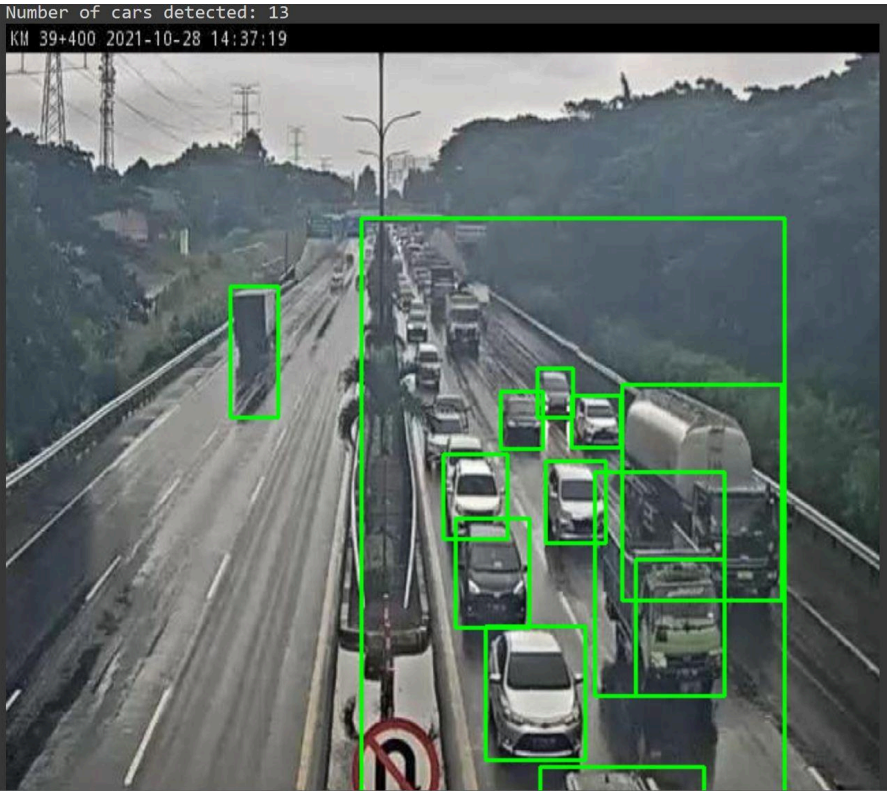
```
+ Code + Text                                                    RAM Disk   Gemini

0s
[16] class CarDetector:
         def __init__(self):
             self.model = fasterrcnn_resnet50_fpn(pretrained=True)
             self.model.eval()

         def detect_cars(self, image_path, labels_dir):
             image = Image.open(image_path)
             image_tensor = F.to_tensor(image)
             # We won't use the model's output, we will use the labels instead
             # outputs = self.model([image_tensor])

             # Load bounding boxes from the labels directory
             label_file = os.path.join(labels_dir, os.path.basename(image_path).replace('.jpg', '.txt'))
             bounding_boxes = []
             if os.path.exists(label_file):
                 with open(label_file, 'r') as f:
                     for line in f:
                         class_label, x_center, y_center, width, height = map(float, line.split())
                         bounding_boxes.append([class_label, x_center, y_center, width, height])

             return image, bounding_boxes

1s
[19] # Usage example
     car_detector = CarDetector()
     images_dir = '/content/Car-2/valid/images'
     labels_dir = '/content/Car-2/valid/labels'
     image_path = '/content/Car-2/valid/images/adit_mp4-1003_jpg.rf.4044e4fdd050b187b388a4f72c80a170.jpg'
     original_image, bounding_boxes = car_detector.detect_cars(image_path, labels_dir)
```

```
[17]  # Draw bounding boxes from label files
      image_width, image_height = original_image.size
      for box in bounding_boxes:
          _, x_center, y_center, width, height = box
          x1 = int((x_center - width / 2) * image_width)
          y1 = int((y_center - height / 2) * image_height)
          x2 = int((x_center + width / 2) * image_width)
          y2 = int((y_center + height / 2) * image_height)
          cv2.rectangle(image_np, (x1, y1), (x2, y2), (0, 255, 0), 2)
```

```
[18]  # Convert the PIL Image to a NumPy array for OpenCV
      image_np = np.array(original_image)
      # Convert RGB to BGR
      image_np = cv2.cvtColor(image_np, cv2.COLOR_RGB2BGR)

      # Draw bounding boxes from label files
      image_width, image_height = original_image.size

      # Counter for the number of cars detected
      car_count = 0

      for box in bounding_boxes:
          _, x_center, y_center, width, height = box
          x1 = int((x_center - width / 2) * image_width)
          y1 = int((y_center - height / 2) * image_height)
          x2 = int((x_center + width / 2) * image_width)
          y2 = int((y_center + height / 2) * image_height)
          cv2.rectangle(image_np, (x1, y1), (x2, y2), (0, 255, 0), 2)
          car_count += 1
```

```
[18]  # Print the total number of cars detected
      print(f"Number of cars detected: {car_count}")

      # Display the image with bounding boxes
      cv2_imshow(image_np)
```

