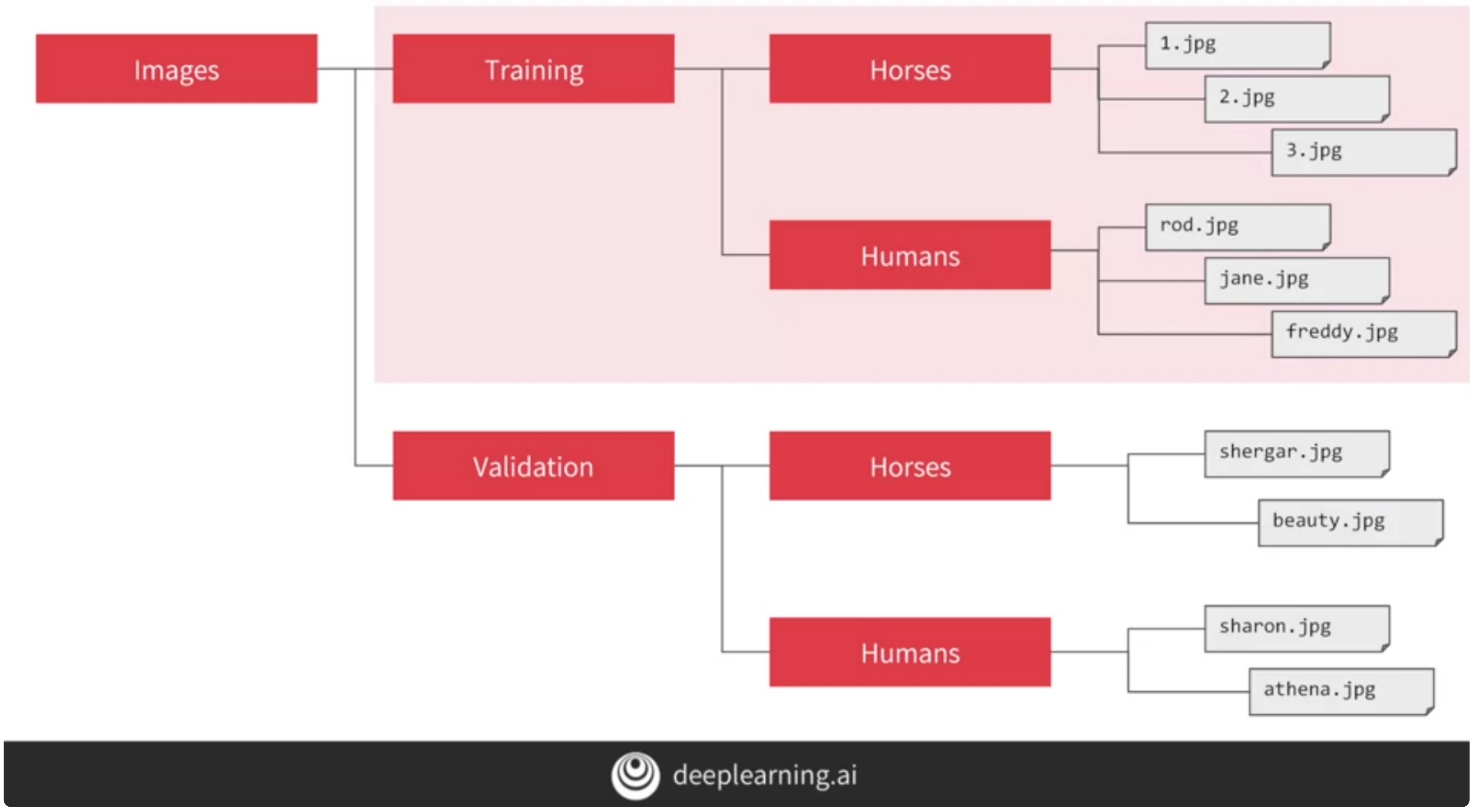
