four

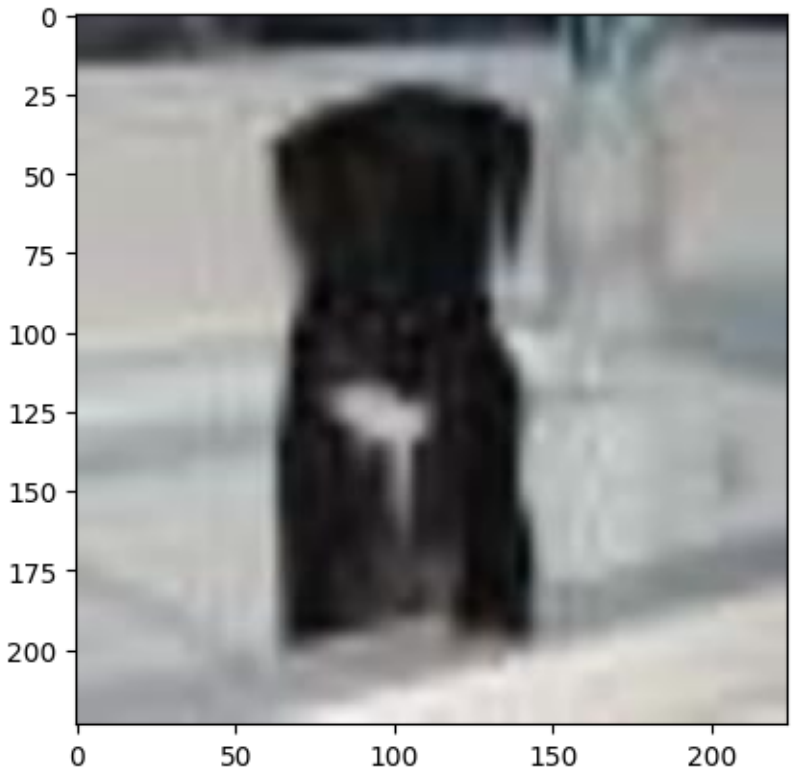
April 17, 2024

[8]:
```
import os
# counting the number of files in train folder
path, dirs, files = next(os.walk('./images'))
file_count = len(files)
print('Number of images: ', file_count)
```

