Bounding box detection - Racoon data

Data files

- images_racoon.rar: contain images of racoons
- train_labels.cv: contains coordinates for bounding box for every image

Import the necessary libraries

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947
Enter your authorization code:
··········
Mounted at /content/drive
```

```
import os
import tensorflow as tf
import csv
import numpy as np
from PIL import Image
import pandas as pd

from keras import Model
from keras.applications.mobilenet import MobileNet, preprocess_input
from keras.callbacks import ModelCheckpoint, EarlyStopping, ReduceLROnPlateau, Callback
from keras.layers import Conv2D, Reshape
from keras.utils import Sequence
from keras.backend import epsilon
```

```
The default version of TensorFlow in Colab will soon switch to TensorFlow 2.x.
We recommend you upgrade now or ensure your notebook will continue to use TensorFlow 1.x via the %t
Using TensorFlow backend.
```

Change directory

```
os.chdir('/content/drive/My Drive/Colab Notebooks/Labs')
```

Load the training data from train.csv file

```
with open('train_labels.csv', 'r') as csvfile:
  reader = csv.reader(csvfile, delimiter=',')
```

```
  for row in reader:
    print(row)
```

```
['filename', 'width', 'height', 'class', 'xmin', 'ymin', 'xmax', 'ymax']
['raccoon-17.jpg', '259', '194', 'raccoon', '95', '60', '167', '118']
['raccoon-11.jpg', '660', '432', 'raccoon', '3', '1', '461', '431']
['raccoon-63.jpg', '600', '400', 'raccoon', '74', '107', '280', '290']
['raccoon-63.jpg', '600', '400', 'raccoon', '227', '93', '403', '298']
['raccoon-60.jpg', '273', '185', 'raccoon', '58', '33', '197', '127']
['raccoon-69.jpg', '205', '246', 'raccoon', '12', '11', '188', '240']
['raccoon-180.jpg', '600', '400', 'raccoon', '119', '21', '368', '399']
['raccoon-200.jpg', '261', '193', 'raccoon', '107', '10', '249', '166']
['raccoon-141.jpg', '249', '202', 'raccoon', '1', '1', '154', '176']
['raccoon-19.jpg', '259', '194', 'raccoon', '87', '8', '182', '89']
['raccoon-84.jpg', '303', '166', 'raccoon', '31', '6', '197', '163']
['raccoon-124.jpg', '259', '194', 'raccoon', '17', '39', '239', '147']
['raccoon-182.jpg', '500', '500', 'raccoon', '17', '122', '279', '499']
['raccoon-111.jpg', '768', '960', 'raccoon', '41', '5', '683', '917']
['raccoon-91.jpg', '236', '314', 'raccoon', '22', '14', '216', '308']
['raccoon-79.jpg', '640', '425', 'raccoon', '120', '1', '568', '425']
['raccoon-93.jpg', '251', '201', 'raccoon', '66', '29', '233', '190']
['raccoon-20.jpg', '720', '540', 'raccoon', '2', '29', '720', '503']
['raccoon-42.jpg', '577', '1024', 'raccoon', '121', '206', '410', '767']
['raccoon-139.jpg', '259', '194', 'raccoon', '20', '6', '177', '167']
['raccoon-58.jpg', '224', '225', 'raccoon', '2', '1', '199', '221']
['raccoon-71.jpg', '640', '426', 'raccoon', '129', '51', '628', '373']
['raccoon-183.jpg', '2000', '1333', 'raccoon', '358', '21', '1354', '1119']
['raccoon-1.jpg', '650', '417', 'raccoon', '81', '88', '522', '408']
['raccoon-169.jpg', '615', '409', 'raccoon', '194', '1', '549', '409']
['raccoon-82.jpg', '750', '500', 'raccoon', '6', '1', '632', '500']
['raccoon-4.jpg', '275', '183', 'raccoon', '21', '11', '200', '183']
['raccoon-101.jpg', '640', '426', 'raccoon', '86', '53', '400', '356']
['raccoon-10.jpg', '450', '495', 'raccoon', '130', '2', '446', '488']
['raccoon-166.jpg', '328', '154', 'raccoon', '108', '31', '208', '120']
['raccoon-184.jpg', '640', '640', 'raccoon', '81', '77', '567', '617']
['raccoon-38.jpg', '259', '194', 'raccoon', '7', '17', '257', '180']
['raccoon-120.jpg', '660', '371', 'raccoon', '129', '12', '510', '331']
['raccoon-142.jpg', '1024', '768', 'raccoon', '171', '162', '811', '740']
['raccoon-149.jpg', '500', '375', 'raccoon', '132', '50', '305', '246']
['raccoon-51.jpg', '800', '599', 'raccoon', '315', '105', '772', '540']
['raccoon-43.jpg', '480', '360', 'raccoon', '1', '65', '239', '316']
['raccoon-123.jpg', '640', '406', 'raccoon', '280', '42', '550', '392']
['raccoon-66.jpg', '860', '484', 'raccoon', '220', '37', '697', '440']
['raccoon-9.jpg', '347', '510', 'raccoon', '10', '7', '347', '471']
['raccoon-178.jpg', '275', '183', 'raccoon', '59', '12', '242', '180']
['raccoon-47.jpg', '262', '193', 'raccoon', '34', '4', '233', '193']
['raccoon-167.jpg', '259', '195', 'raccoon', '1', '5', '175', '195']
['raccoon-54.jpg', '602', '339', 'raccoon', '78', '5', '517', '333']
['raccoon-77.jpg', '640', '360', 'raccoon', '161', '1', '627', '330']
['raccoon-155.jpg', '259', '194', 'raccoon', '46', '91', '143', '169']
['raccoon-89.jpg', '259', '194', 'raccoon', '18', '6', '225', '176']
['raccoon-153.jpg', '700', '700', 'raccoon', '10', '1', '612', '700']
['raccoon-179.jpg', '600', '450', 'raccoon', '1', '176', '270', '427']
['raccoon-115.jpg', '426', '640', 'raccoon', '51', '130', '351', '556']
['raccoon-64.jpg', '259', '194', 'raccoon', '1', '1', '247', '194']
['raccoon-56.jpg', '240', '210', 'raccoon', '20', '6', '224', '201']
['raccoon-44.jpg', '300', '168', 'raccoon', '45', '14', '247', '165']
['raccoon-39.jpg', '250', '172', 'raccoon', '54', '12', '250', '166']
['raccoon-26.jpg', '306', '374', 'raccoon', '114', '5', '306', '337']
['raccoon-162.jpg', '259', '194', 'raccoon', '45', '34', '161', '184']
['raccoon-170.jpg', '259', '194', 'raccoon', '53', '27', '254', '173']
['raccoon-187.jpg', '362', '357', 'raccoon', '161', '112', '292', '276']
['raccoon-131.jpg', '259', '194', 'raccoon', '1', '1', '199', '184']
['raccoon-174.jpg', '960', '639', 'raccoon', '125', '43', '588', '527']
```