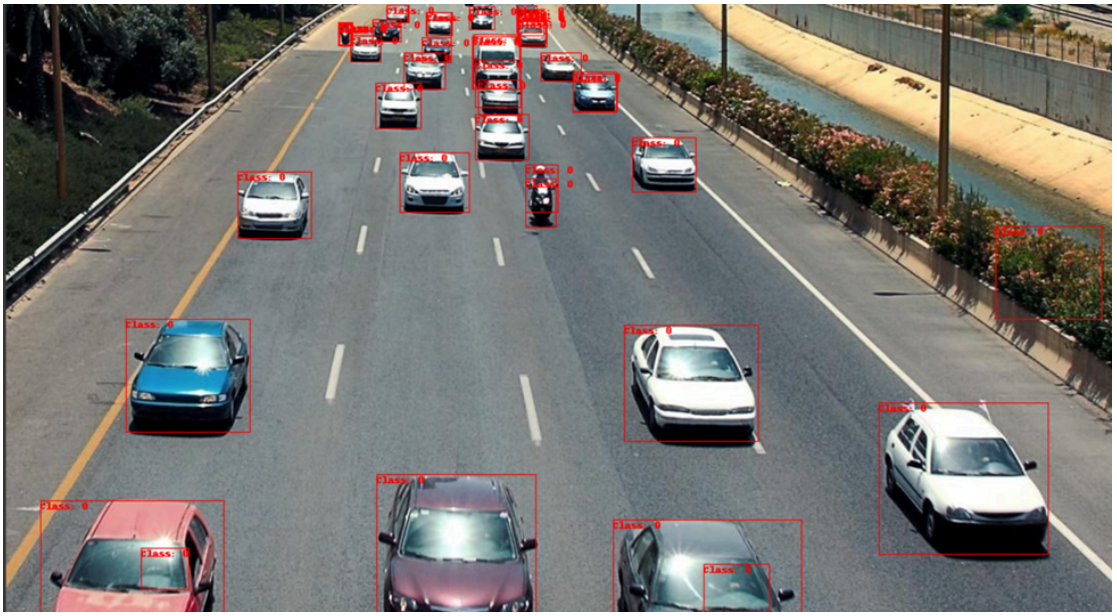
```
[18]  Number of cars detected: 13
```



using faster RCNN

```
[66] class CarDetector:
        def __init__(self):
            self.model = fasterrcnn_resnet50_fpn(pretrained=True)
            self.model.eval()

        def detect_cars(self, image_path):
            image = Image.open(image_path)
            image_tensor = F.to_tensor(image)
            outputs = self.model([image_tensor])

            bounding_boxes = []
            class_0_count = 0  # Counter for class 0
            for box in outputs[0]['boxes']:
                x_min, y_min, x_max, y_max = box.tolist()
                class_label = 0  # Assuming class label 0 for cars
                if class_label == 0:
                    class_0_count += 1
                x_center = (x_min + x_max) / 2
                y_center = (y_min + y_max) / 2
                width = x_max - x_min
                height = y_max - y_min
                bounding_boxes.append([class_label, x_center, y_center, width, height])

            return image, bounding_boxes, class_0_count
```

```
# Example usage
detector = CarDetector()
image_path = "/content/27094_3063d356a3a54cc3859537fd23c5ba9d_1539205710.jpeg"
image, bounding_boxes, class_0_count = detector.detect_cars(image_path)
draw_bounding_boxes(image, bounding_boxes)
print("Number of class 0 labels detected:", class_0_count)
```