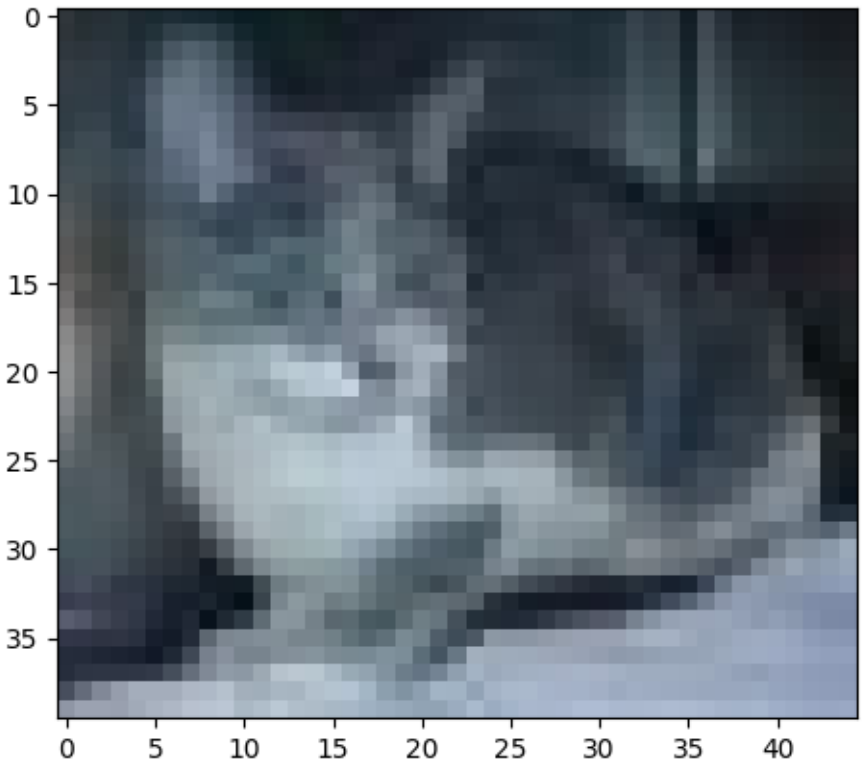
```
Number of images:  25000
```

[4]:
```
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
from sklearn.model_selection import train_test_split
import cv2
```

[5]:
```
# display dog image
img = cv2.imread('./resized/dog.8298.jpg')
imgplt = plt.imshow(img)
plt.show()
```



[6]:
```
# display cat image
img = cv2.imread('./images/cat.4352.jpg')
imgplt = plt.imshow(img)
plt.show()
```



[10]:
```
file_names = os.listdir('./images')

for i in range(5):

  name = file_names[i]
  print(name[0:3])
```

```
cat
cat
cat
cat
cat
```

[ ]:
```
file_names = os.listdir('./images')

dog_count = 0
cat_count = 0

for img_file in file_names:

  name = img_file[0:3]
```

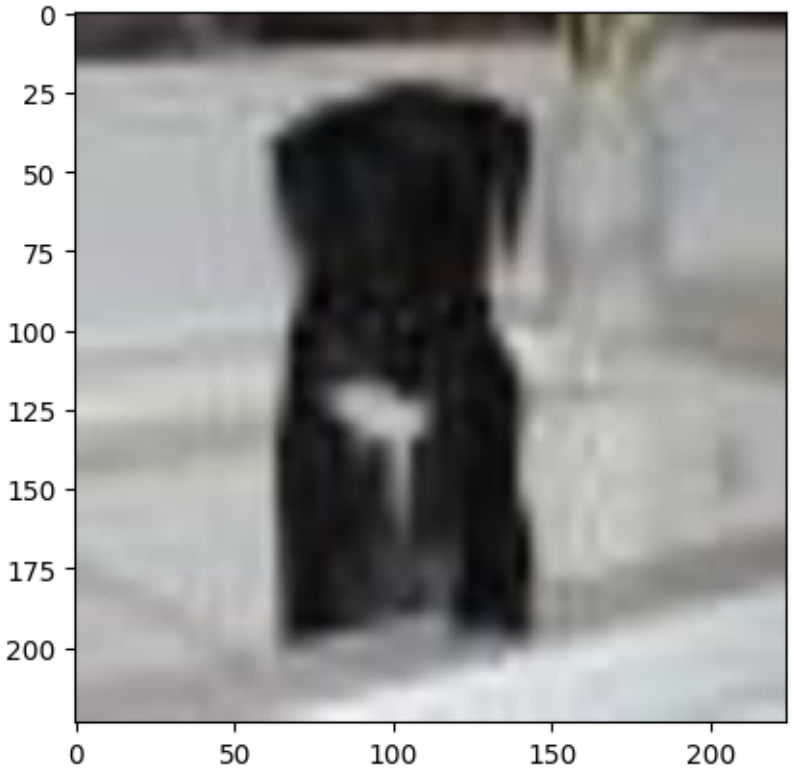
```
  if name == 'dog':
    dog_count += 1

  else:
    cat_count += 1

print('Number of dog images =', dog_count)
print('Number of cat images =', cat_count)
```

```
Number of dog images = 12500
Number of cat images = 12500
```