```
['raccoon-92.jpg', '960', '640', 'raccoon', '37', '32', '729', '543']
['raccoon-193.jpg', '634', '852', 'raccoon', '23', '215', '440', '831']
['raccoon-138.jpg', '259', '194', 'raccoon', '56', '54', '226', '150']
['raccoon-157.jpg', '220', '229', 'raccoon', '1', '1', '144', '209']
['raccoon-108.jpg', '604', '481', 'raccoon', '99', '53', '402', '464']
['raccoon-116.jpg', '660', '432', 'raccoon', '3', '1', '436', '430']
['raccoon-117.jpg', '640', '448', 'raccoon', '100', '124', '266', '324']
['raccoon-117.jpg', '640', '448', 'raccoon', '342', '101', '570', '297']
['raccoon-12.jpg', '259', '194', 'raccoon', '28', '21', '126', '181']
['raccoon-12.jpg', '259', '194', 'raccoon', '85', '33', '235', '193']
['raccoon-16.jpg', '424', '640', 'raccoon', '51', '178', '355', '632']
['raccoon-90.jpg', '640', '426', 'raccoon', '44', '90', '577', '426']
['raccoon-160.jpg', '256', '197', 'raccoon', '7', '42', '162', '197']
['raccoon-75.jpg', '640', '640', 'raccoon', '1', '1', '640', '459']
['raccoon-199.jpg', '640', '428', 'raccoon', '28', '64', '530', '402']
['raccoon-97.jpg', '500', '393', 'raccoon', '1', '32', '343', '307']
['raccoon-188.jpg', '460', '379', 'raccoon', '26', '71', '366', '334']
['raccoon-21.jpg', '290', '174', 'raccoon', '59', '2', '216', '171']
['raccoon-35.jpg', '275', '183', 'raccoon', '1', '1', '164', '183']
['raccoon-85.jpg', '620', '465', 'raccoon', '236', '87', '598', '429']
['raccoon-49.jpg', '640', '395', 'raccoon', '162', '36', '611', '395']
['raccoon-86.jpg', '600', '401', 'raccoon', '129', '34', '475', '401']
['raccoon-34.jpg', '259', '194', 'raccoon', '1', '2', '227', '194']
['raccoon-196.jpg', '233', '216', 'raccoon', '83', '87', '211', '211']
['raccoon-96.jpg', '230', '219', 'raccoon', '28', '25', '203', '175']
['raccoon-3.jpg', '720', '480', 'raccoon', '1', '1', '720', '476']
['raccoon-2.jpg', '800', '573', 'raccoon', '60', '51', '462', '499']
['raccoon-52.jpg', '800', '533', 'raccoon', '105', '10', '502', '501']
['raccoon-81.jpg', '600', '450', 'raccoon', '4', '54', '574', '410']
['raccoon-112.jpg', '800', '574', 'raccoon', '131', '174', '775', '563']
['raccoon-18.jpg', '240', '156', 'raccoon', '32', '25', '201', '130']
['raccoon-94.jpg', '700', '467', 'raccoon', '155', '10', '543', '445']
['raccoon-98.jpg', '480', '360', 'raccoon', '108', '31', '351', '308']
['raccoon-83.jpg', '660', '371', 'raccoon', '104', '3', '509', '369']
['raccoon-36.jpg', '640', '428', 'raccoon', '113', '27', '468', '428']
['raccoon-24.jpg', '268', '188', 'raccoon', '77', '48', '179', '156']
['raccoon-24.jpg', '268', '188', 'raccoon', '139', '77', '202', '145']
['raccoon-195.jpg', '225', '225', 'raccoon', '25', '111', '197', '225']
['raccoon-55.jpg', '634', '417', 'raccoon', '6', '49', '250', '320']
['raccoon-55.jpg', '634', '417', 'raccoon', '274', '27', '563', '410']
['raccoon-175.jpg', '634', '381', 'raccoon', '69', '89', '354', '378']
['raccoon-163.jpg', '248', '203', 'raccoon', '6', '7', '240', '157']
['raccoon-48.jpg', '261', '193', 'raccoon', '43', '28', '240', '176']
['raccoon-70.jpg', '500', '375', 'raccoon', '60', '4', '421', '369']
['raccoon-119.jpg', '400', '533', 'raccoon', '16', '62', '362', '353']
['raccoon-119.jpg', '400', '533', 'raccoon', '211', '359', '277', '402']
['raccoon-119.jpg', '400', '533', 'raccoon', '198', '392', '280', '473']
['raccoon-88.jpg', '640', '480', 'raccoon', '116', '41', '526', '436']
['raccoon-61.jpg', '274', '184', 'raccoon', '94', '63', '195', '148']
['raccoon-61.jpg', '274', '184', 'raccoon', '142', '39', '213', '108']
['raccoon-121.jpg', '600', '399', 'raccoon', '55', '34', '416', '377']
['raccoon-74.jpg', '800', '533', 'raccoon', '141', '6', '472', '505']
['raccoon-133.jpg', '490', '640', 'raccoon', '8', '6', '476', '631']
['raccoon-177.jpg', '276', '183', 'raccoon', '8', '18', '157', '178']
['raccoon-177.jpg', '276', '183', 'raccoon', '146', '13', '263', '146']
['raccoon-159.jpg', '226', '223', 'raccoon', '14', '11', '223', '221']
['raccoon-53.jpg', '259', '194', 'raccoon', '71', '45', '197', '171']
['raccoon-132.jpg', '259', '194', 'raccoon', '6', '2', '240', '131']
['raccoon-198.jpg', '259', '194', 'raccoon', '57', '21', '158', '184']
['raccoon-198.jpg', '259', '194', 'raccoon', '112', '32', '199', '158']
['raccoon-103.jpg', '480', '640', 'raccoon', '92', '54', '460', '545']
['raccoon-46.jpg', '576', '318', 'raccoon', '145', '2', '423', '318']
```

