assignment06-3_MeyerJake

April 23, 2023

0.1 Assignment 6-3

0.1.1 DSC 650

0.1.2 Jake Meyer

0.1.3 04/23/2023

Load the ResNet50 model. Perform image classification on five to ten images of your choice. They can be personal images or publically available images. Include the images in dsc650/assignments/assignment06/images/. Save the predictions dsc650/assignments/assignment06/results/predictions/resnet50 directory. If you are using JupyterHub, you can include those plots in your Jupyter notebook.

Using code from deep-learning-with-python-notebooks Using ResNet50 function api site

[1]:
```
## Import the necessary modules for the assignment above.
import csv
import cv2
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
import keras
import sklearn
from pathlib import Path
import time
import os

## Import the necessary keras components for the data and CNN
from tensorflow.keras.preprocessing.image import load_img
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.preprocessing import image
from tensorflow.keras.applications.imagenet_utils import decode_predictions
from tensorflow.keras.applications.imagenet_utils import preprocess_input
from tensorflow.keras.applications.resnet50 import ResNet50
from tensorflow.keras.applications import resnet50
import tensorflow.compat.v1 as tf
tf.disable_v2_behavior()
```

```
WARNING:tensorflow:From C:\Users\jkmey\anaconda3\envs\dsc650\lib\site-
```

```
packages\tensorflow\python\compat\v2_compat.py:107: disable_resource_variables
(from tensorflow.python.ops.variable_scope) is deprecated and will be removed in
a future version.
Instructions for updating:
non-resource variables are not supported in the long term
```

[2]:
```
## Print versions of essential packages
print("keras version: {}".format(keras.__version__))
print("tensorflow version: {}".format(tf.__version__))
print("pandas version: {}".format(pd.__version__))
print("numpy version: {}".format(np.__version__))
```

```
keras version: 2.11.0
tensorflow version: 2.11.0
pandas version: 1.5.3
numpy version: 1.24.2
```

[3]:
```
## Setup the directories for the assignment
current_dir = Path('C:/Users/jkmey/Documents/Github/DSC650_Course_Assignments/
 ↪dsc650/dsc650/assignments/assignment06')
image_dir = Path('C:/Users/jkmey/Documents/Github/DSC650_Course_Assignments/
 ↪dsc650/dsc650/assignments/assignment06/images')
results_dir = Path('C:/Users/jkmey/Documents/Github/DSC650_Course_Assignments/
 ↪dsc650/dsc650/assignments/assignment06/').joinpath('results')
predictions_dir = Path('C:/Users/jkmey/Documents/Github/
 ↪DSC650_Course_Assignments/dsc650/dsc650/assignments/assignment06/results').
 ↪joinpath('predictions')
resnet_dir = Path('C:/Users/jkmey/Documents/Github/DSC650_Course_Assignments/
 ↪dsc650/dsc650/assignments/assignment06/results/predictions').
 ↪joinpath('resnet50')
resnet_dir.mkdir(parents = True, exist_ok = True)
```

0.1.4 Load the ResNet50 Model

[4]:
```
## Load the ResNet50 model as shown in the ResNet50 site listed above.
model = ResNet50(weights = 'imagenet')
```

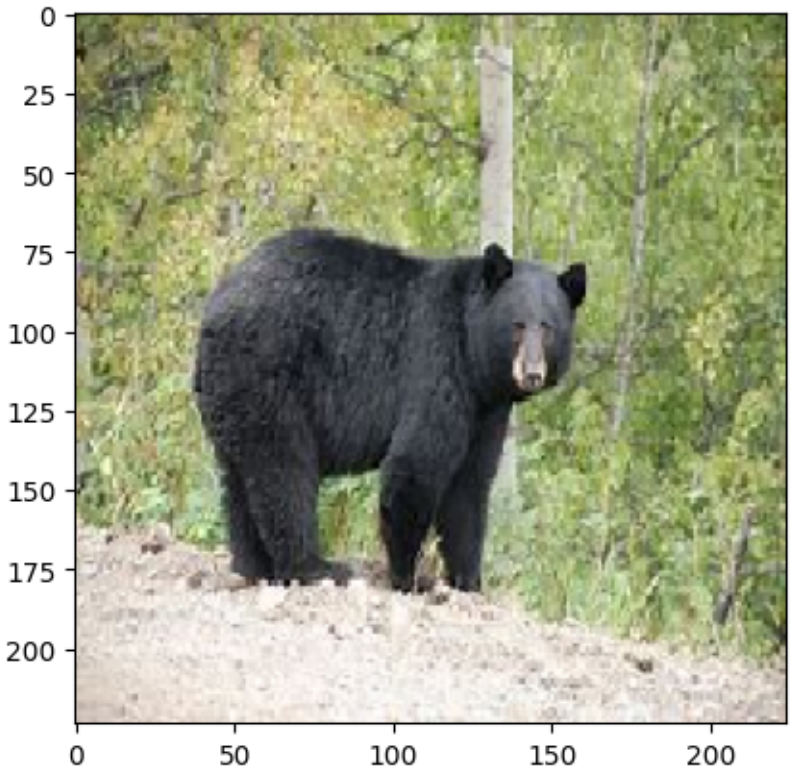
```
WARNING:tensorflow:From C:\Users\jkmey\anaconda3\envs\dsc650\lib\site-
packages\keras\layers\normalization\batch_normalization.py:561: _colocate_with
(from tensorflow.python.framework.ops) is deprecated and will be removed in a
future version.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:OMP_NUM_THREADS is no longer used by the default Keras
config. To configure the number of threads, use tf.config.threading APIs.
```

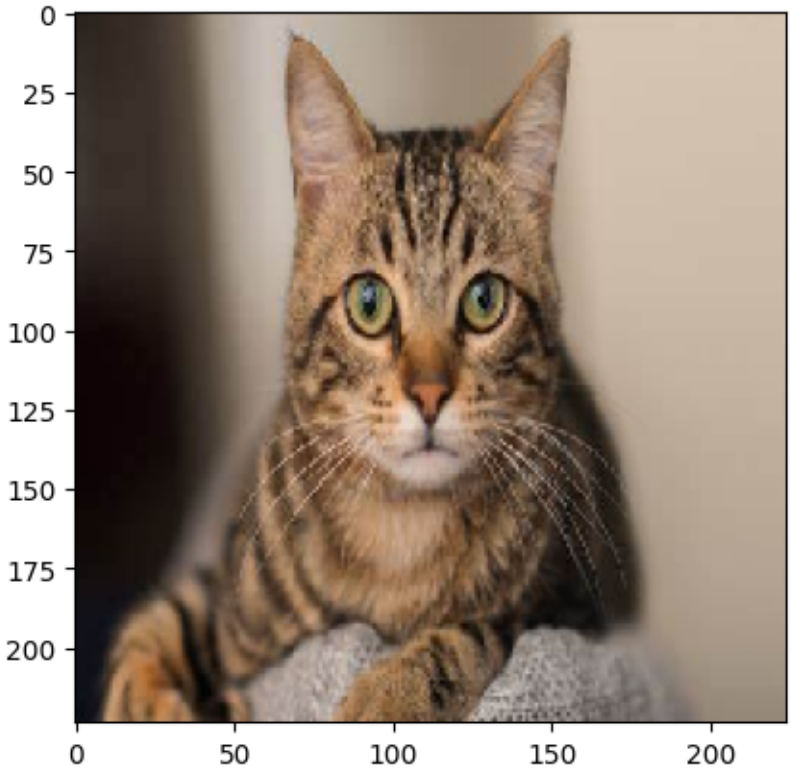
Pulled 10 random images from the internet of various animals. Per the assignment, will write code to classify these images with ResNet50.

