

```
!pip install -q kaggle split-folders
```

```
import os
```

```
os.environ['KAGGLE_USERNAME'] = 'areejm1993'  
os.environ['KAGGLE_KEY'] = 'KGAT_1173feb9968be9f8b0cde46b14b9695c'
```

```
!kaggle datasets list
```

ref	title	size	lastUp
-----	-----	-----	-----
dmahajanbe23/bmw-global-automotive-sales	BMW Global Automotive Sales	55017	2026-0
thuandao/superstore-sales-analytics	SuperStore Sales Analytics	2283450	2026-0
syedaeman2212/airline-ticket-prices-dataset	Airline Ticket Prices Dataset	4409	2026-0
grandmaster07/student-exam-performance-dataset-analysis	Student Exam Performance Dataset Analysis	96178	2026-0
suhanigupta04/student-placement-prediction-dataset	Student Placement Prediction Dataset	1873115	2026-0
kanchana1990/new-york-real-estate-data-2026	New York Real Estate Data 2026	3266120	2026-0
hasanyiitakbulut/ai-and-human-text-dataset	AI and Human Text Dataset	2488505	2026-0
mohamedasak/online-learning-student-dropout-dataset	Online Learning Student Dropout Dataset	95374	2026-0
likithagedipudi/starbucks-customer-ordering-patterns	Starbucks Customer Ordering Patterns	2789356	2026-0
aliiihussain/amazon-sales-dataset	Amazon_Sales_Dataset	1297759	2026-0
alitaqishah/ai-jobs-market-2025-2026-salaries	AI Jobs Market 2025-2026 Salaries	62669	2026-0
thuandao/bank-transactions-dataset-for-fraud-detection	Bank Transactions Dataset for Fraud Detection	1602954	2026-0
mafaqbhhatti/top-100-cities-by-population-2026	Top 100 Cities by Population 2026	2067	2026-0
litonislam/amazon-sales-analysis-cleaned-data	Amazon Sales Analysis- Cleaned Data	1519934	2026-0
shree0910/online-vs-in-store-shopping-behaviour-dataset	Online vs In-Store Shopping Behaviour Dataset	354896	2026-0
algozee/heart-decices	Heart Disease Prediction Using Machine Learning	10070453	2026-0
syedaeman2212/chocolate-sales-dataset-in-2025	Chocolate Sales Dataset in 2025	10105	2026-0
syedaeman2212/apple-prices-dataset	Apple Prices Dataset	9767	2026-0
krupalpatel07/gold-price-dynamics	Gold Price Dynamics	85982	2026-0
algozee/student-productivity-and-behavior-dataset-20k	Student Productivity & Behavior Dataset (20K)	631991	2026-0

```
!kaggle datasets download -d paultimothymooney/breast-histopathology-images
```

