```
# example of using a pre-trained model as a classifier
from tensorflow.keras.preprocessing.image import load_img
from tensorflow.keras.preprocessing.image import img_to_array
from keras.applications.vgg16 import preprocess_input
from keras.applications.vgg16 import decode_predictions
from keras.applications.vgg16 import VGG16

from google.colab import files


uploaded = files.upload()


# load an image from file
image = load_img('download2.jpg', target_size=(224, 224))
# convert the image pixels to a numpy array
image = img_to_array(image)
# reshape data for the model
image = image.reshape((1, image.shape[0], image.shape[1], image.shape[2]))
# prepare the image for the VGG model
image = preprocess_input(image)
# load the model
model = VGG16()
# predict the probability across all output classes
yhat = model.predict(image)
# convert the probabilities to class labels
label = decode_predictions(yhat)
# retrieve the most likely result, e.g. highest probability
label = label[0][0]
# print the classification
print('%s (%.2f%%)' % (label[1], label[2]*100))
```

Choose Files download2.jpg

- **download2.jpg**(image/jpeg) - 172929 bytes, last modified: 11/4/2023 - 100% done

```
Saving download2.jpg to download2.jpg
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg16/vgg16_weights_tf_dim_ordering_tf_kernels.h5
553467096/553467096 [==============================] - 4s 0us/step
1/1 [==============================] - 1s 881ms/step
Downloading data from https://storage.googleapis.com/download.tensorflow.org/data/imagenet_class_index.json
35363/35363 [==============================] - 0s 0us/step
Persian_cat (23.05%)
```

```
image
```

```
array([[[[  54.060997,   87.221   ,  104.32    ],
         [  54.060997,   87.221   ,  104.32    ],
         [  54.060997,   87.221   ,  104.32    ],
         ...,
         [  54.060997,   87.221   ,  104.32    ],
         [  52.060997,   86.221   ,  108.32    ],
         [  52.060997,   86.221   ,  108.32    ]],

        [[  54.060997,   87.221   ,  104.32    ],
         [  54.060997,   87.221   ,  104.32    ],
         [  54.060997,   87.221   ,  104.32    ],
         ...,
         [  55.060997,   87.221   ,  105.32    ],
         [  54.060997,   86.221   ,  108.32    ],
         [  54.060997,   86.221   ,  108.32    ]],

        [[  54.060997,   87.221   ,  104.32    ],
         [  54.060997,   87.221   ,  104.32    ],
         [  54.060997,   87.221   ,  104.32    ],
         ...,
         [  55.060997,   87.221   ,  105.32    ],
         [  54.060997,   86.221   ,  108.32    ],
         [  54.060997,   86.221   ,  108.32    ]],

        ...,

        [[ -54.939003,  -63.779   ,  -59.68    ],
         [ -68.939   ,  -76.779   ,  -74.68    ],
         [ -68.939   ,  -76.779   ,  -74.68    ],
         ...,
         [ -88.939   ,  -98.779   , -104.68    ],
         [ -65.939   ,  -76.779   ,  -82.68    ],
         [ -65.939   ,  -76.779   ,  -82.68    ]],

        [[ -55.939003,  -63.779   ,  -60.68    ],
         [ -64.939   ,  -72.779   ,  -70.68    ],
         [ -64.939   ,  -72.779   ,  -70.68    ],
         ...,
         [ -81.939   ,  -91.779   ,  -97.68    ],
         [ -78.939   ,  -89.779   ,  -95.68    ],
         [ -78.939   ,  -89.779   ,  -95.68    ]],

        [[ -55.939003,  -63.779   ,  -60.68    ],
         [ -64.939   ,  -72.779   ,  -70.68    ],
         [ -64.939   ,  -72.779   ,  -70.68    ],
         ...,
         [ -81.939   ,  -91.779   ,  -97.68    ],
         [ -78.939   ,  -89.779   ,  -95.68    ],
         [ -78.939   ,  -89.779   ,  -95.68    ]]]], dtype=float32)
```

```
import matplotlib.pyplot as plt
plt.imshow(image[0])
plt.show()
```

```
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB d
```