BRAIN TUMOR DETECTION

In [1]:
```python
import numpy as np
import pandas as pd
import PIL
import tensorflow as tf
import keras
import cv2
import gradio as gr
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```python
data_set="C:\\Users\\acer\\Desktop\\sponsorship letter\\Datasets"
```

In [3]:
```python
import pathlib
```

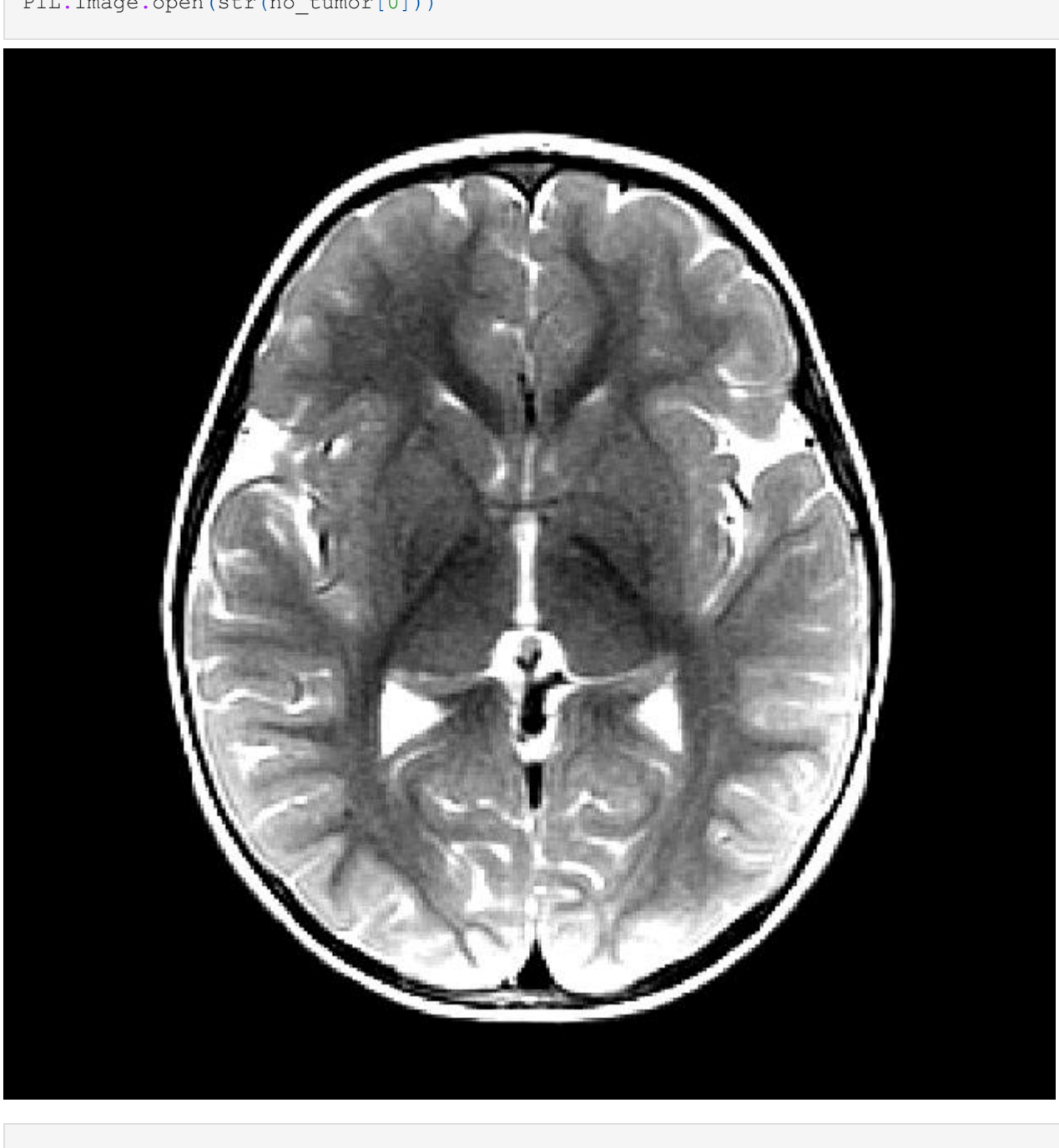
In [4]:
```python
data_dir = pathlib.Path(data_set)
image_count = len(list(data_dir.glob('*/*.jpg')))
print(image_count)
```

```
3000
```