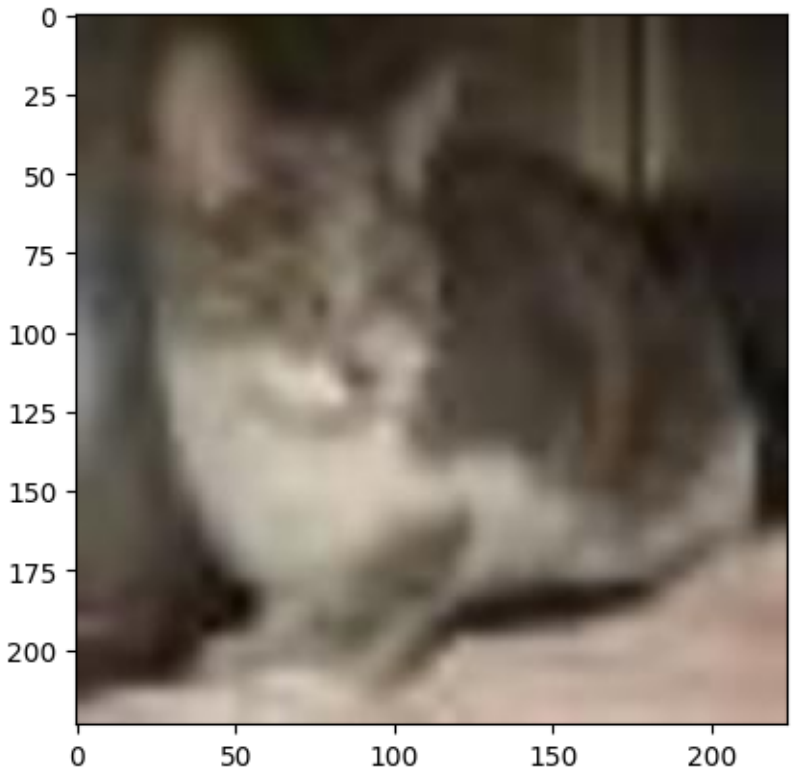
[ ]:
```
#creating a directory for resized images
os.mkdir('./resized')
```

[ ]:
```
original_folder = '/content/train/'
resized_folder = '/content/image resized/'

for i in range(2000):

  filename = os.listdir(original_folder)[i]
  img_path = original_folder+filename

  img = Image.open(img_path)
  img = img.resize((224, 224))
  img = img.convert('RGB')

  newImgPath = resized_folder+filename
  img.save(newImgPath)
```

[11]:
```
# display resized dog image
img = mpimg.imread('./resized/dog.8298.jpg')
imgplt = plt.imshow(img)
plt.show()
```



[12]:
```
# display resized cat image
img = mpimg.imread('./resized/cat.4352.jpg')
imgplt = plt.imshow(img)
plt.show()
```



[ ]:
```
# creaing a for loop to assign labels
filenames = os.listdir('./resized')


labels = []

for i in range(25000):

  file_name = filenames[i]
  label = file_name[0:3]

  if label == 'dog':
    labels.append(1)

  else:
    labels.append(0)
```

[14]:
```
import cv2
import glob
```

[ ]:
```
image_directory = './resized/'
image_extension = ['png', 'jpg']

files = []

[files.extend(glob.glob(image_directory + '*.' + e)) for e in image_extension]

dog_cat_images = np.asarray([cv2.imread(file) for file in files])
```