```
['raccoon-50.jpg', '275', '183', 'raccoon', '36', '2', '174', '172']
['raccoon-37.jpg', '520', '593', 'raccoon', '13', '1', '500', '592']
['raccoon-161.jpg', '500', '347', 'raccoon', '209', '73', '385', '186']
['raccoon-194.jpg', '1080', '1080', 'raccoon', '1', '63', '885', '1042']
['raccoon-32.jpg', '625', '415', 'raccoon', '88', '92', '473', '328']
['raccoon-15.jpg', '640', '360', 'raccoon', '313', '61', '614', '360']
['raccoon-147.jpg', '426', '640', 'raccoon', '13', '1', '426', '486']
['raccoon-151.jpg', '225', '225', 'raccoon', '42', '94', '108', '224']
['raccoon-80.jpg', '225', '225', 'raccoon', '21', '27', '177', '182']
['raccoon-6.jpg', '480', '360', 'raccoon', '1', '44', '307', '316']
['raccoon-154.jpg', '650', '419', 'raccoon', '148', '56', '517', '346']
['raccoon-135.jpg', '640', '426', 'raccoon', '99', '8', '605', '404']
['raccoon-118.jpg', '448', '297', 'raccoon', '109', '31', '307', '297']
['raccoon-150.jpg', '275', '183', 'raccoon', '80', '62', '187', '169']
['raccoon-102.jpg', '259', '194', 'raccoon', '1', '1', '118', '152']
['raccoon-130.jpg', '640', '426', 'raccoon', '223', '62', '497', '307']
['raccoon-130.jpg', '640', '426', 'raccoon', '453', '41', '640', '423']
['raccoon-7.jpg', '410', '308', 'raccoon', '92', '79', '271', '264']
['raccoon-107.jpg', '500', '622', 'raccoon', '165', '51', '496', '590']
['raccoon-173.jpg', '550', '388', 'raccoon', '202', '21', '515', '387']
['raccoon-165.jpg', '199', '253', 'raccoon', '27', '11', '194', '228']
['raccoon-25.jpg', '634', '641', 'raccoon', '31', '82', '325', '641']
['raccoon-67.jpg', '272', '185', 'raccoon', '18', '17', '224', '168']
['raccoon-13.jpg', '660', '495', 'raccoon', '55', '28', '393', '313']
['raccoon-191.jpg', '634', '445', 'raccoon', '100', '89', '478', '331']
['raccoon-45.jpg', '620', '372', 'raccoon', '140', '6', '454', '370']
['raccoon-31.jpg', '236', '214', 'raccoon', '82', '21', '187', '197']
['raccoon-31.jpg', '236', '214', 'raccoon', '11', '55', '80', '145']
['raccoon-172.jpg', '615', '346', 'raccoon', '183', '53', '399', '302']
['raccoon-100.jpg', '960', '576', 'raccoon', '548', '10', '954', '520']
['raccoon-148.jpg', '500', '375', 'raccoon', '32', '177', '174', '316']
['raccoon-148.jpg', '500', '375', 'raccoon', '309', '172', '428', '315']
['raccoon-23.jpg', '259', '194', 'raccoon', '108', '1', '258', '194']
['raccoon-30.jpg', '266', '190', 'raccoon', '78', '25', '182', '177']
['raccoon-33.jpg', '602', '843', 'raccoon', '89', '12', '593', '843']
['raccoon-136.jpg', '256', '197', 'raccoon', '51', '24', '198', '192']
['raccoon-114.jpg', '625', '418', 'raccoon', '242', '35', '523', '264']
['raccoon-76.jpg', '225', '225', 'raccoon', '14', '1', '212', '132']
['raccoon-104.jpg', '600', '304', 'raccoon', '189', '41', '340', '249']
['raccoon-109.jpg', '192', '259', 'raccoon', '9', '1', '177', '252']
['raccoon-87.jpg', '256', '197', 'raccoon', '1', '3', '206', '191']
['raccoon-113.jpg', '640', '480', 'raccoon', '1', '1', '384', '436']
['raccoon-78.jpg', '223', '226', 'raccoon', '28', '15', '221', '216']
['raccoon-65.jpg', '480', '360', 'raccoon', '123', '27', '338', '284']
['raccoon-122.jpg', '178', '283', 'raccoon', '7', '7', '174', '198']
['raccoon-73.jpg', '284', '177', 'raccoon', '56', '16', '274', '166']
['raccoon-137.jpg', '320', '240', 'raccoon', '71', '8', '304', '233']
['raccoon-171.jpg', '224', '225', 'raccoon', '108', '21', '180', '115']
['raccoon-190.jpg', '259', '194', 'raccoon', '78', '54', '153', '135']
```

Print the shape of the train dataset

```
train_df = pd.read_csv('train_labels.csv')
train_df.head()
```

	filename	width	height	class	xmin	ymin	xmax	ymax
0	raccoon-17.jpg	259	194	raccoon	95	60	167	118
1	raccoon-11.jpg	660	432	raccoon	3	1	461	431
2	raccoon-63.jpg	600	400	raccoon	74	107	280	290
3	raccoon-63.jpg	600	400	raccoon	227	93	403	298
4	raccoon-60.jpg	273	185	raccoon	58	33	197	127

```
train_df.shape
```

```
(173, 8)
```

Declare a variable IMAGE_SIZE = 128 as we will be using MobileNet which will

```
IMAGE_SIZE = 128
```

With the help of csv.reader write a for loop which can load the train.csv file and x0,y0,x1,y1 in induvidual variables.

1. Create a list variable known as 'path' which has all the path for all the training images
2. Create an array 'coords' which has the resized coordinates of the bounding box for the training images

Note: All the training images should be downsampled to 128 * 128 as it is the input shape of Mob detection). Hence the corresponding coordinates of the bounding boxes should be changed to ma

```
# Install the tool for unrar the rar file.
!pip install patool
```

```
# First get the images from archive data

import patoolib
patoolib.extract_archive("images_racoon.rar", outdir="/content/drive/My Drive/Colab Noteboc
```

```
train_df.head()
```

	filename	width	height	class	xmin	ymin	xmax	ymax
0	raccoon-17.jpg	259	194	raccoon	95	60	167	118
1	raccoon-11.jpg	660	432	raccoon	3	1	461	431
2	raccoon-63.jpg	600	400	raccoon	74	107	280	290
3	raccoon-63.jpg	600	400	raccoon	227	93	403	298
4	raccoon-60.jpg	273	185	raccoon	58	33	197	127

```
# Loading the csv file and getting the individual variables values
with open('train_labels.csv', 'r') as csvfile:
    paths = []
    coords = np.zeros((sum(1 for line in csvfile)-1, 4))
    reader = csv.reader(csvfile, delimiter=',')
    csvfile.seek(0)
    next(reader)
    for col, row in enumerate(reader):

        path,  image_width,image_height,_, x0, y0, x1, y1 = row
        #print(path)
        #print(row)
        path = "/content/drive/My Drive/Colab Notebooks/Labs/images/"+path
        coords[col, 0] = int(x0) * IMAGE_SIZE / int(image_width) # Normalize bounding box
        coords[col, 1] = int(y0) * IMAGE_SIZE / int(image_height) # Normalize bounding box
        coords[col, 2] = (int(x1)- int(x0)) * IMAGE_SIZE / int(image_width) # Normalize bo
        coords[col, 3] = (int(y1) - int(y0)) * IMAGE_SIZE / int(image_height)
        paths.append(path)