Dataset URL: <https://www.kaggle.com/datasets/paultimothymooney/breast-histopathology-images>
License(s): CC0-1.0
Downloading breast-histopathology-images.zip to /content
99% 3.08G/3.10G [00:35<00:00, 106MB/s]
100% 3.10G/3.10G [00:35<00:00, 94.3MB/s]

```
!unzip -q breast-histopathology-images.zip
```

```
import os
```

```
os.listdir('/content')[:20]
```

```
['.config',  
'12822',  
'15514',  
'13021',  
'16895',  
'14155',  
'16085',  
'16531',  
'12626',  
'8951',  
'10306',  
'9381',  
'12934',  
'16167',  
'12821',  
'9250',  
'13025',  
'16896',  
'12823',  
'15471']
```

```
import os
```

```
os.listdir('/content/12822')
```

```
['1', '0']
```

```
import os
```

```
os.listdir('/content/12822/0')[:10]
```

```
['12822_idx5_x2651_y1501_class0.png',  
'12822_idx5_x1101_y1401_class0.png',  
'12822_idx5_x1451_y1151_class0.png',
```

```
'12822_idx5_x1251_y1101_class0.png',
'12822_idx5_x2851_y501_class0.png',
'12822_idx5_x1801_y1651_class0.png',
'12822_idx5_x1151_y951_class0.png',
'12822_idx5_x2351_y1301_class0.png',
'12822_idx5_x2151_y1301_class0.png',
'12822_idx5_x1451_y751_class0.png']
```

```
import os

base_dir = '/content'

count_0 = 0
count_1 = 0

for folder in os.listdir(base_dir):
    folder_path = os.path.join(base_dir, folder)

    if os.path.isdir(folder_path) and folder.isdigit():
        class_0_path = os.path.join(folder_path, '0')
        class_1_path = os.path.join(folder_path, '1')

        if os.path.exists(class_0_path):
            count_0 += len(os.listdir(class_0_path))
        if os.path.exists(class_1_path):
            count_1 += len(os.listdir(class_1_path))

print("Total class 0 images:", count_0)
print("Total class 1 images:", count_1)
```

```
Total class 0 images: 198738
Total class 1 images: 78786
```

```
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator

img_size = (50,50)
batch_size = 32

datagen = ImageDataGenerator(
    rescale=1./255,
    validation_split=0.2
)

train_generator = datagen.flow_from_directory(
    '/content',
    target_size=img_size,
    batch_size=batch_size,
    class_mode='binary',
    subset='training'
)

val_generator = datagen.flow_from_directory(
    '/content',
    target_size=img_size,
    batch_size=batch_size,
    class_mode='binary',
    subset='validation'
)
```

```
Found 666167 images belonging to 283 classes.
Found 166405 images belonging to 283 classes.
```

```
import os
import shutil

base_dir = '/content'
output_dir = '/content/breast_dataset'

os.makedirs(os.path.join(output_dir, '0'), exist_ok=True)
os.makedirs(os.path.join(output_dir, '1'), exist_ok=True)

copied_0 = 0
copied_1 = 0

for folder in os.listdir(base_dir):
    folder_path = os.path.join(base_dir, folder)

    if os.path.isdir(folder_path) and folder.isdigit():
        for class_label in ['0', '1']:
            class_path = os.path.join(folder_path, class_label)
```

```
if os.path.exists(class_path):
    for img in os.listdir(class_path):
        src = os.path.join(class_path, img)
        dst = os.path.join(output_dir, class_label, img)

        if not os.path.exists(dst):
            shutil.copy2(src, dst)
            if class_label == '0':
                copied_0 += 1
            else:
                copied_1 += 1

print("Class 0 copied:", copied_0)
print("Class 1 copied:", copied_1)
print("Dataset organized successfully.")
```

```
Class 0 copied: 0
Class 1 copied: 0
Dataset organized successfully.
```

```
import os

count_0 = len(os.listdir('/content/breast_dataset/0'))
count_1 = len(os.listdir('/content/breast_dataset/1'))

print("Class 0 images in breast_dataset:", count_0)
print("Class 1 images in breast_dataset:", count_1)
```

```
Class 0 images in breast_dataset: 198738
Class 1 images in breast_dataset: 78786
```

```
import splitfolders

splitfolders.ratio(
    "/content/breast_dataset",
    output="/content/dataset",
    seed=42,
    ratio=(0.7, 0.1, 0.2)
)
```

```
Copying files: 277524 files [02:32, 1820.38 files/s]
```

```
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator

img_size = (50, 50)
batch_size = 32

train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=40,
    horizontal_flip=True,
    vertical_flip=True,
    zoom_range=0.2,
    brightness_range=[0.8, 1.2]
)

val_test_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
    '/content/dataset/train',
    target_size=img_size,
    batch_size=batch_size,
    class_mode='binary'
)

val_generator = val_test_datagen.flow_from_directory(
    '/content/dataset/val',
    target_size=img_size,
    batch_size=batch_size,
    class_mode='binary'
)

test_generator = val_test_datagen.flow_from_directory(
    '/content/dataset/test',
    target_size=img_size,
    batch_size=batch_size,
    class_mode='binary',
    shuffle=False
)
```

Found 194266 images belonging to 2 classes.
 Found 27751 images belonging to 2 classes.
 Found 55507 images belonging to 2 classes.