[ ]:
```
print(dog_cat_images)
```

```
[[[[ 79  93 142]
   [ 67  79 127]
   [ 77  87 135]
   …
   [121 113 113]
   [130 117 119]
   [129 116 118]]

  [[ 43  54 106]
   [ 45  56 106]
   [ 72  79 128]
   …
   [121 113 113]
   [131 119 119]
   [131 119 119]]

  [[ 64  71 126]
   [ 77  85 138]
   [103 110 160]
   …
   [120 113 110]
   [131 119 117]
   [132 120 118]]

  …

  [[ 73  79  98]
   [ 73  79  98]
   [ 76  82 101]
   …
   [121  99  88]
   [115  93  81]
   [130 108  96]]

  [[ 59  65  84]
   [ 66  72  91]
```

```
  [ 80  86 105]
  …
  [121 101  90]
  [134 115 102]
  [175 156 143]]

 [[ 84  90 109]
  [ 83  89 108]
  [ 83  89 108]
  …
  [118  98  87]
  [149 130 117]
  [212 193 180]]]


[[[  1 248 228]
  [ 13 255 237]
  [ 20 251 242]
  …
  [  8 177 235]
  [  2 174 232]
  [  0 172 229]]

 [[ 12 254 236]
  [ 18 255 241]
  [ 20 248 241]
  …
  [ 11 180 238]
  [  6 178 236]
  [  2 177 234]]

 [[ 24 253 244]
  [ 29 255 248]
  [ 25 245 245]
  …
  [ 13 182 240]
  [  8 180 238]
  [  5 180 237]]

 …

 [[255 255 255]
  [255 255 255]
  [255 255 255]
  …
  [169 224 251]
  [169 223 254]
  [169 223 254]]
```

```
 [[255 255 255]
  [255 255 255]
  [255 255 255]
  …
  [170 225 252]
  [170 224 255]
  [171 225 255]]

 [[255 255 255]
  [255 255 255]
  [255 255 255]
  …
  [171 226 253]
  [172 226 255]
  [173 227 255]]]


[[[200 197 176]
  [183 182 162]
  [218 214 203]
  …
  [253 251 250]
  [249 247 247]
  [149 147 147]]

 [[205 202 181]
  [179 178 158]
  [219 215 204]
  …
  [254 252 251]
  [251 249 249]
  [157 155 155]]

 [[203 200 179]
  [164 163 143]
  [212 208 197]
  …
  [255 253 252]
  [253 253 253]
  [167 167 167]]

 …

 [[ 52  55  46]
  [ 53  56  47]
  [ 52  56  51]
  …
```

```
  [151 149 141]
  [156 154 146]
  [160 158 150]]

 [[ 51  54  45]
  [ 53  56  47]
  [ 54  55  51]
  …
  [159 157 149]
  [165 162 154]
  [169 166 158]]

 [[ 52  55  46]
  [ 53  56  47]
  [ 54  55  51]
  …
  [164 162 154]
  [171 168 160]
  [175 172 164]]]


…


[[[  1   0   4]
  [  4   3   7]
  [  7   6   8]
  …
  [ 87  87  73]
  [105 109  90]
  [ 56  61  40]]

 [[  3   2   6]
  [  4   3   7]
  [  5   4   6]
  …
  [ 89  89  75]
  [105 109  90]
  [ 64  69  48]]

 [[  4   3   7]
  [  4   3   7]
  [  3   2   4]
  …
  [ 92  92  78]
  [103 107  88]
  [ 75  80  59]]
```

```
 …

 [[123 121  91]
  [126 124  94]
  [129 127  97]
  …
  [126 123 115]
  [127 124 116]
  [128 125 117]]

 [[124 122  92]
  [124 122  92]
  [125 123  93]
  …
  [125 122 114]
  [126 123 115]
  [128 125 117]]

 [[129 127  97]
  [128 126  96]
  [126 124  94]
  …
  [126 123 115]
  [128 125 117]
  [130 127 119]]]


[[[131 133 144]
  [143 145 155]
  [120 121 131]
  …
  [168 159 162]
  [189 178 181]
  [195 184 187]]

 [[129 131 142]
  [143 145 155]
  [119 120 130]
  …
  [175 166 169]
  [192 181 184]
  [191 180 183]]

 [[126 128 139]
  [144 146 156]
  [118 119 129]
  …
  [184 175 178]
```

```
  [198 187 190]
  [190 179 182]]

 …

 [[210 212 212]
  [197 197 197]
  [183 181 181]
  …
  [ 72  70  89]
  [ 65  63  82]
  [ 75  73  92]]

 [[209 211 211]
  [199 199 199]
  [186 184 184]
  …
  [ 73  71  90]
  [ 91  89 108]
  [116 114 133]]

 [[209 211 211]
  [200 200 200]
  [187 185 185]
  …
  [ 89  87 106]
  [135 133 152]
  [157 155 174]]]


[[[126 115 161]
  [119 108 154]
  [115 104 150]
  …
  [ 56  44  62]
  [ 66  51  72]
  [ 73  56  77]]

 [[173 163 205]
  [163 153 195]
  [154 144 186]
  …
  [ 78  67  87]
  [ 87  72  93]
  [ 83  68  89]]

 [[225 216 249]
  [215 206 239]
```

```
 [205 196 229]
 …
 [107  98 119]
 [115 102 124]
 [102  89 111]]

…

[[249 242 239]
 [245 238 235]
 [245 238 235]
 …
 [ 46  41  40]
 [ 65  60  59]
 [ 78  73  72]]

[[255 251 248]
 [253 246 243]
 [253 246 243]
 …
 [ 27  22  21]
 [ 44  39  38]
 [ 56  51  50]]

[[251 244 241]
 [247 240 237]
 [245 238 235]
 …
 [ 11   6   5]
 [ 22  17  16]
 [ 32  27  26]]]]
```