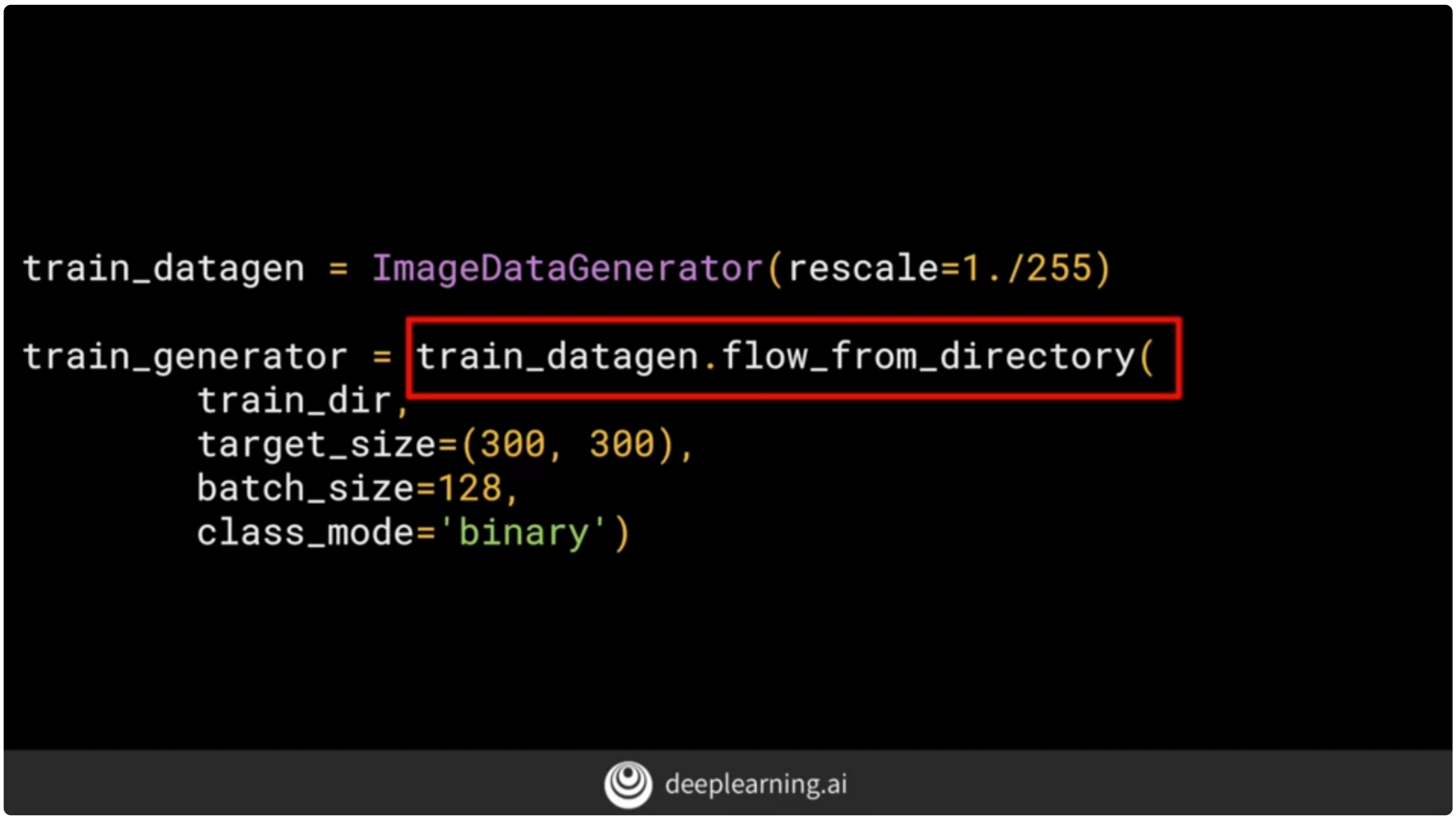
ImageDataGenerator