# Just for checking
print(coords.shape)
print(paths)
print(image_height)
```

```
(173, 4)
['/content/drive/My Drive/Colab Notebooks/Labs/images/raccoon-17.jpg', '/content/driv
183
```

Write a for loop which can load all the training images into a variable 'batch_im 'paths' variable

Note: Convert the image to RGB scale as the MobileNet accepts 3 channels as inputs

```
batch_images = np.zeros((len(paths), IMAGE_SIZE, IMAGE_SIZE, 3), dtype=np.float32)
for i, f in enumerate(paths):
    img = Image.open(f) # Read image
    img = img.resize((IMAGE_SIZE, IMAGE_SIZE)) # Resize image
    img = img.convert('RGB')
    batch_images[i] = preprocess_input(np.array(img, dtype=np.float32))
```

Import MobileNet and load MobileNet into a variable named 'model' which tak Freeze all the layers. Add convolution and reshape layers at the end to ensure

```
ALPHA = 1.0 # Width hyper parameter for MobileNet (0.25, 0.5, 0.75, 1.0). Higher width mea

EPOCHS = 10 # Number of epochs. I got decent performance with just 5.
BATCH_SIZE = 32 # Depends on your GPU or CPU RAM.

model = MobileNet(input_shape=(IMAGE_SIZE, IMAGE_SIZE, 3), include_top=False, alpha=ALPHA)
```

```
# Do not include classification (top) layer

# to freeze layers, except the new top layer, of course, which will be added below
for layer in model.layers:
    layer.trainable = False

# Add new top layer which is a conv layer of the same size as the previous layer so that c
x = model.layers[-1].output
x = Conv2D(4, kernel_size=4, name="coords")(x)
# In the line above kernel size should be 3 for img size 96, 4 for img size 128, 5 for img
x = Reshape((4,))(x) # These are the 4 predicted coordinates of one BBox

model = Model(inputs=model.input, outputs=x)
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorfl

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorfl

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorfl

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorfl

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorfl

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorfl

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorfl

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorfl

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorfl

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorfl

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorfl

Downloading data from https://github.com/fchollet/deep-learning-models/releases/downl
17227776/17225924 [==============================] - 1s 0us/step
```

```
model.summary()
```

```
Model: "model_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 128, 128, 3)       0
_________________________________________________________________
conv1_pad (ZeroPadding2D)    (None, 129, 129, 3)       0
_________________________________________________________________
conv1 (Conv2D)               (None, 64, 64, 32)        864
_________________________________________________________________
conv1_bn (BatchNormalization (None, 64, 64, 32)        128
_________________________________________________________________
conv1_relu (ReLU)            (None, 64, 64, 32)        0
_________________________________________________________________
conv_dw_1 (DepthwiseConv2D)  (None, 64, 64, 32)        288
_________________________________________________________________
conv_dw_1_bn (BatchNormaliza (None, 64, 64, 32)        128
_________________________________________________________________
conv_dw_1_relu (ReLU)        (None, 64, 64, 32)        0
_________________________________________________________________
conv_pw_1 (Conv2D)           (None, 64, 64, 64)        2048
_________________________________________________________________
conv_pw_1_bn (BatchNormaliza (None, 64, 64, 64)        256
_________________________________________________________________
conv_pw_1_relu (ReLU)        (None, 64, 64, 64)        0
_________________________________________________________________
conv_pad_2 (ZeroPadding2D)   (None, 65, 65, 64)        0
_________________________________________________________________
conv_dw_2 (DepthwiseConv2D)  (None, 32, 32, 64)        576
_________________________________________________________________
conv_dw_2_bn (BatchNormaliza (None, 32, 32, 64)        256
_________________________________________________________________
conv_dw_2_relu (ReLU)        (None, 32, 32, 64)        0
_________________________________________________________________
conv_pw_2 (Conv2D)           (None, 32, 32, 128)       8192
_________________________________________________________________
conv_pw_2_bn (BatchNormaliza (None, 32, 32, 128)       512
_________________________________________________________________
conv_pw_2_relu (ReLU)        (None, 32, 32, 128)       0
_________________________________________________________________
conv_dw_3 (DepthwiseConv2D)  (None, 32, 32, 128)       1152
_________________________________________________________________
conv_dw_3_bn (BatchNormaliza (None, 32, 32, 128)       512
_________________________________________________________________
conv_dw_3_relu (ReLU)        (None, 32, 32, 128)       0
_________________________________________________________________
conv_pw_3 (Conv2D)           (None, 32, 32, 128)       16384
_________________________________________________________________
conv_pw_3_bn (BatchNormaliza (None, 32, 32, 128)       512
_________________________________________________________________
conv_pw_3_relu (ReLU)        (None, 32, 32, 128)       0
_________________________________________________________________
conv_pad_4 (ZeroPadding2D)   (None, 33, 33, 128)       0
_________________________________________________________________
conv_dw_4 (DepthwiseConv2D)  (None, 16, 16, 128)       1152
_________________________________________________________________
conv_dw_4_bn (BatchNormaliza (None, 16, 16, 128)       512
_________________________________________________________________
conv_dw_4_relu (ReLU)        (None, 16, 16, 128)       0
_________________________________________________________________
conv_pw_4 (Conv2D)           (None, 16, 16, 256)       32768
```

conv_pw_4_bn (BatchNormaliza	(None, 16, 16, 256)	1024
conv_pw_4_relu (ReLU)	(None, 16, 16, 256)	0
conv_dw_5 (DepthwiseConv2D)	(None, 16, 16, 256)	2304
conv_dw_5_bn (BatchNormaliza	(None, 16, 16, 256)	1024
conv_dw_5_relu (ReLU)	(None, 16, 16, 256)	0
conv_pw_5 (Conv2D)	(None, 16, 16, 256)	65536
conv_pw_5_bn (BatchNormaliza	(None, 16, 16, 256)	1024
conv_pw_5_relu (ReLU)	(None, 16, 16, 256)	0
conv_pad_6 (ZeroPadding2D)	(None, 17, 17, 256)	0
conv_dw_6 (DepthwiseConv2D)	(None, 8, 8, 256)	2304
conv_dw_6_bn (BatchNormaliza	(None, 8, 8, 256)	1024
conv_dw_6_relu (ReLU)	(None, 8, 8, 256)	0
conv_pw_6 (Conv2D)	(None, 8, 8, 512)	131072
conv_pw_6_bn (BatchNormaliza	(None, 8, 8, 512)	2048
conv_pw_6_relu (ReLU)	(None, 8, 8, 512)	0
conv_dw_7 (DepthwiseConv2D)	(None, 8, 8, 512)	4608
conv_dw_7_bn (BatchNormaliza	(None, 8, 8, 512)	2048
conv_dw_7_relu (ReLU)	(None, 8, 8, 512)	0
conv_pw_7 (Conv2D)	(None, 8, 8, 512)	262144
conv_pw_7_bn (BatchNormaliza	(None, 8, 8, 512)	2048
conv_pw_7_relu (ReLU)	(None, 8, 8, 512)	0
conv_dw_8 (DepthwiseConv2D)	(None, 8, 8, 512)	4608
conv_dw_8_bn (BatchNormaliza	(None, 8, 8, 512)	2048
conv_dw_8_relu (ReLU)	(None, 8, 8, 512)	0
conv_pw_8 (Conv2D)	(None, 8, 8, 512)	262144
conv_pw_8_bn (BatchNormaliza	(None, 8, 8, 512)	2048
conv_pw_8_relu (ReLU)	(None, 8, 8, 512)	0
conv_dw_9 (DepthwiseConv2D)	(None, 8, 8, 512)	4608
conv_dw_9_bn (BatchNormaliza	(None, 8, 8, 512)	2048
conv_dw_9_relu (ReLU)	(None, 8, 8, 512)	0
conv_pw_9 (Conv2D)	(None, 8, 8, 512)	262144