[ ]:
```
type(dog_cat_images)
```

[ ]:
```
numpy.ndarray
```

[ ]:
```
X = dog_cat_images
Y = np.asarray(labels)
```

[ ]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2,␣
↪random_state=2)
```

[ ]:
```
# scaling the data
X_train_scaled = X_train/255

X_test_scaled = X_test/255
```

[ ]:
```
print(X_train_scaled)
```

```
[[[[0.75686275 0.9254902  0.88235294]
   [0.79215686 0.96078431 0.91764706]
   [0.77254902 0.95294118 0.90588235]
   …
   [0.53333333 0.92156863 0.84313725]
   [0.50588235 0.90196078 0.81960784]
   [0.5372549  0.94117647 0.85882353]]

  [[0.75294118 0.94117647 0.89411765]
   [0.76862745 0.95686275 0.90980392]
   [0.76078431 0.94901961 0.90196078]
   …
   [0.57647059 0.96470588 0.88627451]
   [0.49019608 0.88627451 0.80392157]
   [0.4627451  0.86666667 0.78431373]]

  [[0.6745098  0.88627451 0.83529412]
   [0.65882353 0.87843137 0.82745098]
   [0.6627451  0.88235294 0.83137255]
   …
   [0.59215686 0.96862745 0.90196078]
   [0.50196078 0.89019608 0.81960784]
   [0.48627451 0.87843137 0.80784314]]


  …

  [[0.50588235 0.88627451 0.78039216]
   [0.38431373 0.75686275 0.65098039]
   [0.39607843 0.74901961 0.65882353]
   …
   [0.54117647 0.87058824 0.8       ]
   [0.57647059 0.87843137 0.82352941]
   [0.60784314 0.89019608 0.83921569]]

  [[0.49411765 0.88627451 0.77647059]
   [0.41176471 0.78431373 0.67843137]
   [0.4        0.74509804 0.65490196]
   …
   [0.52941176 0.88627451 0.80392157]
   [0.49803922 0.83529412 0.76470588]
   [0.56862745 0.89803922 0.82745098]]

  [[0.48627451 0.88627451 0.77647059]
   [0.35686275 0.7372549  0.63137255]
   [0.46666667 0.8        0.71372549]
```

```
  …
   [0.47058824 0.84313725 0.76078431]
   [0.30588235 0.65882353 0.58431373]
   [0.39215686 0.74117647 0.66666667]]]


[[[0.23921569 0.37254902 0.58039216]
   [0.23529412 0.36862745 0.57647059]
   [0.23529412 0.37254902 0.56862745]
   …
   [0.14117647 0.17254902 0.23921569]
   [0.14509804 0.16470588 0.24705882]
   [0.14901961 0.16862745 0.25098039]]

