1. Importing Libraries:
   1. os & glob for accessing operating system paths and performing operations
   2. matplotlib for viewing the images as they can’t be viewed via opencv in jupyter notebook
   3. cv2 to read and perform operations on the image
   4. requests to download data from internet via url
   5. numpy for mathematical operations
2. Setting Constant Values:

2.1 Setting seed to a random value for getting different results everytime the notebook is executed

* 1. Setting TRAIN = True for the execution of train function
  2. Setting EPOCHS = 50 to train the model on the dataset for 50 epochs
  3. Downloading the dataset from roboflow using api

1. Defined a function to download files from the internet via url.
2. Defined a yolo2bbox function to convert bounding boxes in YOLO format to a normal opencv format.
3. Defined a plot\_box function:

5.1 It will have image, bboxed and labels as parameters.

5.2 Iterating through the bboxes and normalize the coordinates.

5.3 Draw rectangles on the image using opencv.

1. Defined a plot function to extract the images, labels from the image and label path.

6.1 Store the images and labels from their path.

6.2 Sort the images and labels.

6.3 Get the number of images.

6.4 Iterate through the images and read the image.

6.5 open the label file of the image and extract the bounding box and coordinates.

6.6 pass the image coordinates, labels to plot\_box to draw the rectangle.

1. Defined a monitor\_tensorboard function to load the tensorboard for viewing the results.
2. Check if yolov5 repo already exists in the file folder. If not then clone the repository:
3. Install the requirements.txt to make the system ready for yolo
4. Train the model on custom dataset having the following parameters.

--data => path of training images

--wieghts => which yolov5 model to be used

--img => image size

--epochs => no of iterations

--batch\_size => how much data should be passed in one batch

--name => directory name where results will be stored

1. Function to set\_resdir to create directory for storing results.
2. Function show\_valid\_results to show validation set predictions by the model saved during training.
3. We take snaps from the video frames i.e 2 frames per second instead of 60 to make data short and save them in directory
4. We run detect.py and pass the above saved results of snaps and detec.py will be modified to target only those who have 80% confidence and their bounding box area is greater than 144600 and smaller than 200000. Detect.py will save in the RES\_DIR the image with bounding box as well as the cropped image of only the object inside the bbox.