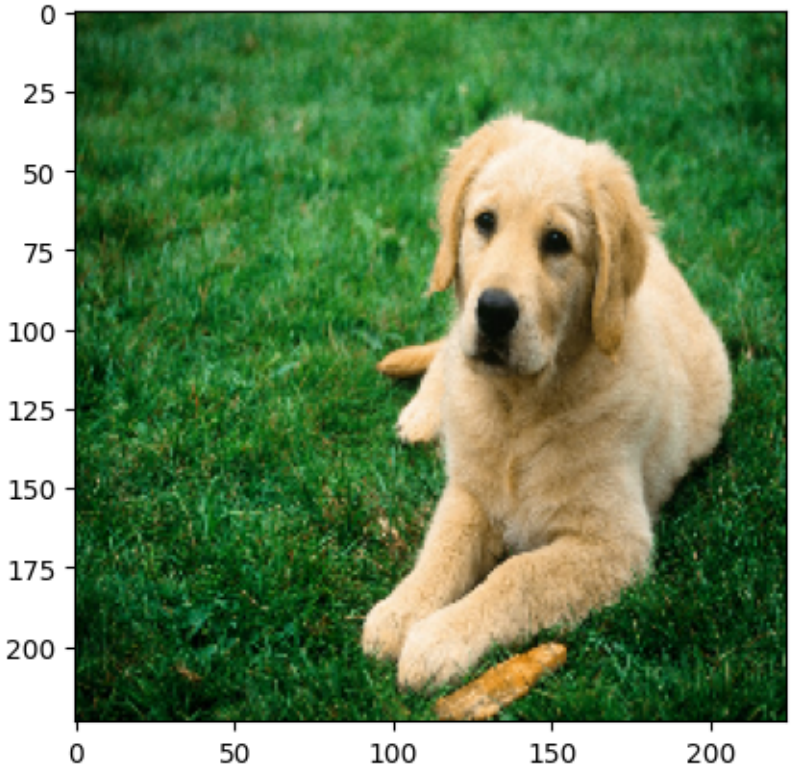
0.1.5 Using ResNet50 Model for Predictions on 10 Images

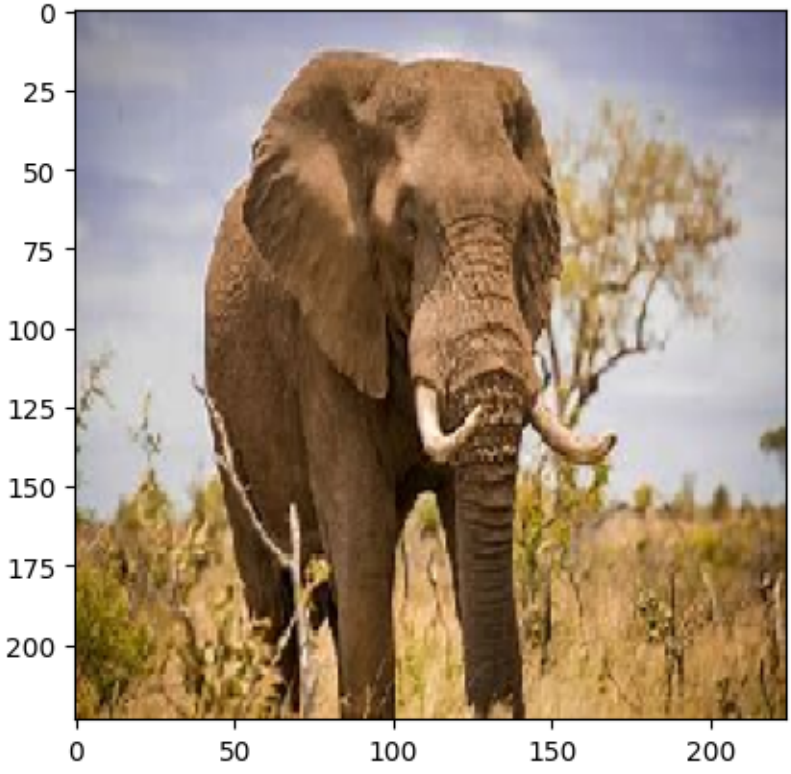
[5]:
```
## Load and resize  (224x224 per the ResNet50 site) for the 10 images.
img1 = image.load_img('images/bear.png', target_size = (224, 224))
img2 = image.load_img('images/cat.png', target_size = (224, 224))
img3 = image.load_img('images/dog.png', target_size = (224, 224))
img4 = image.load_img('images/elephant.png', target_size = (224, 224))
img5 = image.load_img('images/giraffe.png', target_size = (224, 224))
img6 = image.load_img('images/gorilla.png', target_size = (224, 224))
img7 = image.load_img('images/hippo.png', target_size = (224, 224))
img8 = image.load_img('images/leapord.png', target_size = (224, 224))
img9 = image.load_img('images/penguin.png', target_size = (224, 224))
img10 = image.load_img('images/tiger.png', target_size = (224, 224))
```