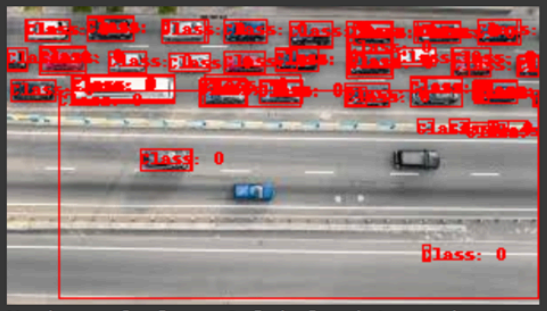


```
Number of class 0 labels detected: 44
```

```
[68] # Example usage
     detector = CarDetector()
     image_path = "//content/download.jfif"
     image, bounding_boxes, class_0_count = detector.detect_cars(image_path)
     draw_bounding_boxes(image, bounding_boxes)
     print("Number of class 0 labels detected:", class_0_count)
```

```
Number of class 0 labels detected: 94
```

```
     # Example usage
     detector = CarDetector()
     image_path = "/content/images.jfif"
     image, bounding_boxes, class_0_count = detector.detect_cars(image_path)
     draw_bounding_boxes(image, bounding_boxes)
     print("Number of class 0 labels detected:", class_0_count)
```



```
Number of class 0 labels detected: 65
```

3- Drive_behavior_extra.ipynb

Import Required Libraries

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import BaggingClassifier, RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.metrics import accuracy_score
import joblib
import numpy as np
```

Load and Preprocess the Dataset

```
[ ] df = pd.read_csv('/content/Drive_behavior-Dataset.csv')
    df
```

	AccX	AccY	AccZ	GyroX	GyroY	GyroZ	Class	Timestamp
0	0.758194	-0.217791	0.457263	0.000000	0.000000	0.000000	AGGRESSIVE	818922
1	0.667560	-0.038610	0.231416	-0.054367	-0.007712	0.225257	AGGRESSIVE	818923
2	2.724449	-7.584121	2.390926	0.023824	0.013668	-0.038026	AGGRESSIVE	818923
3	2.330950	-7.621754	2.529024	0.056810	-0.180587	-0.052076	AGGRESSIVE	818924
4	2.847215	-6.755621	2.224640	-0.031765	-0.035201	0.035277	AGGRESSIVE	818924
...	...	...	...	...	...	...	...	...
6723	0.915688	-2.017489	1.687505	0.450360	0.384845	-1.236468	SLOW	3583789

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6728 entries, 0 to 6727
Data columns (total 8 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   AccX       6728 non-null   float64
 1   AccY       6728 non-null   float64
 2   AccZ       6728 non-null   float64
 3   GyroX      6728 non-null   float64
 4   GyroY      6728 non-null   float64
 5   GyroZ      6728 non-null   float64
 6   Class      6728 non-null   object
 7   Timestamp  6728 non-null   int64
dtypes: float64(6), int64(1), object(1)
memory usage: 420.6+ KB
```

Summry Statistices

```
df.describe()
```

	AccX	AccY	AccZ	GyroX	GyroY	GyroZ	Timestamp
count	6728.000000	6728.000000	6728.000000	6728.000000	6728.000000	6728.000000	6.728000e+03
mean	0.070613	-0.100175	0.025112	0.002326	-0.000024	0.009564	2.316251e+06
std	0.978118	0.904262	0.990935	0.066789	0.123828	0.114710	1.376747e+06
min	-4.854163	-7.621754	-7.143998	-0.751822	-1.587028	-1.236468	8.189220e+05
25%	-0.478331	-0.605450	-0.519489	-0.026267	-0.050702	-0.028253	8.199088e+05
50%	0.029885	-0.087355	0.028696	0.001374	-0.001222	0.002367	3.581780e+06
75%	0.587433	0.423664	0.563334	0.030085	0.047877	0.038332	3.582777e+06
max	5.864980	4.308813	5.564037	1.490511	1.707598	1.190500	3.583791e+06

Encode categorical variables using Label Encoder

```
label_encoder = LabelEncoder()
df['Class'] = label_encoder.fit_transform(df['Class'])
```

Split the Dataset

```
[ ] X = df.drop('Class', axis=1)
    y = df['Class']

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=42)
```

Scale the features

```
[ ] scaler = StandardScaler()
    X = scaler.fit_transform(X)
```

Implement a Random Forest model

```
[ ] random_forest_classifier = RandomForestClassifier(n_estimators=50, random_state=42)
    random_forest_classifier.fit(X_train, y_train)
```

```
RandomForestClassifier
RandomForestClassifier(n_estimators=50, random_state=42)
```

Evaluate the Model

```
[ ] predictions = random_forest_classifier.predict(X_test)

    accuracy = accuracy_score(y_test, predictions)
    print(f'Random Forest Model Accuracy: {accuracy * 100:.2f}%')
```

```
Random Forest Model Accuracy: 99.78%
```

Save The Model

```
joblib.dump(random_forest_classifier, 'model.pkl')
model = joblib.load('model.pkl')
```