```
conv_pw_9 (Conv2D)           (None, 8, 8, 512)         262144
_________________________________________________________________
conv_pw_9_bn (BatchNormaliza (None, 8, 8, 512)         2048
_________________________________________________________________
conv_pw_9_relu (ReLU)        (None, 8, 8, 512)         0
_________________________________________________________________
conv_dw_10 (DepthwiseConv2D) (None, 8, 8, 512)         4608
_________________________________________________________________
conv_dw_10_bn (BatchNormaliz (None, 8, 8, 512)         2048
_________________________________________________________________
conv_dw_10_relu (ReLU)       (None, 8, 8, 512)         0
_________________________________________________________________
conv_pw_10 (Conv2D)          (None, 8, 8, 512)         262144
_________________________________________________________________
conv_pw_10_bn (BatchNormaliz (None, 8, 8, 512)         2048
_________________________________________________________________
conv_pw_10_relu (ReLU)       (None, 8, 8, 512)         0
_________________________________________________________________
conv_dw_11 (DepthwiseConv2D) (None, 8, 8, 512)         4608
_________________________________________________________________
conv_dw_11_bn (BatchNormaliz (None, 8, 8, 512)         2048
_________________________________________________________________
conv_dw_11_relu (ReLU)       (None, 8, 8, 512)         0
_________________________________________________________________
conv_pw_11 (Conv2D)          (None, 8, 8, 512)         262144
_________________________________________________________________
conv_pw_11_bn (BatchNormaliz (None, 8, 8, 512)         2048
_________________________________________________________________
conv_pw_11_relu (ReLU)       (None, 8, 8, 512)         0
_________________________________________________________________
conv_pad_12 (ZeroPadding2D)  (None, 9, 9, 512)         0
_________________________________________________________________
conv_dw_12 (DepthwiseConv2D) (None, 4, 4, 512)         4608
_________________________________________________________________
conv_dw_12_bn (BatchNormaliz (None, 4, 4, 512)         2048
_________________________________________________________________
conv_dw_12_relu (ReLU)       (None, 4, 4, 512)         0
_________________________________________________________________
conv_pw_12 (Conv2D)          (None, 4, 4, 1024)        524288
_________________________________________________________________
conv_pw_12_bn (BatchNormaliz (None, 4, 4, 1024)        4096
_________________________________________________________________
conv_pw_12_relu (ReLU)       (None, 4, 4, 1024)        0
_________________________________________________________________
conv_dw_13 (DepthwiseConv2D) (None, 4, 4, 1024)        9216
_________________________________________________________________
conv_dw_13_bn (BatchNormaliz (None, 4, 4, 1024)        4096
_________________________________________________________________
conv_dw_13_relu (ReLU)       (None, 4, 4, 1024)        0
_________________________________________________________________
conv_pw_13 (Conv2D)          (None, 4, 4, 1024)        1048576
_________________________________________________________________
conv_pw_13_bn (BatchNormaliz (None, 4, 4, 1024)        4096
_________________________________________________________________
conv_pw_13_relu (ReLU)       (None, 4, 4, 1024)        0
_________________________________________________________________
coords (Conv2D)              (None, 1, 1, 4)           65540
_________________________________________________________________
reshape_1 (Reshape)          (None, 4)                 0
=================================================================
Total params: 3,294,404
Trainable params: 65,540
```

Non-trainable params: 2,228,864

Define a custom loss function IoU which calculates Intersection Over Union

```
gt = coords
def loss(gt,pred):
    intersections = 0
    unions = 0
    diff_width = np.minimum(gt[:,0] + gt[:,2], pred[:,0] + pred[:,2]) - np.maximum(gt[:,0]
    diff_height = np.minimum(gt[:,1] + gt[:,3], pred[:,1] + pred[:,3]) - np.maximum(gt[:,1
    intersection = diff_width * diff_height

    # Compute union
    area_gt = gt[:,2] * gt[:,3]
    area_pred = pred[:,2] * pred[:,3]
    union = area_gt + area_pred - intersection

#     Compute intersection and union over multiple boxes
    for j, _ in enumerate(union):
        if union[j] > 0 and intersection[j] > 0 and union[j] >= intersection[j]:
            intersections += intersection[j]
            unions += union[j]

    # Compute IOU. Use epsilon to prevent division by zero
    iou = np.round(intersections / (unions + epsilon()), 4)
    iou = iou.astype(np.float32)
    return iou

def IoU(y_true, y_pred):
    iou = tf.py_func(loss, [y_true, y_pred], tf.float32)
    return iou
```

Write model.compile function & model.fit function with:

1. Optimizer = Adam, Loss = 'mse' and metrics = IoU
2. Epochs = 30, batch_size = 32, verbose = 1

```
model.compile(optimizer='Adam', loss='mse', metrics=[IoU]) # Regression loss is MSE

#checkpoint = ModelCheckpoint("model-{val_iou:.2f}.h5", verbose=1, save_best_only=True,
#                             save_weights_only=True, mode="max", period=1) # Checkpoint
#stop = EarlyStopping(monitor="val_iou", patience=PATIENCE, mode="max") # Stop early, if t
#reduce_lr = ReduceLROnPlateau(monitor="val_iou", factor=0.2, patience=10, min_lr=1e-7, ve
# Reduce learning rate if Validation IOU does not improve

model.fit(batch_images,gt,
            epochs=30,batch_size = 32,
            verbose=1)
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorfl