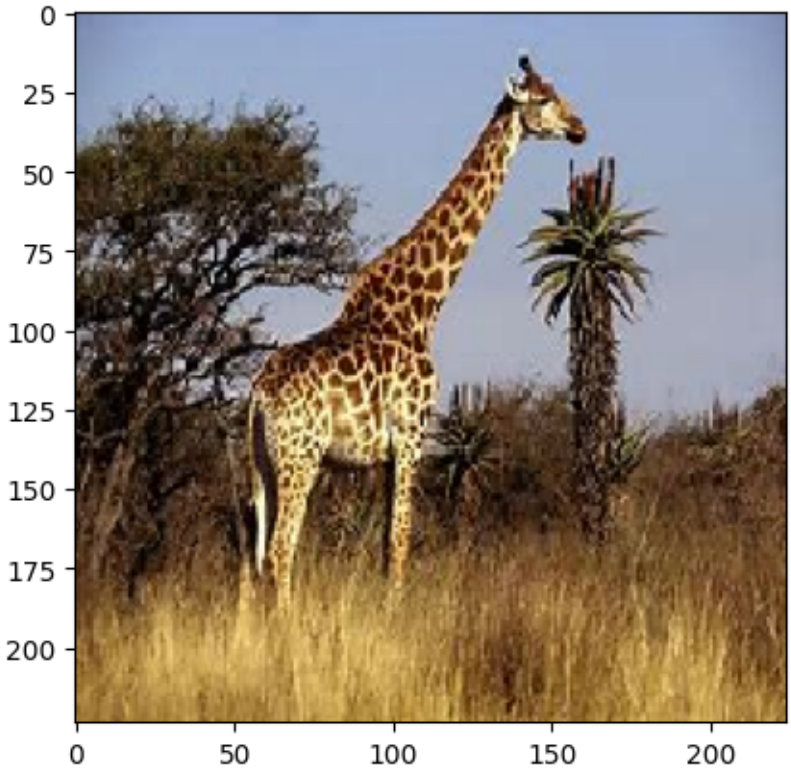
[6]:
```
## Show the 10 images (after resizing) for reference prior to classifying.
image_list = [img1, img2, img3, img4, img5, img6, img7, img8, img9, img10]

for animal in image_list:
    plt.imshow(animal)
    plt.show()
```