  [[0.23921569 0.37254902 0.58039216]
   [0.23529412 0.37254902 0.56862745]
   [0.23529412 0.37254902 0.56862745]
   …
   [0.1372549  0.16470588 0.23137255]
   [0.1372549  0.16078431 0.23529412]
   [0.1372549  0.16078431 0.23529412]]

  [[0.23137255 0.37647059 0.57254902]
   [0.22745098 0.37254902 0.56862745]
   [0.23529412 0.37254902 0.56862745]
   …
   [0.14509804 0.16078431 0.23137255]
   [0.14509804 0.16078431 0.23137255]
   [0.14509804 0.16078431 0.23137255]]

 …

  [[0.04313725 0.18823529 0.38431373]
   [0.03921569 0.17647059 0.37254902]
   [0.03529412 0.16078431 0.36862745]
   …
   [0.05098039 0.08235294 0.2       ]
   [0.04705882 0.07843137 0.19607843]
   [0.04705882 0.07843137 0.19607843]]

  [[0.03921569 0.18823529 0.37647059]
   [0.03921569 0.17647059 0.37254902]
   [0.03137255 0.15686275 0.36470588]
   …
   [0.05098039 0.08235294 0.2       ]
   [0.04705882 0.07843137 0.19607843]
   [0.04705882 0.07843137 0.19607843]]
```

```
 [[0.03921569 0.18823529 0.37647059]
  [0.03529412 0.17647059 0.36470588]
  [0.03137255 0.15686275 0.36470588]
  …
  [0.05098039 0.08235294 0.2       ]
  [0.04705882 0.07843137 0.19607843]
  [0.04705882 0.07843137 0.19607843]]]


[[[0.1372549  0.14509804 0.14509804]
  [0.12941176 0.1372549  0.1372549 ]
  [0.11764706 0.1254902  0.1254902 ]
  …
  [0.81960784 0.81176471 0.78039216]
  [0.82352941 0.81568627 0.78431373]
  [0.82352941 0.81568627 0.78431373]]

 [[0.1372549  0.14509804 0.14509804]
  [0.12941176 0.1372549  0.1372549 ]
  [0.11764706 0.1254902  0.1254902 ]
  …
  [0.81960784 0.81176471 0.78039216]
  [0.82352941 0.81568627 0.78431373]
  [0.82352941 0.81568627 0.78431373]]

 [[0.13333333 0.14117647 0.14117647]
  [0.1254902  0.13333333 0.13333333]
  [0.10980392 0.11764706 0.11764706]
  …
  [0.81960784 0.81176471 0.78039216]
  [0.82352941 0.81568627 0.78431373]
  [0.82745098 0.81960784 0.78823529]]

 …

 [[0.28235294 0.4627451  0.50588235]
  [0.30588235 0.48235294 0.5254902 ]
  [0.3372549  0.49019608 0.54509804]
  …
  [0.78039216 0.81176471 0.83921569]
  [0.75294118 0.79215686 0.81960784]
  [0.71764706 0.76470588 0.78823529]]

 [[0.26666667 0.45098039 0.48235294]
  [0.29803922 0.4745098  0.51764706]
  [0.33333333 0.49803922 0.54901961]
  …
  [0.75686275 0.78823529 0.78431373]
```

```
  [0.71372549 0.75686275 0.74901961]
  [0.6627451  0.71372549 0.70588235]]

 [[0.25098039 0.43529412 0.46666667]
  [0.28627451 0.46666667 0.49803922]
  [0.32941176 0.49411765 0.54509804]
  …
  [0.7254902  0.76078431 0.74901961]
  [0.67058824 0.70980392 0.68627451]
  [0.61176471 0.65882353 0.63529412]]]


…


[[[0.55294118 0.56862745 0.54901961]
  [0.54901961 0.56470588 0.54509804]
  [0.54509804 0.56078431 0.54117647]
  …
  [0.33333333 0.37647059 0.39215686]
  [0.33333333 0.37647059 0.39215686]
  [0.33333333 0.37647059 0.39215686]]