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorfl

Epoch 1/30
173/173 [==============================] - 8s 46ms/step - loss: 2783.7782 - IoU: 0.09
Epoch 2/30
173/173 [==============================] - 0s 517us/step - loss: 667.0259 - IoU: 0.43
Epoch 3/30
173/173 [==============================] - 0s 496us/step - loss: 644.1832 - IoU: 0.53
Epoch 4/30
173/173 [==============================] - 0s 498us/step - loss: 546.2977 - IoU: 0.53
Epoch 5/30
173/173 [==============================] - 0s 524us/step - loss: 348.8700 - IoU: 0.59
Epoch 6/30
173/173 [==============================] - 0s 497us/step - loss: 270.3356 - IoU: 0.57
Epoch 7/30
173/173 [==============================] - 0s 503us/step - loss: 247.8495 - IoU: 0.61
Epoch 8/30
173/173 [==============================] - 0s 502us/step - loss: 194.6591 - IoU: 0.65
Epoch 9/30
173/173 [==============================] - 0s 491us/step - loss: 167.8069 - IoU: 0.69
Epoch 10/30
173/173 [==============================] - 0s 504us/step - loss: 146.8834 - IoU: 0.71
Epoch 11/30
173/173 [==============================] - 0s 506us/step - loss: 129.2607 - IoU: 0.71
Epoch 12/30
173/173 [==============================] - 0s 518us/step - loss: 115.5648 - IoU: 0.74
Epoch 13/30
173/173 [==============================] - 0s 524us/step - loss: 108.0287 - IoU: 0.74
Epoch 14/30
173/173 [==============================] - 0s 504us/step - loss: 95.9128 - IoU: 0.770
Epoch 15/30
173/173 [==============================] - 0s 507us/step - loss: 90.4219 - IoU: 0.771
Epoch 16/30
173/173 [==============================] - 0s 510us/step - loss: 87.8742 - IoU: 0.784
Epoch 17/30
173/173 [==============================] - 0s 501us/step - loss: 83.8819 - IoU: 0.783
Epoch 18/30
173/173 [==============================] - 0s 489us/step - loss: 76.1848 - IoU: 0.790
Epoch 19/30
173/173 [==============================] - 0s 495us/step - loss: 78.4024 - IoU: 0.790
Epoch 20/30
173/173 [==============================] - 0s 500us/step - loss: 72.7697 - IoU: 0.797
Epoch 21/30
173/173 [==============================] - 0s 484us/step - loss: 68.9950 - IoU: 0.816
Epoch 22/30
173/173 [==============================] - 0s 480us/step - loss: 64.9422 - IoU: 0.809
Epoch 23/30
173/173 [==============================] - 0s 505us/step - loss: 64.8965 - IoU: 0.815
Epoch 24/30
173/173 [==============================] - 0s 470us/step - loss: 62.5142 - IoU: 0.821
Epoch 25/30
173/173 [==============================] - 0s 479us/step - loss: 62.7122 - IoU: 0.819
Epoch 26/30
173/173 [==============================] - 0s 479us/step - loss: 60.0789 - IoU: 0.831
Epoch 27/30
173/173 [==============================] - 0s 503us/step - loss: 55.4476 - IoU: 0.837
Epoch 28/30
173/173 [==============================] - 0s 512us/step - loss: 61.0951 - IoU: 0.835
Epoch 29/30
```

```
173/173 [==============================] - 0s 499us/step - loss: 60.6017 - IoU: 0.832
Epoch 30/30
173/173 [==============================] - 0s 491us/step - loss: 58.3179 - IoU: 0.837
<keras.callbacks.History at 0x7f2d89a8d940>
```

Pick a test image from the given data

```
# Pick a test image, run model, show image, and show predicted bounding box overlaid on th
import cv2
filename = '/content/drive/My Drive/Colab Notebooks/Labs/images/raccoon-92.jpg'
unscaled = cv2.imread(filename) # Original image for display
```

Resize the image to 128 * 128 and preprocess the image for the MobileNet mo

```
image_height, image_width, _ = unscaled.shape
image = cv2.resize(unscaled, (IMAGE_SIZE, IMAGE_SIZE)) # Rescaled image to run the network
feat_scaled = preprocess_input(np.array(image, dtype=np.float32))
```

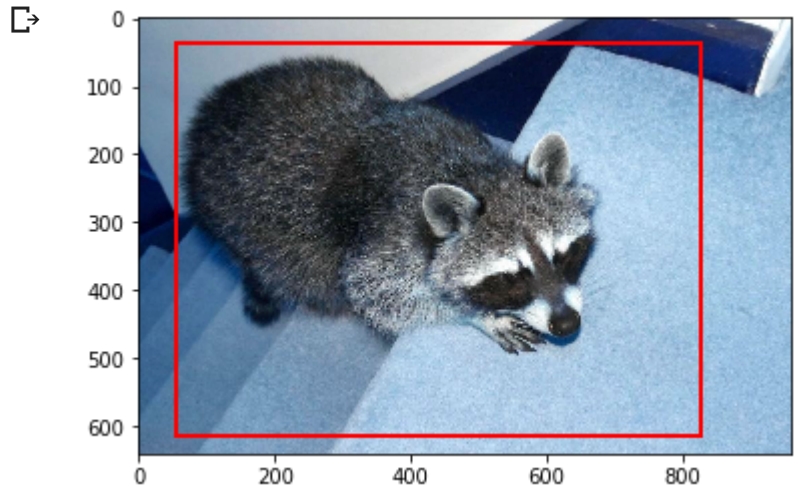
Predict the coordinates of the bounding box for the given test image

```
region = model.predict(x=np.array([feat_scaled]))[0] # Predict the BBox
```

Plot the test image using .imshow and draw a boundary box around the image the model

```
x0 = int(region[0] * image_width / IMAGE_SIZE) # Scale the BBox
y0 = int(region[1] * image_height / IMAGE_SIZE)

x1 = int((region[2]) * image_width / IMAGE_SIZE)
y1 = int((region[3]) * image_height / IMAGE_SIZE)


import matplotlib.pyplot as plt
import matplotlib.patches as patches
from PIL import Image
import numpy as np


# Create figure and axes
fig,ax = plt.subplots(1)

# Display the image
ax.imshow(unscaled)

# Create a Rectangle patch
rect = patches.Rectangle((x0, y0), (x1 - x0) , (y1 - y0) , linewidth=2, edgecolor='r', fac
```

