


```
1 from zipfile import ZipFile
```

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

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).

```
1 !unzip '/content/drive/MyDrive/Animal_Dataset.zip'
```

 Archive: /content/drive/MyDrive/Animal_Dataset.zip

```

inflating: dataset/Testing/bears/k4 (100).jpeg
inflating: dataset/Testing/bears/k4 (100).jpg
inflating: dataset/Testing/bears/k4 (101).jpeg
inflating: dataset/Testing/bears/k4 (101).jpg
inflating: dataset/Testing/bears/k4 (102).jpeg
inflating: dataset/Testing/bears/k4 (102).jpg
inflating: dataset/Testing/bears/k4 (103).jpeg
inflating: dataset/Testing/bears/k4 (104).jpeg
inflating: dataset/Testing/bears/k4 (105).jpeg
inflating: dataset/Testing/bears/k4 (106).jpeg
inflating: dataset/Testing/bears/k4 (107).jpeg
inflating: dataset/Testing/bears/k4 (108).jpeg
inflating: dataset/Testing/bears/k4 (109).jpeg
inflating: dataset/Testing/bears/k4 (110).jpeg
inflating: dataset/Testing/bears/k4 (71).jpg
inflating: dataset/Testing/bears/k4 (72).jpeg
inflating: dataset/Testing/bears/k4 (72).jpg
inflating: dataset/Testing/bears/k4 (73).jpeg
inflating: dataset/Testing/bears/k4 (73).jpg
inflating: dataset/Testing/bears/k4 (74).jpeg
inflating: dataset/Testing/bears/k4 (74).jpg
inflating: dataset/Testing/bears/k4 (75).jpeg
inflating: dataset/Testing/bears/k4 (75).jpg
inflating: dataset/Testing/bears/k4 (76).jpeg
inflating: dataset/Testing/bears/k4 (76).jpg
inflating: dataset/Testing/bears/k4 (77).jpeg
inflating: dataset/Testing/bears/k4 (77).jpg
inflating: dataset/Testing/bears/k4 (78).jpeg
inflating: dataset/Testing/bears/k4 (78).jpg
inflating: dataset/Testing/bears/k4 (79).jpeg
inflating: dataset/Testing/bears/k4 (79).jpg
inflating: dataset/Testing/bears/k4 (80).jpeg
inflating: dataset/Testing/bears/k4 (80).jpg
inflating: dataset/Testing/bears/k4 (81).jpeg
inflating: dataset/Testing/bears/k4 (81).jpg
inflating: dataset/Testing/bears/k4 (82).jpeg
inflating: dataset/Testing/bears/k4 (82).jpg
inflating: dataset/Testing/bears/k4 (83).jpeg
inflating: dataset/Testing/bears/k4 (83).jpg
inflating: dataset/Testing/bears/k4 (84).jpeg
inflating: dataset/Testing/bears/k4 (84).jpg
inflating: dataset/Testing/bears/k4 (85).jpeg
inflating: dataset/Testing/bears/k4 (85).jpg
inflating: dataset/Testing/bears/k4 (86).jpeg
inflating: dataset/Testing/bears/k4 (86).jpg
inflating: dataset/Testing/bears/k4 (87).jpeg
inflating: dataset/Testing/bears/k4 (87).jpg
inflating: dataset/Testing/bears/k4 (88).jpeg
inflating: dataset/Testing/bears/k4 (88).jpg
inflating: dataset/Testing/bears/k4 (89).jpeg
inflating: dataset/Testing/bears/k4 (89).jpg
inflating: dataset/Testing/bears/k4 (90).jpeg
inflating: dataset/Testing/bears/k4 (90).jpg
inflating: dataset/Testing/bears/k4 (91).jpeg
inflating: dataset/Testing/bears/k4 (91).jpg
inflating: dataset/Testing/bears/k4 (92).jpeg
inflating: dataset/Testing/bears/k4 (92).jpg
```

▼ watch the first video "for unzip!" path bold text

```

1
2 # Data Augmentation
3
4 from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

```

1 train_gen = ImageDataGenerator(rescale=(1./255),horizontal_flip=True,shear_range=0.2)
2 test_gen = ImageDataGenerator(rescale=(1./255)) #--> (0 to 255) convert to (0 to 1)

1
2 train = train_gen.flow_from_directory('/content/dataset/Training',
3                                     target_size=(120, 120),
4                                     class_mode='categorical',
5                                     batch_size=8)
6 test = test_gen.flow_from_directory('/content/dataset/Testing',
7                                    target_size=(120, 120),
8                                    class_mode='categorical',
9                                    batch_size=8)
10

Found 1238 images belonging to 4 classes.
Found 326 images belonging to 4 classes.