 [[0.56078431 0.57647059 0.55686275]
  [0.55686275 0.57254902 0.55294118]
  [0.55686275 0.57254902 0.55294118]
  …
  [0.3372549  0.38039216 0.39607843]
  [0.33333333 0.37647059 0.39215686]
  [0.33333333 0.37647059 0.39215686]]

 [[0.57254902 0.58823529 0.56862745]
  [0.56862745 0.58431373 0.56470588]
  [0.56862745 0.58431373 0.56470588]
  …
  [0.3372549  0.38431373 0.39215686]
  [0.3372549  0.38431373 0.39215686]
  [0.3372549  0.38431373 0.39215686]]

 …

 [[0.98823529 0.98431373 0.99215686]
  [0.98431373 0.98039216 0.98823529]
  [0.97254902 0.96862745 0.97647059]
  …
  [0.68235294 0.65882353 0.63137255]
  [0.68235294 0.65882353 0.63137255]
  [0.68235294 0.65882353 0.63137255]]
```

```
 [[0.97647059 0.97254902 0.98039216]
  [0.97647059 0.97254902 0.98039216]
  [0.97254902 0.96862745 0.97647059]
  …
  [0.68235294 0.65882353 0.63137255]
  [0.68235294 0.65882353 0.63137255]
  [0.68235294 0.65882353 0.63137255]]

 [[0.96862745 0.96470588 0.97254902]
  [0.96862745 0.96470588 0.97254902]
  [0.96862745 0.96470588 0.97254902]
  …
  [0.67843137 0.65490196 0.62745098]
  [0.68235294 0.65882353 0.63137255]
  [0.68235294 0.65882353 0.63137255]]]


[[[0.17254902 0.18431373 0.17647059]
  [0.17254902 0.18431373 0.17647059]
  [0.17254902 0.18431373 0.17647059]
  …
  [0.10980392 0.15294118 0.18431373]
  [0.09803922 0.13333333 0.16862745]
  [0.08627451 0.12156863 0.15686275]]

 [[0.17254902 0.18431373 0.17647059]
  [0.17254902 0.18431373 0.17647059]
  [0.17254902 0.18431373 0.17647059]
  …
  [0.10196078 0.14509804 0.17647059]
  [0.09411765 0.12941176 0.16470588]
  [0.08235294 0.11764706 0.15294118]]

 [[0.16862745 0.18039216 0.17254902]
  [0.17254902 0.18431373 0.17647059]
  [0.17254902 0.18431373 0.17647059]
  …
  [0.09411765 0.1372549  0.16862745]
  [0.09411765 0.12941176 0.16470588]
  [0.08627451 0.12156863 0.15686275]]

 …

 [[0.40784314 0.50196078 0.54901961]
  [0.43137255 0.5254902  0.57254902]
  [0.40392157 0.49803922 0.54509804]
  …
```

```
  [0.3254902  0.39607843 0.42352941]
  [0.31372549 0.38431373 0.41176471]
  [0.30980392 0.38039216 0.40784314]]

 [[0.39215686 0.48627451 0.53333333]
  [0.45882353 0.55294118 0.6       ]
  [0.45098039 0.54509804 0.59215686]
  …
  [0.32941176 0.4        0.42745098]
  [0.3254902  0.39607843 0.42352941]
  [0.30980392 0.38039216 0.40784314]]

 [[0.36862745 0.4627451  0.50980392]
  [0.4745098  0.56862745 0.61568627]
  [0.48627451 0.58039216 0.62745098]
  …
  [0.34117647 0.41176471 0.43921569]
  [0.35686275 0.42745098 0.45490196]
  [0.33333333 0.40392157 0.43137255]]]


[[[0.01960784 0.02745098 0.02745098]
  [0.01960784 0.02745098 0.02745098]
  [0.01960784 0.02745098 0.02745098]
  …
  [0.00784314 0.03137255 0.08235294]
  [0.01568627 0.03529412 0.09411765]
  [0.01960784 0.03529412 0.10588235]]