```
# Add the patch to the Axes
ax.add_patch(rect)

plt.show()
```



Time Series Prediction using LSTM

Download Data

Link: https://datamarket.com/data/set/2324/daily-minimum-temperatures-in-melbourne-australia

Description

Daily minimum temperatures in Melbourne, Australia, 1981-1990

Units: Degrees Celcius

Steps before loading

- Rename the column name with temprature values to "Temprature"
- In the last, there is one extra row in the data, remove it by opening the file and save it again.
- There are some values in Temprature column which have a "?" before them, they will give err
- If you don't want to do these steps, just load the data file given by Great Learning.

Mount google drive

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.m
```

Change your present working directory

```
os.chdir('/content/drive/My Drive/Colab Notebooks/Labs')
```

Load your data file

```
df = pd.read_csv('daily-minimum-temperatures-in-me.csv')
df.sort_index(inplace=True)
df.head()
```

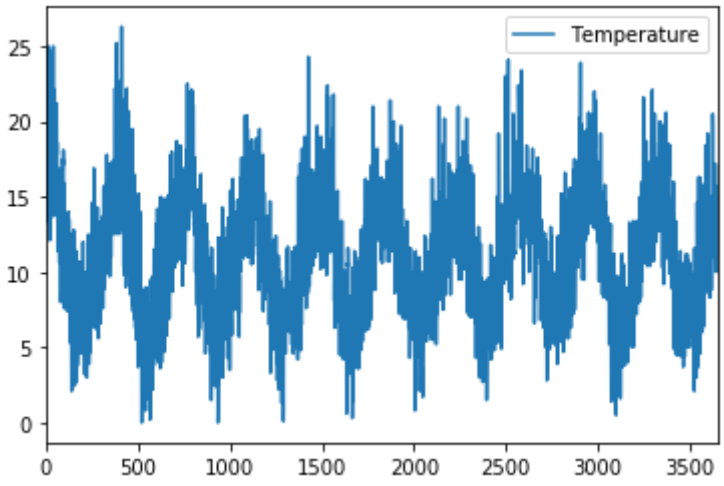
	Date	Temperature
0	1981-01-01	20.7
1	1981-01-02	17.9
2	1981-01-03	18.8
3	1981-01-04	14.6
4	1981-01-05	15.8

Plot data

```
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from keras.utils import np_utils
```

```
df.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f2b65218ac8>
```



Descibe your dataframe

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3650 entries, 0 to 3649
Data columns (total 2 columns):
Date           3650 non-null object
Temperature    3650 non-null float64
dtypes: float64(1), object(1)
memory usage: 57.2+ KB
```

```
df.head()
# Dataframe contains only 2 columns, date and Temperature
```

	Date	Temperature
0	1981-01-01	20.7
1	1981-01-02	17.9
2	1981-01-03	18.8
3	1981-01-04	14.6
4	1981-01-05	15.8

Check for null values

```
#Check for null values
df.isnull().sum()
```

```
Date           0
Temperature    0
dtype: int64
```

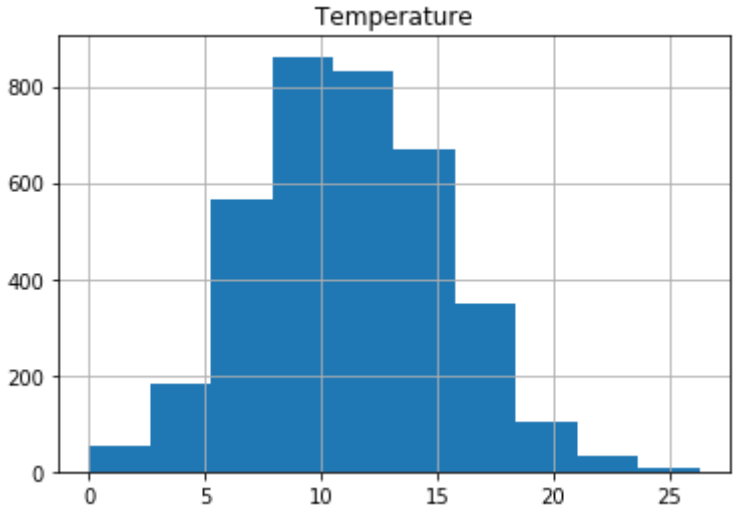
Drop null values

```
# No null values in the dataframe
```

Get the representation of the distribution of data in the form of histogram

```
# Display the data using historgram.
df.hist()
```

```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7f2b65163400>]],
      dtype=object)
```




```
# There are very few values containing temperature 20 to 25.
```

Check the maximum and minimum values

```
#Check Data Range and Maximum and minimum values
print('Min', np.min(df))
print('Max', np.max(df))
```

```
Min Date           1981-01-01
Temperature             0
dtype: object
Max Date           1990-12-31
Temperature          26.3
dtype: object
```

Normalize the data

```
df.drop("Date", axis=1, inplace=True)
```

```
# Normalize the data using min max scaler
#Normalize the data
scaler = MinMaxScaler(feature_range=(0, 1))
scaled = scaler.fit_transform(df)
```

Check the maximum and minimum values of scaled data

```
print('Min', np.min(scaled))
print('Max', np.max(scaled))
```

```
Min 0.0
Max 1.0
```

Look into some of the scaled values

```
scaled[0:5]
```

```
array([[0.78707224],
       [0.68060837],
       [0.7148289 ],
       [0.55513308],
       [0.60076046]])
```

Split data into Training and Testing

```
#70% examples will used for training (in the begining)
train_size = int(len(scaled) * 0.70)

#30% will be used for Test
test_size = len(scaled - train_size)
```

Print train and test size

```
#Split the data
train, test = scaled[0:train_size, :], scaled[train_size: len(scaled), :]
print('train: {}\ntest: {}'.format(len(train), len(test)))
```

```
train: 2555
test: 1095
```

Create the sequential data

Map the temprature at a particular time t to the temprature at time t+n, where n is any number you

For example: to map tempratures of consecutive days, use t+1, i.e. loop_back = 1

Define your function to create dataset

```
#window - how long the sequence will be
def create_dataset(dataset, window=1):

    dataX, dataY = [], []

    for i in range(len(dataset)-window):

        a = dataset[i:(i+window), 0]
        dataX.append(a)
        dataY.append(dataset[i + window, 0])

    return np.array(dataX), np.array(dataY)
```

Use function to get training and test set

```
#Create Input and Output
window_size = 1
X_train, y_train = create_dataset(train, window_size)
X_test, y_test = create_dataset(test, window_size)
```

Transform the prepared train and test input data into the expected structure using num

```
X_train.shape
```

```
(2554, 1)
```

```
y_train.shape

(X_train.shape[0], X_train.shape[1], 1)
```

```
(2554, 1, 1)
```

```
#Make it 3 Dimensional Data - needed for LSTM
X_train = np.reshape(X_train, (X_train.shape[0], X_train.shape[1], 1))
X_test = np.reshape(X_test, (X_test.shape[0], X_test.shape[1], 1))
print(X_train.shape)
print(X_test.shape)
```