[ImageDataGenerator]



on pointing to a directory containing diffferent folders/sub-directories for different label's images, the function automatically reads and auto labels these images based on their location in different sub-directories.



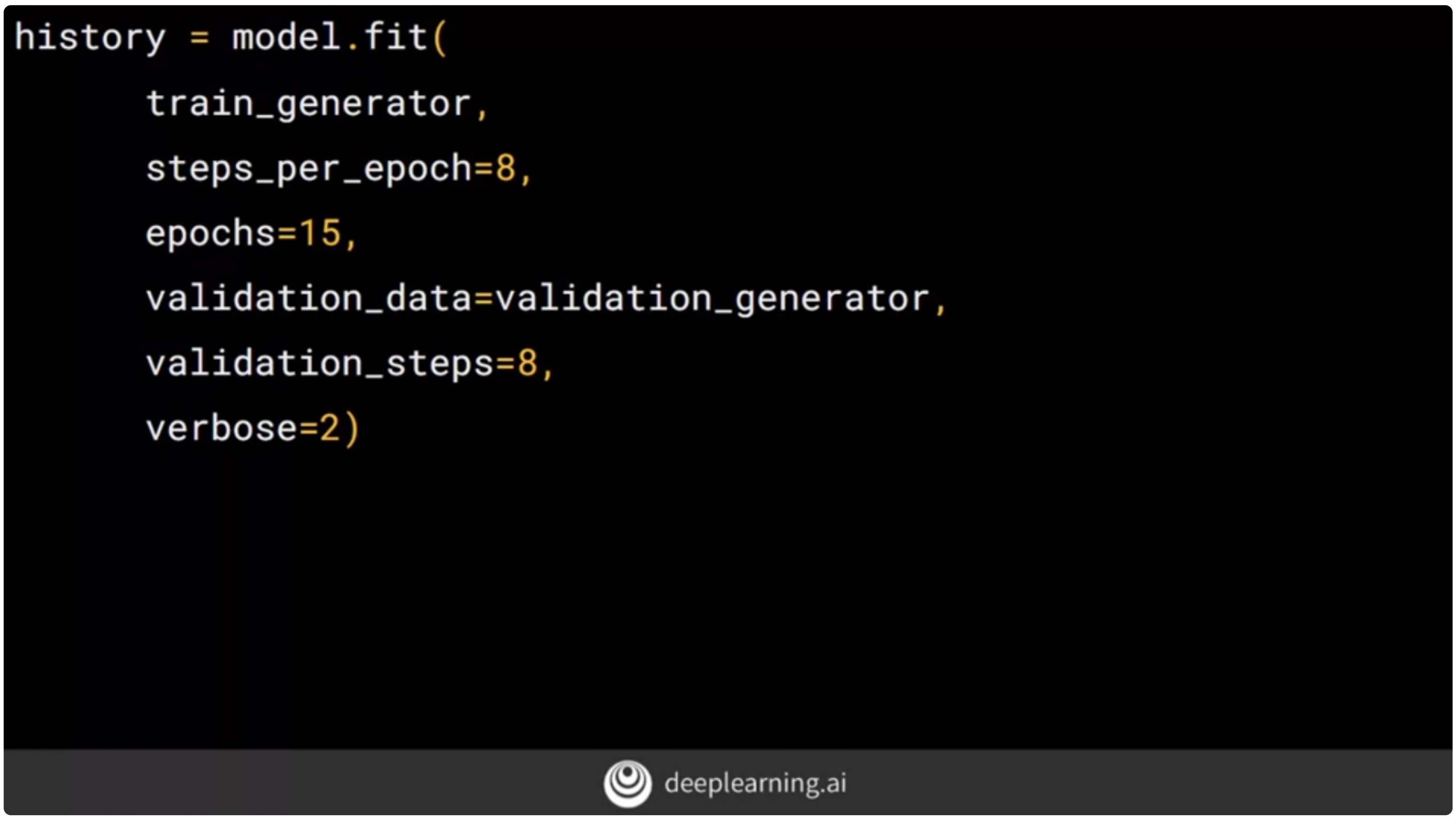
```
train_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
        train_dir,
        target_size=(300, 300),
        batch_size=128,
        class_mode='binary')
```

[flow_from_directory]

`train_dir` will point to `Images/Training` and `Images/Training/Horses` or `Images/Training/Humans` `target_size` defines the new pixel based size of the input images. The function preprocess the images to confirm all instances resolution to this value.

`batch_szie` the function will load the images in batches. this value decides the number of images loaded in each batch