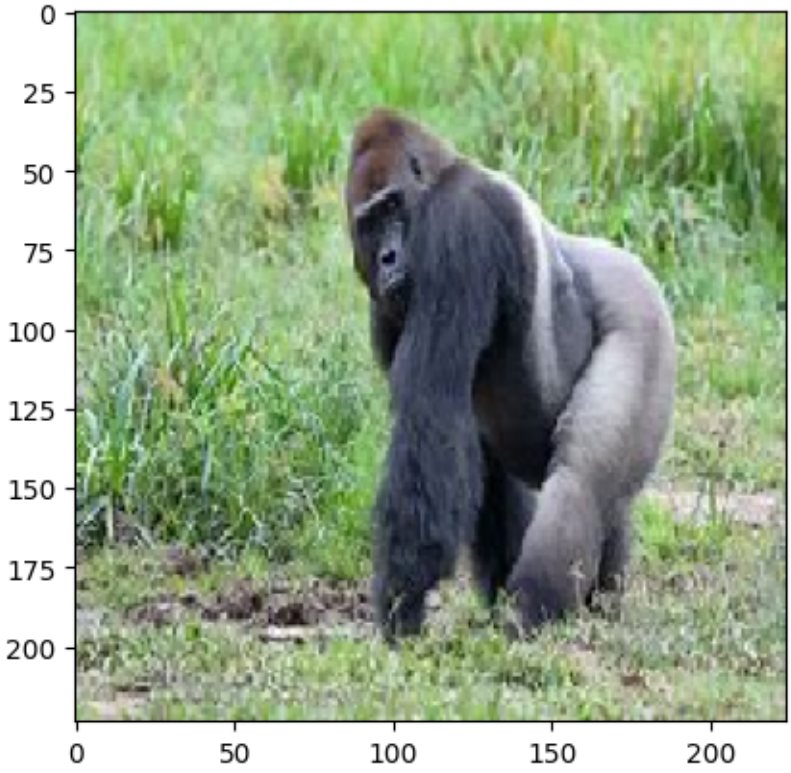


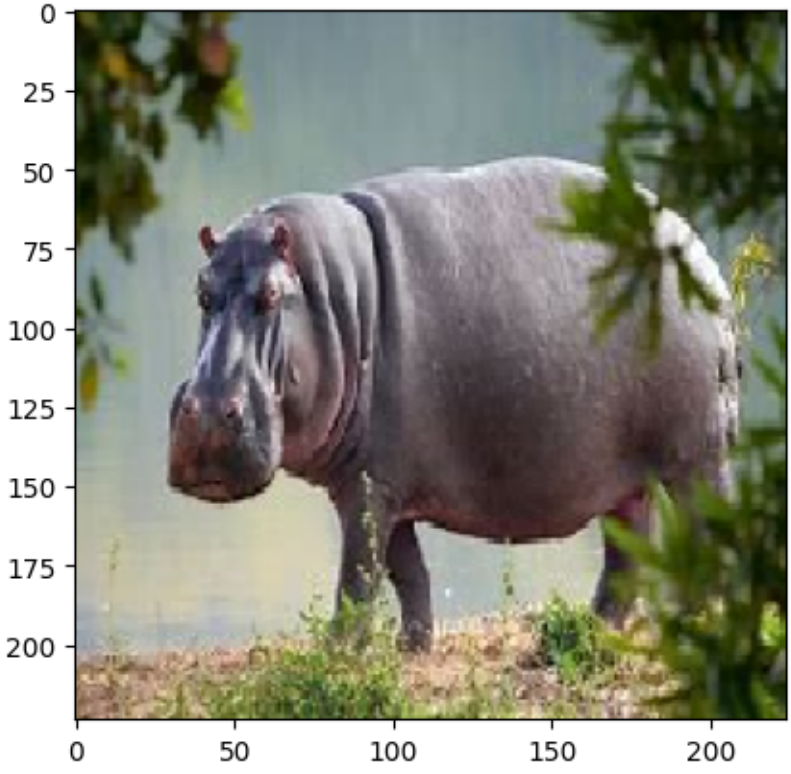

0
25
50
75
100
125
150
175
200
0
50
100
150
200

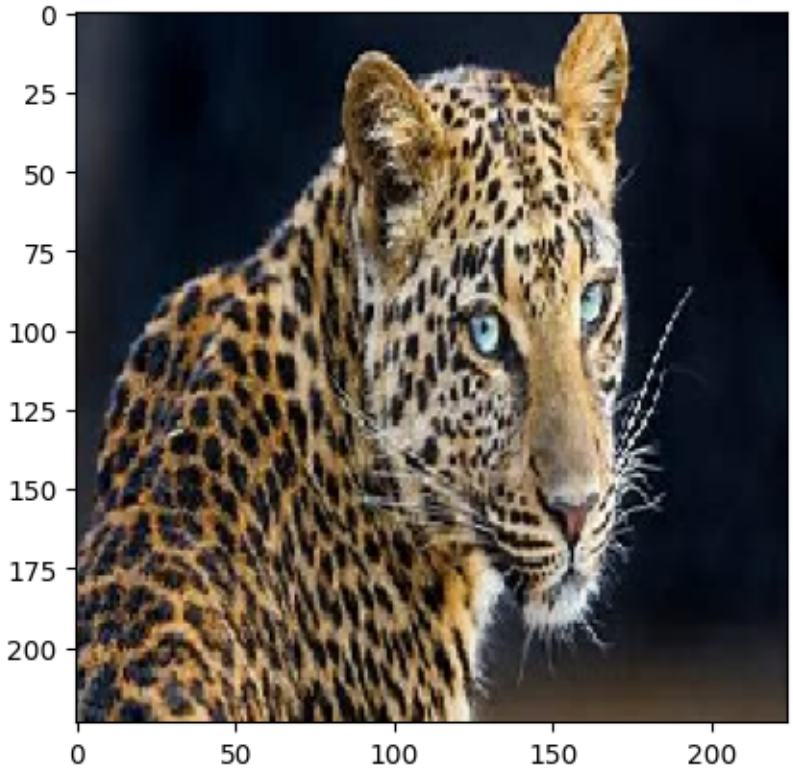

0
25
50
75
100
125
150
175
200
0
50
100
150
200

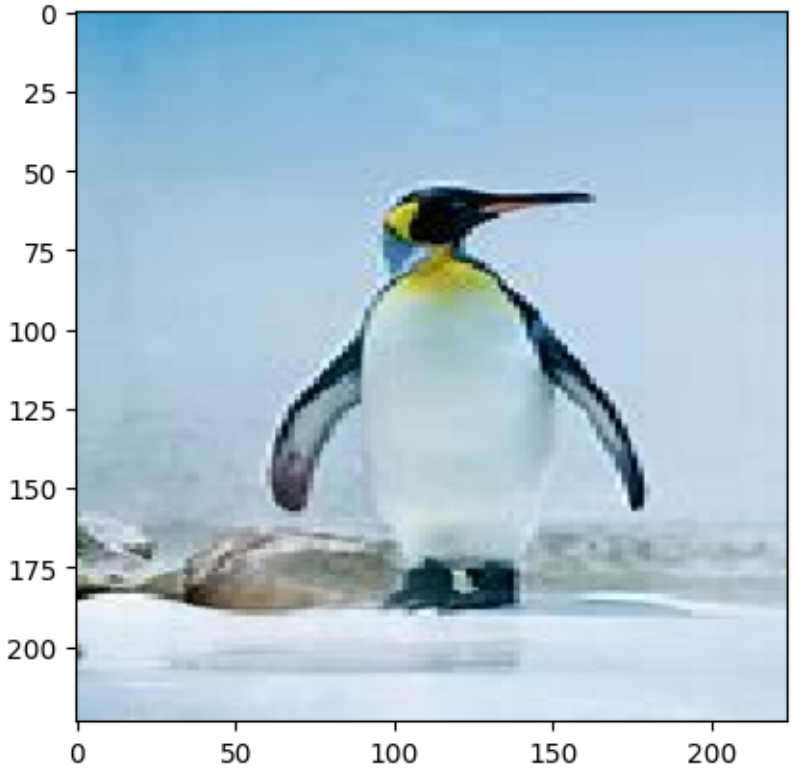

0
25
50
75
100
125
150
175
200
0
50
100
150
200

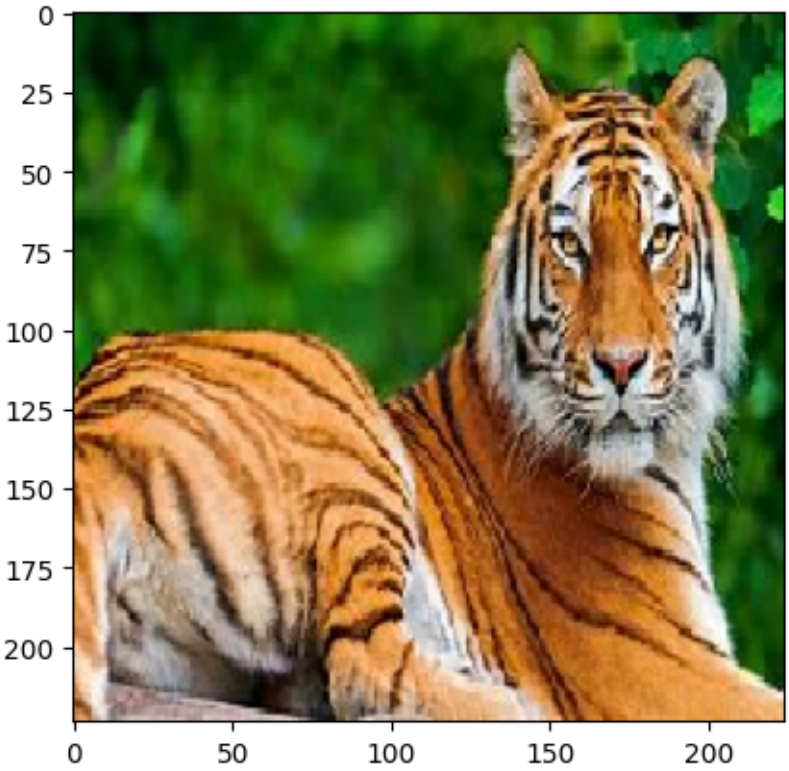

0
25
50
75
100
125
150
175
200
0
50
100
150
200


0
25
50
75
100
125
150
175
200
0
50
100
150
200


0
25
50
75
100
125
150
175
200
0
50
100
150
200


0
25
50
75
100
125
150
175
200
0
50
100
150
200


0
25
50
75
100
125
150
175
200
0
50
100
150
200

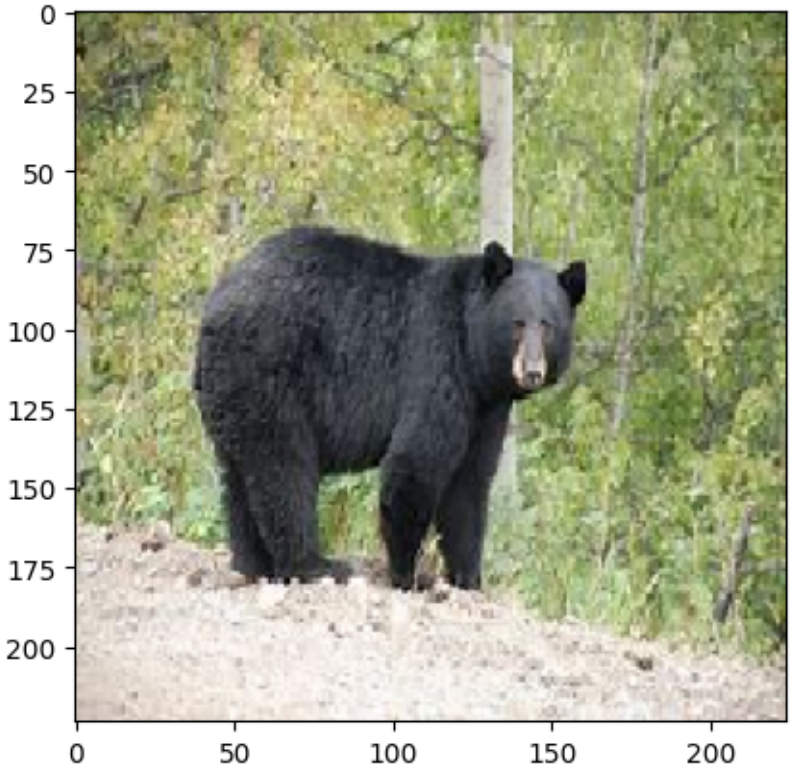


0.1.6 Create the predictions for each image using the ResNet50 model.

[16]:
```
## Adjust the image. Convert to numpy array, add batch dimension, and␣
 ↪preprocess.
image_array1 = image.img_to_array(img1)
image_array1 = np.expand_dims(image_array1, axis =0)
image_array1 = preprocess_input(image_array1)
prediction1 = model.predict(image_array1)

## Print the image and prediction here in the Jupyter Notebook.
print("Prediction: {}".format(decode_predictions(prediction1, top = 1)[0]))
plt.imshow(img1)
plt.show()

## Export the prediction to a prediction file as specified in the document.
resnet50_predictions = resnet_dir.joinpath('assignment06-3_resnet50_predictions.
 ↪csv')