1 train.class_indices

{'bears': 0, 'crows': 1, 'elephants': 2, 'rats': 3}

1
2 # CNN
3
4 from tensorflow.keras.layers import Convolution2D,MaxPooling2D,Flatten,Dense
5 from tensorflow.keras.models import Sequential

1 model = Sequential()
2 model.add(Convolution2D(20,(3,3),activation='relu',input_shape=(120, 120, 3)))
3 model.add(MaxPooling2D(pool_size=(2,2)))
4 model.add(Flatten())
5 model.add(Dense(45,activation='relu'))
6 model.add(Dense(4,activation='softmax'))

1 model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])

1 model.fit(train,batch_size=8,validation_data=test,epochs=10)
2

Epoch 1/10
155/155 [=====] - 13s 79ms/step - loss: 1.6671 - accuracy: 0.3538 - val_loss: 1.3134 - val_accuracy: 0.3006
Epoch 2/10
155/155 [=====] - 11s 74ms/step - loss: 1.1568 - accuracy: 0.4491 - val_loss: 1.1566 - val_accuracy: 0.4049
Epoch 3/10
155/155 [=====] - 12s 78ms/step - loss: 0.9421 - accuracy: 0.5137 - val_loss: 0.8513 - val_accuracy: 0.5399
Epoch 4/10
155/155 [=====] - 12s 79ms/step - loss: 0.7991 - accuracy: 0.6607 - val_loss: 0.7358 - val_accuracy: 0.7147
Epoch 5/10
155/155 [=====] - 12s 78ms/step - loss: 0.6958 - accuracy: 0.7060 - val_loss: 0.5419 - val_accuracy: 0.7607
Epoch 6/10
155/155 [=====] - 12s 78ms/step - loss: 0.5570 - accuracy: 0.7674 - val_loss: 0.4711 - val_accuracy: 0.7699
Epoch 7/10
155/155 [=====] - 12s 80ms/step - loss: 0.4816 - accuracy: 0.7956 - val_loss: 0.3160 - val_accuracy: 0.8834
Epoch 8/10
155/155 [=====] - 12s 77ms/step - loss: 0.3106 - accuracy: 0.9039 - val_loss: 0.1449 - val_accuracy: 0.9785
Epoch 9/10
155/155 [=====] - 12s 77ms/step - loss: 0.2236 - accuracy: 0.9362 - val_loss: 0.1160 - val_accuracy: 0.9816
Epoch 10/10
155/155 [=====] - 12s 78ms/step - loss: 0.1628 - accuracy: 0.9572 - val_loss: 0.0598 - val_accuracy: 0.9939
<keras.src.callbacks.History at 0x7adf4a881f00>

1
2 model.save('animal.h5')

/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3000: UserWarning: You are saving your model as an HDF5 file via `model.save()`. This API is deprecated. Please use `model.save(filepath, save_format='h5')` instead.

```



```
1 img =img_to_array(img)
2 img

array([[ [255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.],
        ...,
        [252., 247., 244.],
        [251., 246., 243.],
        [251., 246., 243.]],

       [ [255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.],
        ...,
        [252., 247., 244.],
        [251., 246., 243.],
        [251., 246., 243.]],

       [ [255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.],
        ...,
        [252., 247., 244.],
        [251., 246., 243.],
        [251., 246., 243.]],

       ...,

       [ [255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.],
        ...,
        [254., 253., 251.],
        [254., 253., 251.],
        [254., 253., 251.]],

       [ [255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.],
        ...,
        [255., 254., 252.],
        [254., 253., 251.],
        [254., 253., 251.]],

       [ [255., 255., 255.],
        [255., 255., 255.],
        [255., 255., 255.],
        ...,
        [255., 254., 252.],
        [254., 253., 251.],
        [254., 253., 251.]]], dtype=float32)

1 img = np.expand_dims(img,axis=0)
2 img
```

```

array([[[[255., 255., 255.],
         [255., 255., 255.],
         [255., 255., 255.],
         ...,
         [252., 247., 244.],
         [251., 246., 243.],
         [251., 246., 243.]],

        [[255., 255., 255.],
         [255., 255., 255.],
         [255., 255., 255.],
         ...,
         [252., 247., 244.],
         [251., 246., 243.],
         [251., 246., 243.]],

        [[255., 255., 255.],
         [255., 255., 255.],
         [255., 255., 255.],
         ...,
         [252., 247., 244.],
         [251., 246., 243.],
         [251., 246., 243.]],

        ...,

        [[255., 255., 255.],
         [255., 255., 255.],
         [255., 255., 255.],
         ...,
         [254., 253., 251.],
         [254., 253., 251.],
         [254., 253., 251.]],

        [[255., 255., 255.],
         [255., 255., 255.],
         [255., 255., 255.],
         ...,
         [255., 254., 252.],
         [254., 253., 251.],
         [254., 253., 251.]],

        [[255., 255., 255.],
         [255., 255., 255.],
         [255., 255., 255.],
         ...,
         [255., 254., 252.],
         [254., 253., 251.],
         [254., 253., 251.]]]], dtype=float32)

1 np.argmax(model.predict(img))

1/1 [=====] - 0s 96ms/step
1

```