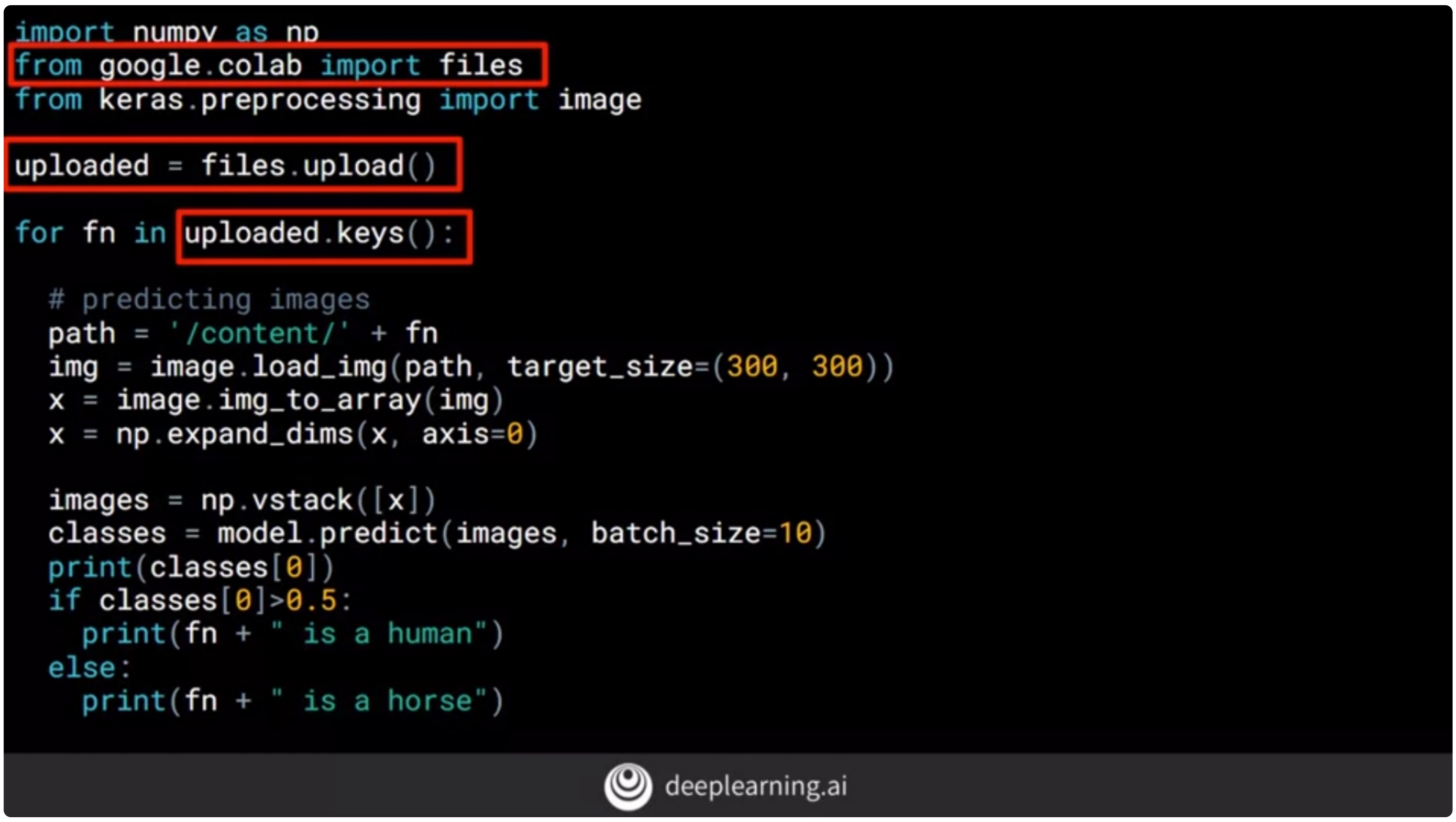
`class_mode` Determines the type of label arrays that are returned:



```
history = model.fit(
      train_generator,
      steps_per_epoch=8,
      epochs=15,
      validation_data=validation_generator,
      validation_steps=8,
      verbose=2)
```

`steps_per_epoch` : since we are using an imagedatagenerator which loads images in batches, we would need to run the generator for 8 times each taining cycle or epoch to load all training images (total training images: 1024 and with batch size for train set set to 128, one needs 8 (1024/128) reruns of generator for loading all images)

`verbose` controls the amount of information to be shown while training



```
import numpy as np
from google.colab import files
from keras.preprocessing import image

uploaded = files.upload()

for fn in uploaded.keys():

  # predicting images
  path = '/content/' + fn
  img = image.load_img(path, target_size=(300, 300))
  x = image.img_to_array(img)
  x = np.expand_dims(x, axis=0)

  images = np.vstack([x])
  classes = model.predict(images, batch_size=10)
  print(classes[0])
  if classes[0]>0.5:
    print(fn + " is a human")
  else:
    print(fn + " is a horse")
```

Highlighted code is colab specific for providing user with options to upload the image path.

The path provided will be used to get all the images loaded at specific `target_size` or resolution on which model has been trained.

This is follwed by prediction of the images using the trained model.