test_dict1 = {'Image 1': str(decode_predictions(prediction1, top = 1)[0])}
```

```
with open(resnet50_predictions, 'w') as csv_file:
    writer = csv.writer(csv_file)
    for key, value in test_dict1.items():
        writer.writerow([key,value])
```

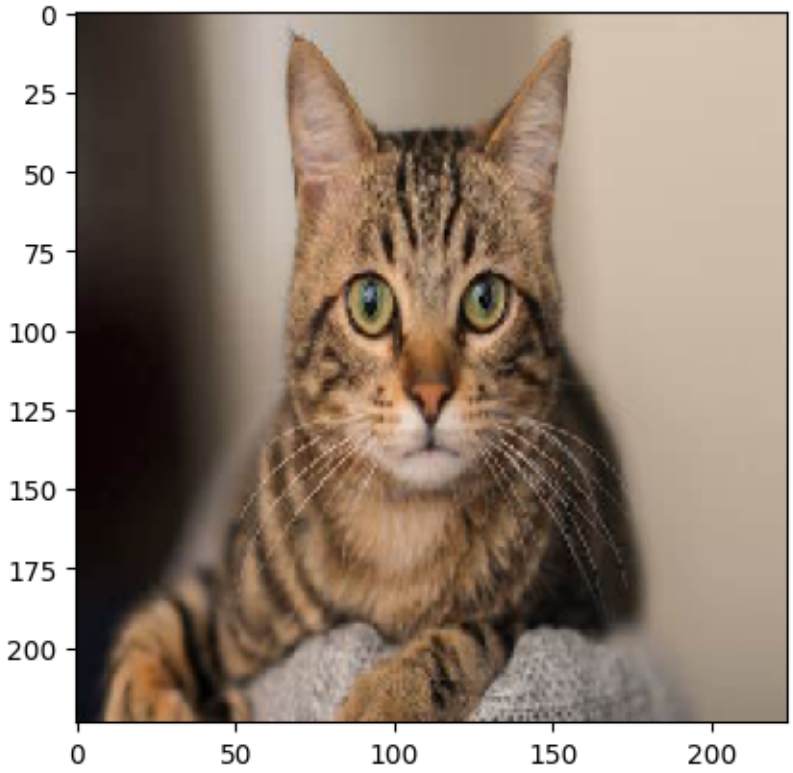
```
Prediction: [('n02133161', 'American_black_bear', 0.99549896)]
```



[17]:
```
## Adjust the image. Convert to numpy array, add batch dimension, and␣
 ↪preprocess.
image_array2 = image.img_to_array(img2)
image_array2 = np.expand_dims(image_array2, axis =0)
image_array2 = preprocess_input(image_array2)
prediction2 = model.predict(image_array2)

## Print the image and prediction here in the Jupyter Notebook.
print("Prediction: {}".format(decode_predictions(prediction2, top = 1)[0]))
plt.imshow(img2)
plt.show()

## Export the prediction to a prediction file as specified in the document.
```

```
resnet50_predictions = resnet_dir.joinpath('assignment06-3_resnet50_predictions.
 ↪csv')

test_dict2 = {'Image 2': str(decode_predictions(prediction2, top = 1)[0])}

with open(resnet50_predictions, 'a') as csv_file:
    writer = csv.writer(csv_file)
    for key, value in test_dict2.items():
        writer.writerow([key,value])
```

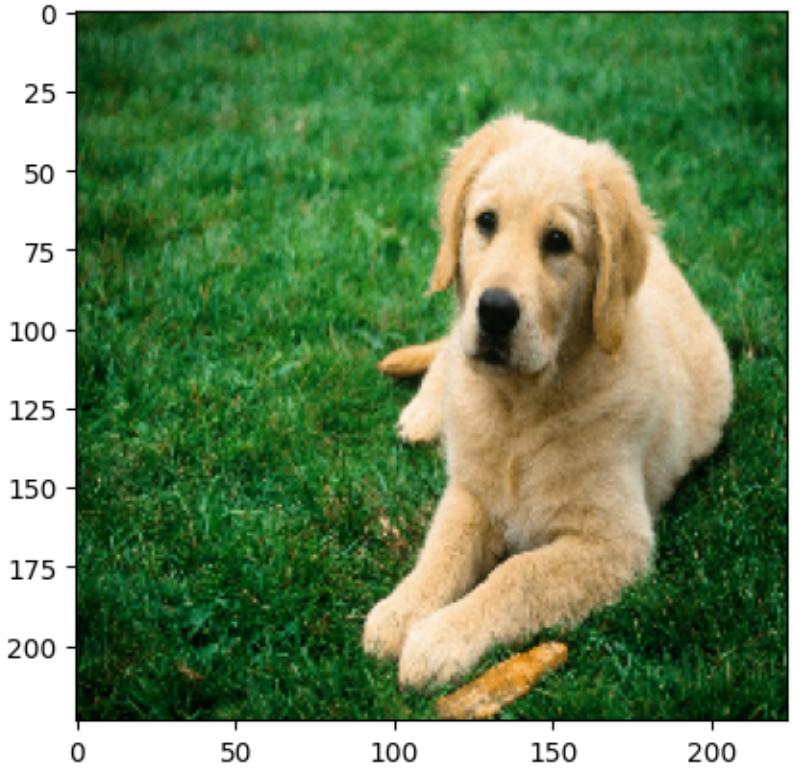
```
Prediction: [('n02123045', 'tabby', 0.64965636)]
```