 [[0.01960784 0.02745098 0.02745098]
  [0.01960784 0.02745098 0.02745098]
  [0.01960784 0.02745098 0.02745098]
  …
  [0.00784314 0.03137255 0.08235294]
  [0.01568627 0.03529412 0.09411765]
  [0.01960784 0.03529412 0.10588235]]

 [[0.01568627 0.02352941 0.02352941]
  [0.01568627 0.02352941 0.02352941]
  [0.01568627 0.02352941 0.02352941]
  …
  [0.01176471 0.03529412 0.08627451]
  [0.00784314 0.03529412 0.09411765]
  [0.01176471 0.03921569 0.09803922]]

 …

 [[0.16078431 0.27843137 0.41568627]
```

```
   [0.17254902 0.29019608 0.42745098]
   [0.18823529 0.30588235 0.44313725]
   …
   [0.82745098 0.84705882 0.84313725]
   [0.83529412 0.85490196 0.85098039]
   [0.84313725 0.8627451  0.85882353]]

  [[0.16862745 0.28627451 0.42352941]
   [0.18039216 0.29803922 0.43529412]
   [0.19607843 0.31372549 0.45098039]
   …
   [0.83529412 0.85490196 0.85098039]
   [0.84313725 0.8627451  0.85882353]
   [0.84705882 0.86666667 0.8627451 ]]

  [[0.17254902 0.29019608 0.42745098]
   [0.18431373 0.30196078 0.43921569]
   [0.2        0.31764706 0.45490196]
   …
   [0.84313725 0.8627451  0.85882353]
   [0.84705882 0.86666667 0.8627451 ]
   [0.85098039 0.87058824 0.86666667]]]]
```

[ ]:
```
import tensorflow as tf
import tensorflow_hub as hub
```

[ ]:
```
mobilenet_model = 'https://tfhub.dev/google/tf2-preview/mobilenet_v2/
 ↪feature_vector/4'

pretrained_model = hub.KerasLayer(mobilenet_model, input_shape=(224,224,3),␣
 ↪trainable=False)
```

[ ]:
```
num_of_classes = 2

model = tf.keras.Sequential([

    pretrained_model,
    tf.keras.layers.Dense(num_of_classes)

])

model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 keras_layer (KerasLayer)    (None, 1280)              2257984
```

```
 dense (Dense)               (None, 2)                 2562

=================================================================
Total params: 2,260,546
Trainable params: 2,562
Non-trainable params: 2,257,984
_________________________________________________________________
```

[ ]:
```
model.compile(
    optimizer = 'adam',
    loss = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
    metrics = ['acc']
)
```

[ ]:

[ ]:

[ ]:
```
model.fit(X_train_scaled, Y_train, epochs=5)
```

```
Epoch 1/5
50/50 [==============================] - 47s 861ms/step - loss: 0.2163 - acc:
0.9162
Epoch 2/5
50/50 [==============================] - 42s 838ms/step - loss: 0.0746 - acc:
0.9756
Epoch 3/5
50/50 [==============================] - 44s 872ms/step - loss: 0.0552 - acc:
0.9825
Epoch 4/5
50/50 [==============================] - 41s 824ms/step - loss: 0.0417 - acc:
0.9894
Epoch 5/5
50/50 [==============================] - 41s 825ms/step - loss: 0.0345 - acc:
0.9937
```

[ ]:
```
<keras.callbacks.History at 0x7faedc598090>
```

[ ]:
```
score, acc = model.evaluate(X_test_scaled, Y_test)
print('Test Loss =', score)
print('Test Accuracy =', acc)
```

```
13/13 [==============================] - 12s 866ms/step - loss: 0.0812 - acc:
0.9775
Test Loss = 0.0812455490231514
Test Accuracy = 0.9775000214576721
```