```
(2554, 1, 1)
(1094, 1, 1)
```

Define Model

Define sequntial model, add LSTM layer and compile the model

```
tf.keras.backend.clear_session()
model = tf.keras.Sequential()
model.add(tf.keras.layers.LSTM(32, input_shape=(window_size, 1)))
model.add(tf.keras.layers.Dense(1))
model.compile(optimizer='adam', loss='mse',metrics=['mse'])
```

Summarize your model

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
lstm (LSTM)                  (None, 32)                4352      
_________________________________________________________________
dense (Dense)                (None, 1)                 33        
=================================================================
Total params: 4,385
Trainable params: 4,385
Non-trainable params: 0
_________________________________________________________________
```

Train the model

```
model.fit(X_train, y_train, epochs=100, validation_data=(X_test, y_test), batch_size=32)
```

```
==========================] - 1s 334us/sample - loss: 0.1019 - mean_squared_error: 0.101
==========================] - 0s 155us/sample - loss: 0.0173 - mean_squared_error: 0.017
==========================] - 0s 161us/sample - loss: 0.0138 - mean_squared_error: 0.013
==========================] - 0s 162us/sample - loss: 0.0131 - mean_squared_error: 0.013
==========================] - 0s 169us/sample - loss: 0.0124 - mean_squared_error: 0.012
==========================] - 0s 192us/sample - loss: 0.0117 - mean_squared_error: 0.011
==========================] - 0s 172us/sample - loss: 0.0112 - mean_squared_error: 0.011
==========================] - 0s 167us/sample - loss: 0.0108 - mean_squared_error: 0.010
==========================] - 0s 164us/sample - loss: 0.0105 - mean_squared_error: 0.010
==========================] - 0s 164us/sample - loss: 0.0103 - mean_squared_error: 0.010
==========================] - 0s 178us/sample - loss: 0.0102 - mean_squared_error: 0.010
==========================] - 0s 166us/sample - loss: 0.0101 - mean_squared_error: 0.010
==========================] - 0s 168us/sample - loss: 0.0101 - mean_squared_error: 0.010
==========================] - 0s 152us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 169us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 177us/sample - loss: 0.0101 - mean_squared_error: 0.010
==========================] - 0s 166us/sample - loss: 0.0101 - mean_squared_error: 0.010
==========================] - 0s 174us/sample - loss: 0.0101 - mean_squared_error: 0.010
==========================] - 0s 168us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 188us/sample - loss: 0.0101 - mean_squared_error: 0.010
==========================] - 0s 170us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 173us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 175us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 176us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 182us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 166us/sample - loss: 0.0101 - mean_squared_error: 0.010
==========================] - 0s 179us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 172us/sample - loss: 0.0101 - mean_squared_error: 0.010
==========================] - 0s 175us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 169us/sample - loss: 0.0100 - mean_squared_error: 0.010
```

```
==========] - 0s 182us/sample - loss: 0.0101 - mean_squared_error: 0.010
==========] - 0s 181us/sample - loss: 0.0101 - mean_squared_error: 0.010
==========] - 0s 191us/sample - loss: 0.0101 - mean_squared_error: 0.010
==========] - 0s 193us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 176us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 167us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 179us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 177us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 172us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 189us/sample - loss: 0.0101 - mean_squared_error: 0.010
==========] - 0s 173us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 180us/sample - loss: 0.0101 - mean_squared_error: 0.010
==========] - 0s 163us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 170us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 171us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 172us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 183us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 180us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 175us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 175us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 188us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 176us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 179us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 186us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 166us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 167us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 169us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 178us/sample - loss: 0.0101 - mean_squared_error: 0.010
==========] - 0s 155us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 166us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========] - 0s 170us/sample - loss: 0.0100 - mean_squared_error: 0.010
```

```
==========================] - 0s 172us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 170us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 173us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 170us/sample - loss: 0.0101 - mean_squared_error: 0.010
==========================] - 0s 160us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 166us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 168us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 162us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 165us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 161us/sample - loss: 0.0101 - mean_squared_error: 0.010
==========================] - 0s 160us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 161us/sample - loss: 0.0101 - mean_squared_error: 0.010
==========================] - 0s 176us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 176us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 169us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 162us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 163us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 159us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 165us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 157us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 181us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 161us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 165us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 170us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 1s 204us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 1s 211us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 177us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 165us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 151us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 159us/sample - loss: 0.0100 - mean_squared_error: 0.010
==========================] - 0s 161us/sample - loss: 0.0100 - mean_squared_error: 0.010
```

```
=========================] - 0s 162us/sample - loss: 0.0100 - mean_squared_error: 0.010
=========================] - 0s 161us/sample - loss: 0.0100 - mean_squared_error: 0.010
=========================] - 0s 159us/sample - loss: 0.0100 - mean_squared_error: 0.010
=========================] - 0s 166us/sample - loss: 0.0100 - mean_squared_error: 0.010
=========================] - 0s 165us/sample - loss: 0.0100 - mean_squared_error: 0.010
=========================] - 0s 174us/sample - loss: 0.0100 - mean_squared_error: 0.010
=========================] - 0s 164us/sample - loss: 0.0100 - mean_squared_error: 0.010
=========================] - 0s 162us/sample - loss: 0.0100 - mean_squared_error: 0.010
.keras.callbacks.History at 0x7f2b5dc3c780>
```

Make Predictions and Evaluate your model

```
loss_and_metrics = model.evaluate(X_train, y_train)
print(loss_and_metrics)
loss_and_metrics = model.evaluate(X_test, y_test)
print(loss_and_metrics)
```

```
2554/2554 [==============================] - 0s 75us/sample - loss: 0.0099 - mean_squ
[0.009941597668251305, 0.0099415975]
1094/1094 [==============================] - 0s 72us/sample - loss: 0.0087 - mean_squ
[0.008675231253115335, 0.008675232]
```

```
#Get prediction for both Training and Test Data
trainPredict = model.predict(X_train)
testPredict = model.predict(X_test)
```

Plot the results

```
#Un-normalize the predited data
trainPredict = model.predict(X_train)
testPredict = model.predict(X_test)
trainPredict = scaler.inverse_transform(trainPredict)
testPredict = scaler.inverse_transform(testPredict)
```

```
trainPredictPlot = np.empty_like(scaled)
trainPredictPlot[:, :] = np.nan
trainPredictPlot[window_size:len(trainPredict)+window_size, :] = trainPredict
# shift test predictions for plotting
testPredictPlot = np.empty_like(scaled)
testPredictPlot[:, :] = np.nan
testPredictPlot[len(trainPredict)+(window_size*2):len(scaled), :] = testPredict
# plot baseline and predictions
plt.figure(figsize=(10,6))
```