[18]:
```
## Adjust the image. Convert to numpy array, add batch dimension, and␣
 ↪preprocess.
image_array3 = image.img_to_array(img3)
image_array3 = np.expand_dims(image_array3, axis =0)
image_array3 = preprocess_input(image_array3)
prediction3 = model.predict(image_array3)

## Print the image and prediction here in the Jupyter Notebook.
print("Prediction: {}".format(decode_predictions(prediction3, top = 1)[0]))
plt.imshow(img3)
```

```
plt.show()

## Export the prediction to a prediction file as specified in the document.
resnet50_predictions = resnet_dir.joinpath('assignment06-3_resnet50_predictions.
 ↪csv')

test_dict3 = {'Image 3': str(decode_predictions(prediction3, top = 1)[0])}

with open(resnet50_predictions, 'a') as csv_file:
    writer = csv.writer(csv_file)
    for key, value in test_dict3.items():
        writer.writerow([key,value])
```

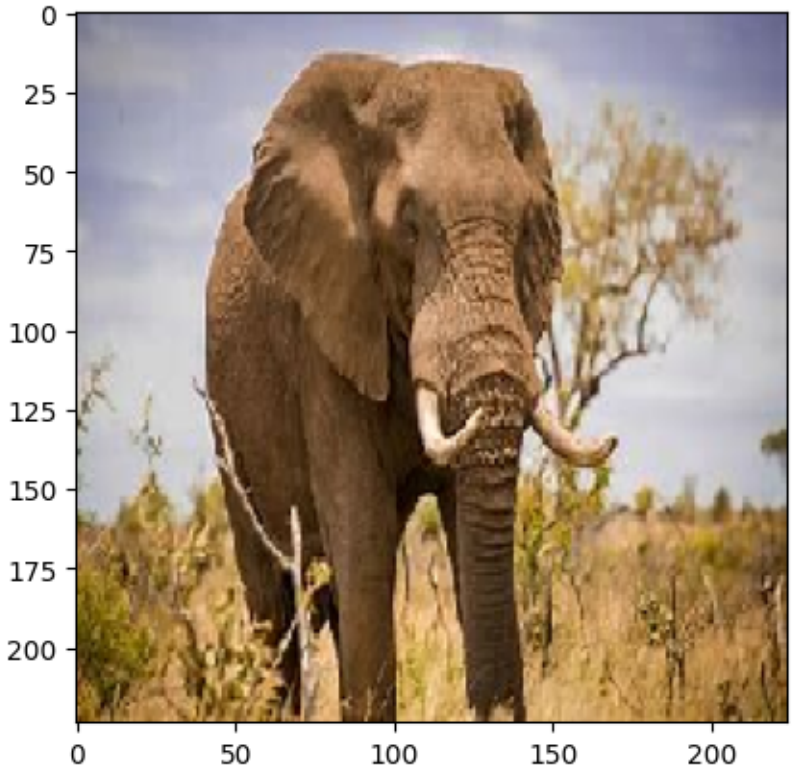
```
Prediction: [('n02099601', 'golden_retriever', 0.9001567)]
```



[19]:
```
## Adjust the image. Convert to numpy array, add batch dimension, and␣
 ↪preprocess.
image_array4 = image.img_to_array(img4)
image_array4 = np.expand_dims(image_array4, axis =0)
image_array4 = preprocess_input(image_array4)
prediction4 = model.predict(image_array4)
```

```
## Print the image and prediction here in the Jupyter Notebook.
print("Prediction: {}".format(decode_predictions(prediction4, top = 1)[0]))
plt.imshow(img4)
plt.show()

## Export the prediction to a prediction file as specified in the document.
resnet50_predictions = resnet_dir.joinpath('assignment06-3_resnet50_predictions.
 ↪csv')

test_dict4 = {'Image 4': str(decode_predictions(prediction4, top = 1)[0])}

with open(resnet50_predictions, 'a') as csv_file:
    writer = csv.writer(csv_file)
    for key, value in test_dict4.items():
        writer.writerow([key,value])
```

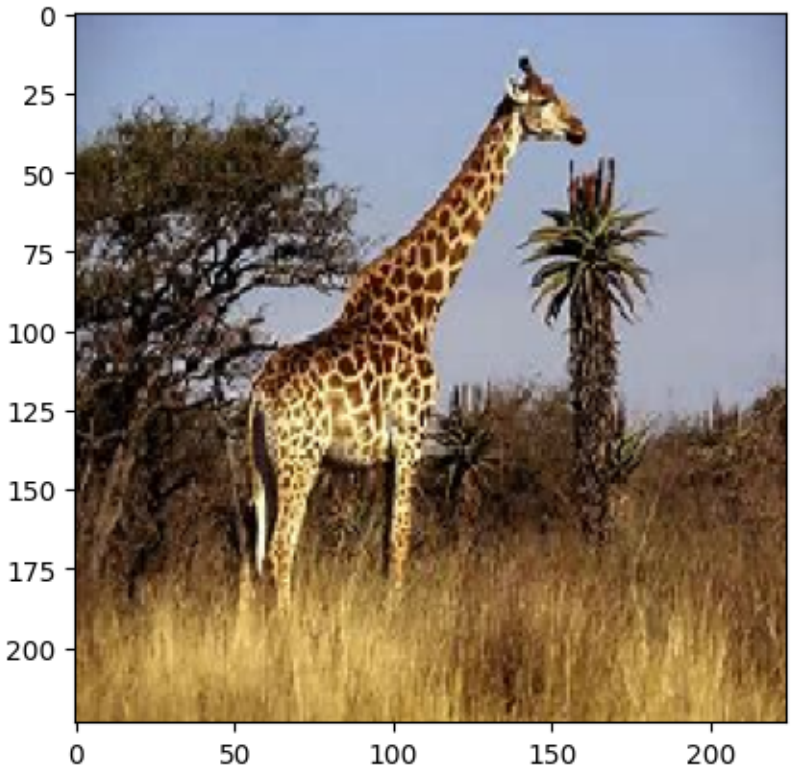
```
Prediction: [('n01871265', 'tusker', 0.6794936)]
```



[20]:
```
## Adjust the image. Convert to numpy array, add batch dimension, and␣
 ↪preprocess.
image_array5 = image.img_to_array(img5)
image_array5 = np.expand_dims(image_array5, axis =0)
```