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout

cnn_model = Sequential([
    Conv2D(32, (3,3), activation='relu', input_shape=(50,50,3)),
    MaxPooling2D(2,2),

    Conv2D(64, (3,3), activation='relu'),
    MaxPooling2D(2,2),

    Conv2D(128, (3,3), activation='relu'),
    MaxPooling2D(2,2),

    Flatten(),
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(1, activation='sigmoid')
])

cnn_model.compile(
    optimizer='adam',
    loss='binary_crossentropy',
    metrics=['accuracy']
)

cnn_model.summary()
```

/usr/local/lib/python3.12/dist-packages/keras/src/layers/convolutional/base_conv.py:113: UserWarning: Do not pass an `input`
 super().__init__(activity_regularizer=activity_regularizer, **kwargs)

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 48, 48, 32)	896
max_pooling2d (MaxPooling2D)	(None, 24, 24, 32)	0
conv2d_1 (Conv2D)	(None, 22, 22, 64)	18,496
max_pooling2d_1 (MaxPooling2D)	(None, 11, 11, 64)	0
conv2d_2 (Conv2D)	(None, 9, 9, 128)	73,856
max_pooling2d_2 (MaxPooling2D)	(None, 4, 4, 128)	0
flatten (Flatten)	(None, 2048)	0
dense (Dense)	(None, 128)	262,272
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 1)	129

Total params: 355,649 (1.36 MB)
 Trainable params: 355,649 (1.36 MB)
 Non-trainable params: 0 (0.00 B)

```
history = cnn_model.fit(
    train_generator,
    epochs=5,
    validation_data=val_generator
)
```

/usr/local/lib/python3.12/dist-packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:121: UserWarning: Your `PyData`
 self._warn_if_super_not_called()

```
Epoch 1/5
6071/6071 ————— 423s 69ms/step - accuracy: 0.8067 - loss: 0.4412 - val_accuracy: 0.8258 - val_loss: 0.3918
Epoch 2/5
6071/6071 ————— 323s 53ms/step - accuracy: 0.8443 - loss: 0.3666 - val_accuracy: 0.8554 - val_loss: 0.3327
Epoch 3/5
6071/6071 ————— 316s 52ms/step - accuracy: 0.8531 - loss: 0.3487 - val_accuracy: 0.8629 - val_loss: 0.3167
Epoch 4/5
6071/6071 ————— 317s 52ms/step - accuracy: 0.8570 - loss: 0.3409 - val_accuracy: 0.8591 - val_loss: 0.3257
Epoch 5/5
6071/6071 ————— 313s 52ms/step - accuracy: 0.8563 - loss: 0.3406 - val_accuracy: 0.8566 - val_loss: 0.3326
```

```
test_loss, test_accuracy = cnn_model.evaluate(test_generator)
```

```
print("Test Accuracy:", test_accuracy)
print("Test Loss:", test_loss)
```

1735/1735 ————— 47s 27ms/step - accuracy: 0.9246 - loss: 0.1885
Test Accuracy: 0.8536761403083801
Test Loss: 0.34146490693092346

```
import numpy as np
from sklearn.metrics import confusion_matrix, classification_report
import seaborn as sns
import matplotlib.pyplot as plt

# Predictions
y_pred_probs = cnn_model.predict(test_generator)
y_pred = (y_pred_probs > 0.5).astype(int).reshape(-1)

# True labels
y_true = test_generator.classes

# Confusion Matrix
cm = confusion_matrix(y_true, y_pred)

plt.figure(figsize=(6,5))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()

# Classification Report
print(classification_report(y_true, y_pred))
```

1735/1735 ————— 29s 17ms/step