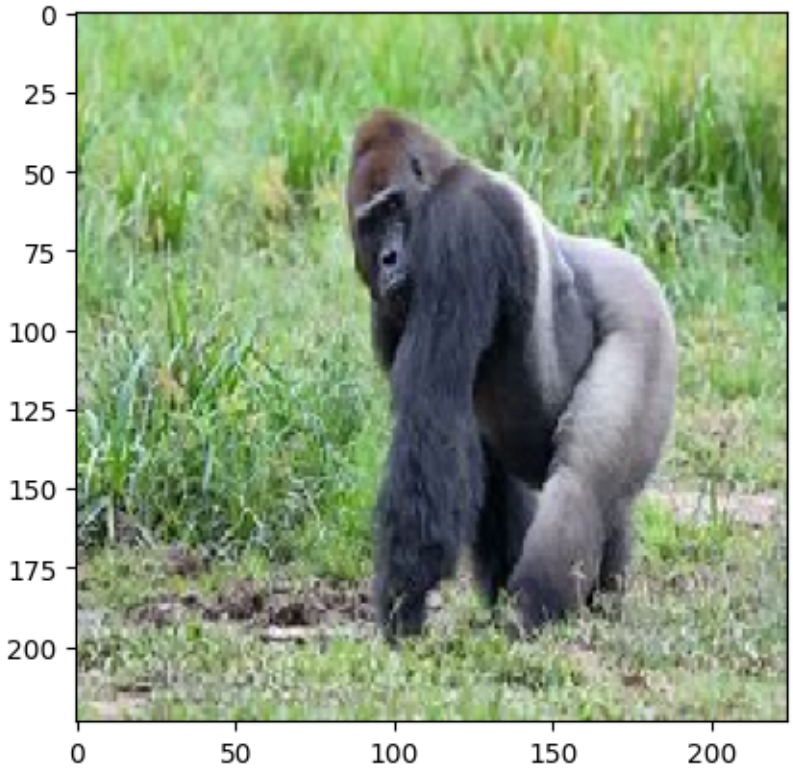
```
image_array5 = preprocess_input(image_array5)
prediction5 = model.predict(image_array5)

## Print the image and prediction here in the Jupyter Notebook.
print("Prediction: {}".format(decode_predictions(prediction5, top = 1)[0]))
plt.imshow(img5)
plt.show()

## Export the prediction to a prediction file as specified in the document.
resnet50_predictions = resnet_dir.joinpath('assignment06-3_resnet50_predictions.
 ↪csv')

test_dict5 = {'Image 5': str(decode_predictions(prediction5, top = 1)[0])}

with open(resnet50_predictions, 'a') as csv_file:
    writer = csv.writer(csv_file)
    for key, value in test_dict5.items():
        writer.writerow([key,value])
```

```
Prediction: [('n02130308', 'cheetah', 0.98081505)]
```



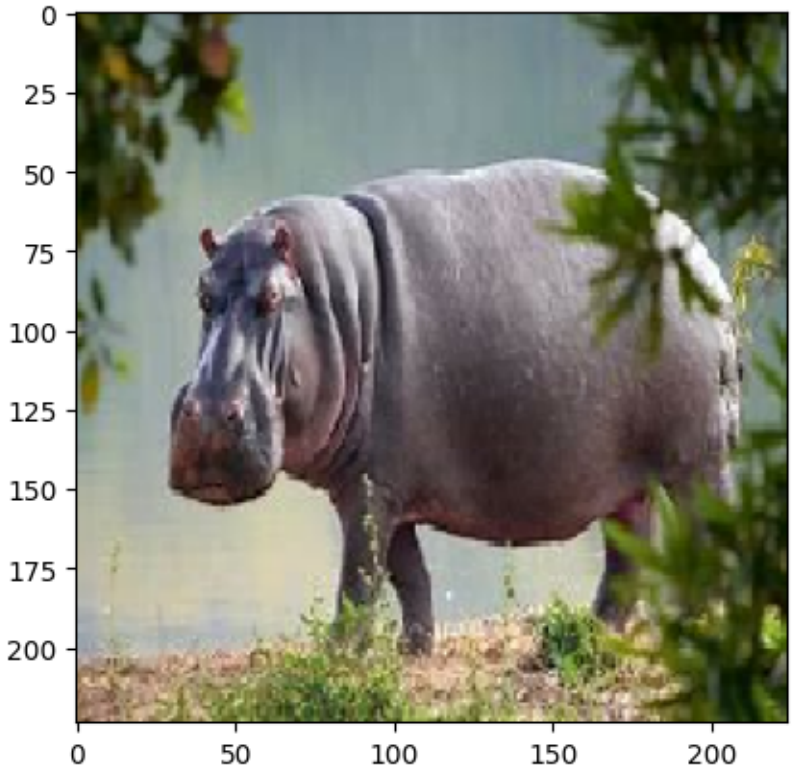
[21]:
```
## Adjust the image. Convert to numpy array, add batch dimension, and␣
 ↪preprocess.
image_array6 = image.img_to_array(img6)
image_array6 = np.expand_dims(image_array6, axis =0)
image_array6 = preprocess_input(image_array6)
prediction6 = model.predict(image_array6)

## Print the image and prediction here in the Jupyter Notebook.
print("Prediction: {}".format(decode_predictions(prediction6, top = 1)[0]))
plt.imshow(img6)
plt.show()

## Export the prediction to a prediction file as specified in the document.
resnet50_predictions = resnet_dir.joinpath('assignment06-3_resnet50_predictions.
 ↪csv')

test_dict6 = {'Image 6': str(decode_predictions(prediction6, top = 1)[0])}

with open(resnet50_predictions, 'a') as csv_file:
    writer = csv.writer(csv_file)
    for key, value in test_dict6.items():
        writer.writerow([key,value])
```

```
Prediction: [('n02480855', 'gorilla', 0.9995832)]
```



[22]:
```
## Adjust the image. Convert to numpy array, add batch dimension, and␣
 ↪preprocess.
image_array7 = image.img_to_array(img7)
image_array7 = np.expand_dims(image_array7, axis =0)
image_array7 = preprocess_input(image_array7)
prediction7 = model.predict(image_array7)

## Print the image and prediction here in the Jupyter Notebook.
print("Prediction: {}".format(decode_predictions(prediction7, top = 1)[0]))
plt.imshow(img7)
plt.show()

## Export the prediction to a prediction file as specified in the document.
resnet50_predictions = resnet_dir.joinpath('assignment06-3_resnet50_predictions.
 ↪csv')

test_dict7 = {'Image 7': str(decode_predictions(prediction7, top = 1)[0])}

with open(resnet50_predictions, 'a') as csv_file:
    writer = csv.writer(csv_file)
    for key, value in test_dict7.items():
        writer.writerow([key,value])
```

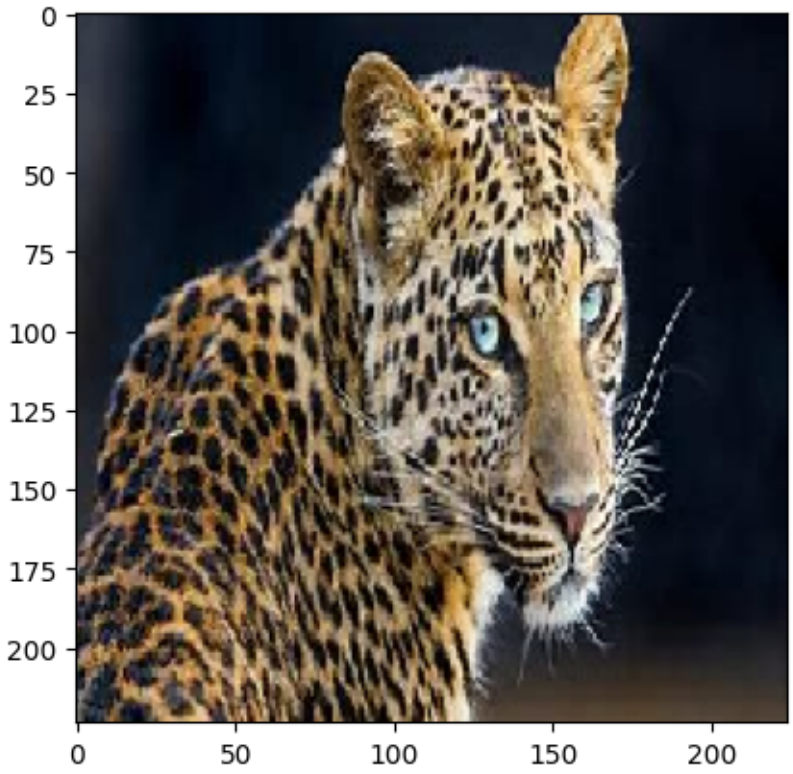
```
Prediction: [('n02398521', 'hippopotamus', 0.9997882)]
```



[23]:
```
## Adjust the image. Convert to numpy array, add batch dimension, and␣
 ↪preprocess.
image_array8 = image.img_to_array(img8)
image_array8 = np.expand_dims(image_array8, axis =0)
image_array8 = preprocess_input(image_array8)
prediction8 = model.predict(image_array8)

## Print the image and prediction here in the Jupyter Notebook.
print("Prediction: {}".format(decode_predictions(prediction8, top = 1)[0]))
plt.imshow(img8)
plt.show()

## Export the prediction to a prediction file as specified in the document.
resnet50_predictions = resnet_dir.joinpath('assignment06-3_resnet50_predictions.
 ↪csv')

test_dict8 = {'Image 8': str(decode_predictions(prediction8, top = 1)[0])}

with open(resnet50_predictions, 'a') as csv_file:
    writer = csv.writer(csv_file)
    for key, value in test_dict8.items():
```

```
        writer.writerow([key,value])
```

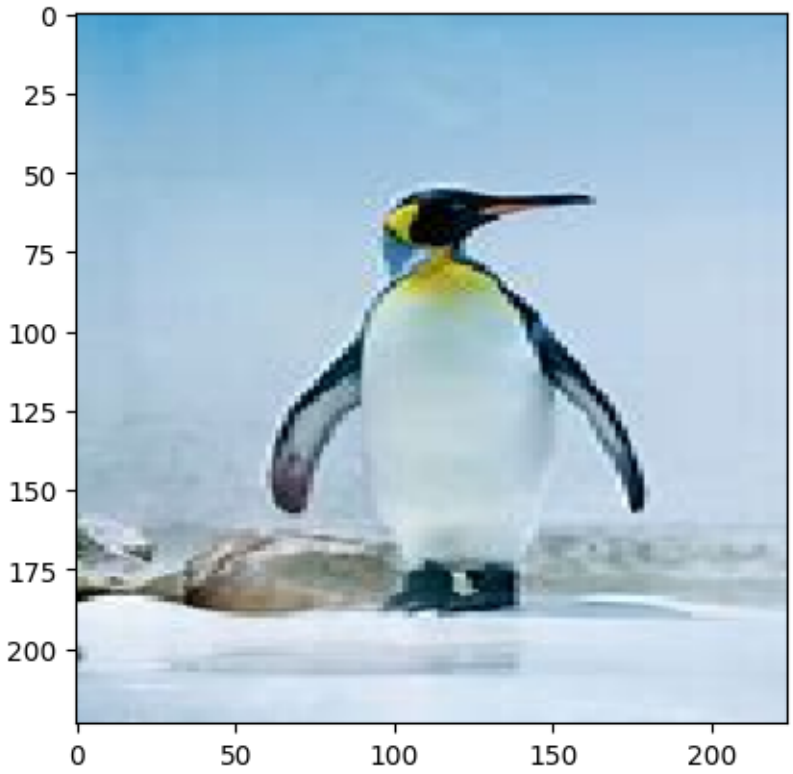
```
Prediction: [('n02128385', 'leopard', 0.91118056)]
```



[24]:
```
## Adjust the image. Convert to numpy array, add batch dimension, and␣
 ↪preprocess.
image_array9 = image.img_to_array(img9)
image_array9 = np.expand_dims(image_array9, axis =0)
image_array9 = preprocess_input(image_array9)
prediction9 = model.predict(image_array9)

## Print the image and prediction here in the Jupyter Notebook.
print("Prediction: {}".format(decode_predictions(prediction9, top = 1)[0]))
plt.imshow(img9)
plt.show()

## Export the prediction to a prediction file as specified in the document.
resnet50_predictions = resnet_dir.joinpath('assignment06-3_resnet50_predictions.
 ↪csv')

test_dict9 = {'Image 9': str(decode_predictions(prediction9, top = 1)[0])}
```

```
with open(resnet50_predictions, 'a') as csv_file:
    writer = csv.writer(csv_file)
    for key, value in test_dict9.items():
        writer.writerow([key,value])
```

```
Prediction: [('n02056570', 'king_penguin', 0.99983263)]
```



[25]:
```
## Adjust the image. Convert to numpy array, add batch dimension, and preprocess.
image_array10 = image.img_to_array(img10)
image_array10 = np.expand_dims(image_array10, axis =0)
image_array10 = preprocess_input(image_array10)
prediction10 = model.predict(image_array10)

## Print the image and prediction here in the Jupyter Notebook.
print("Prediction: {}".format(decode_predictions(prediction10, top = 1)[0]))
plt.imshow(img10)
plt.show()

## Export the prediction to a prediction file as specified in the document.
resnet50_predictions = resnet_dir.joinpath('assignment06-3_resnet50_predictions.csv')
```

```
test_dict10 = {'Image 10': str(decode_predictions(prediction10, top = 1)[0])}

with open(resnet50_predictions, 'a') as csv_file:
    writer = csv.writer(csv_file)
    for key, value in test_dict10.items():
        writer.writerow([key,value])
```

```
Prediction: [('n02129604', 'tiger', 0.85221666)]
```