In [5]:
```python
no_tumor=list(data_dir.glob('no/*'))
```

In [6]:
```python
PIL.Image.open(str(no_tumor[0]))
```

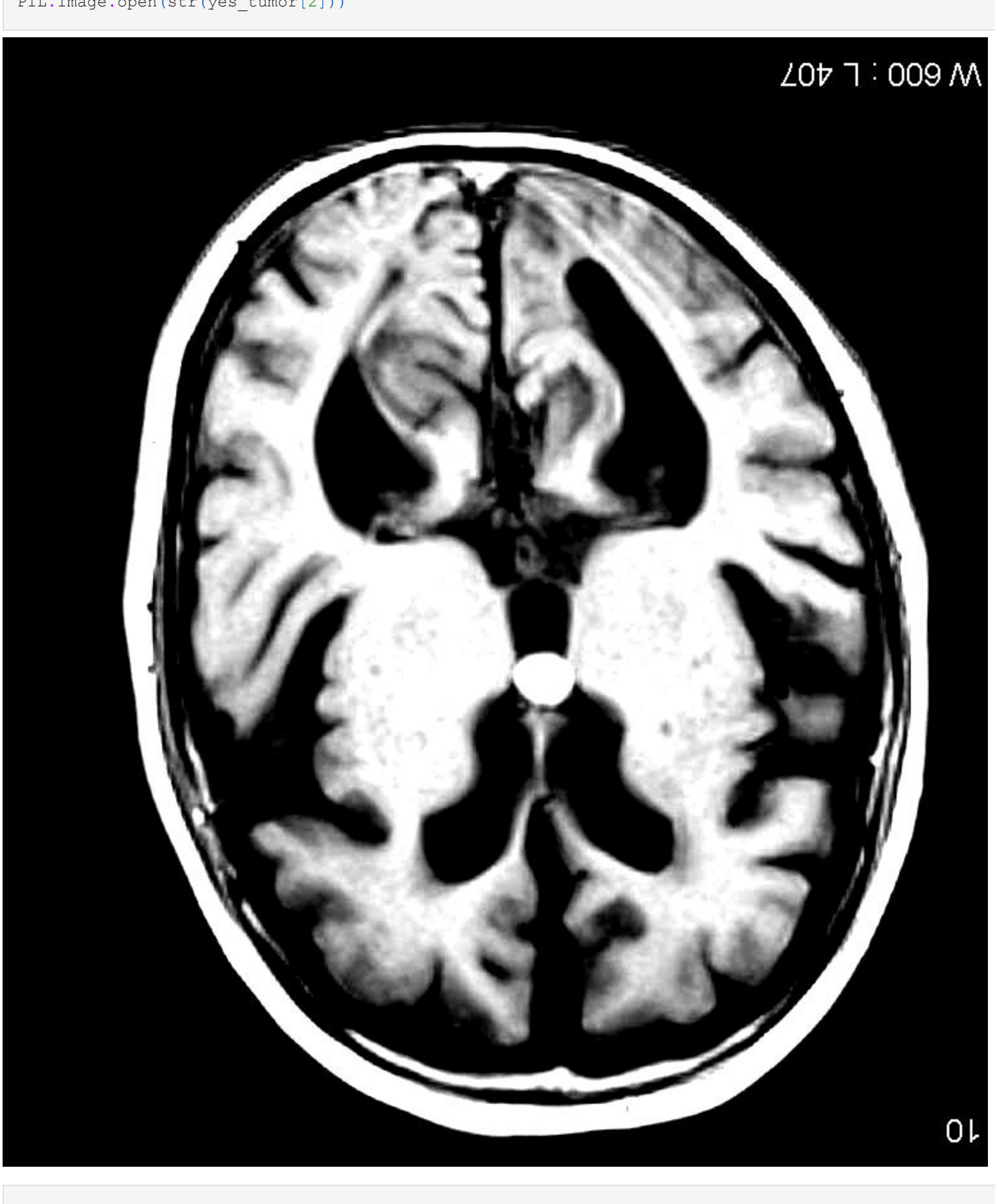
Out[6]:



In [7]:
```python
yes_tumor=list(data_dir.glob('yes/*'))
```

In [8]:
```python
PIL.Image.open(str(yes_tumor[2]))
```

Out[8]:



In [9]:
```python
cv2.imread(str(yes_tumor[5])).shape
```

Out[9]:
```
(303, 223, 3)
```

In [10]:
```python
image_dict={
    "no":list(data_dir.glob('no/*')),
    "yes":list(data_dir.glob('yes/*'))
}
```

In [11]:
```python
image_label={
    "no":0,
    "yes":1
}
```

In [12]:
```python
X,y=[],[]
for tumor,images in image_dict.items():
    for image in images:
        image=cv2.imread(str(image))
        image_resize=cv2.resize(image,(180,180))
        X.append(image_resize)
        y.append(image_label[tumor])
```

In [13]:
```python
X=np.array(X)
y=np.array(y)
X
```

Out[13]:
```
array([[[[ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0],
         ...,
         [ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0]],

        [[ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0],
         ...,
         [ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0]],

        [[ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0],
         ...,
         [ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0]],

        ...,

        [[ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0],
         ...,
         [ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0]],

        [[ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0],
         ...,
         [ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0]],

        [[ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0],
         ...,
         [ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0]]],


       [[[ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0],
         ...,
         [ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0]],

        [[ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0],
         ...,
         [ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0]],

        [[ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0],
         ...,
         [ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0]],

        ...,

        [[ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0],
         ...,
         [ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0]],

        [[ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0],
         ...,
         [ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0]],

        [[ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0],
         ...,
         [ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0]]],


       [[[16, 16, 16],
         [16, 16, 16],
         [16, 16, 16],
         ...,
         [16, 16, 16],
         [16, 16, 16],
         [16, 16, 16]],

        [[16, 16, 16],
         [16, 16, 16],
         [16, 16, 16],
         ...,
         [16, 16, 16],
         [16, 16, 16],
         [16, 16, 16]],

        [[16, 16, 16],
         [16, 16, 16],
         [16, 16, 16],
         ...,
         [16, 16, 16],
         [16, 16, 16],
         [16, 16, 16]],

        ...,

        [[16, 16, 16],
         [16, 16, 16],
         [16, 16, 16],
         ...,
         [16, 16, 16],
         [16, 16, 16],
         [16, 16, 16]],

        [[16, 16, 16],
         [16, 16, 16],
         [16, 16, 16],
         ...,
         [16, 16, 16],
         [16, 16, 16],
         [16, 16, 16]],

        [[16, 16, 16],
         [16, 16, 16],
         [16, 16, 16],
         ...,
         [16, 16, 16],
         [16, 16, 16],
         [16, 16, 16]]],


       ...,


       [[[ 0,  0,  0],
         [ 0,  0,  0],
         [ 1,  1,  1],
         ...,
         [ 1,  1,  1],
         [ 1,  1,  1],
         [ 1,  1,  1]],

        [[ 1,  1,  1],
         [ 1,  1,  1],
         [ 2,  2,  2],
         ...,
         [ 1,  1,  1],
         [ 1,  1,  1],
         [ 1,  1,  1]],

        [[ 3,  3,  3],
         [ 2,  2,  2],
         [ 2,  2,  2],
         ...,
         [ 2,  2,  2],
         [ 2,  2,  2],
         [ 2,  2,  2]],

        ...,

        [[ 1,  1,  1],
         [ 1,  1,  1],
         [ 1,  1,  1],
         ...,
         [ 2,  2,  2],
         [ 2,  2,  2],
         [ 2,  2,  2]],

        [[ 2,  2,  2],
         [ 2,  2,  2],
         [ 2,  2,  2],
         ...,
         [ 1,  1,  1],
         [ 1,  1,  1],
         [ 1,  1,  1]],

        [[ 2,  2,  2],
         [ 2,  2,  2],
         [ 2,  2,  2],
         ...,
         [ 1,  1,  1],
         [ 1,  1,  1],
         [ 1,  1,  1]]],


       [[[ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0],
         ...,
         [ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0]],

        [[ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0],
         ...,
         [ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0]],

        [[ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0],
         ...,
         [ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0]],

        ...,

        [[ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0],
         ...,
         [ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0]],

        [[ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0],
         ...,
         [ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0]],

        [[ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0],
         ...,
         [ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0]]],


       [[[ 4,  4,  4],
         [ 4,  4,  4],
         [ 4,  4,  4],
         ...,
         [ 1,  1,  1],
         [ 0,  0,  0],
         [ 0,  0,  0]],

        [[ 4,  4,  4],
         [ 4,  4,  4],
         [ 4,  4,  4],
         ...,
         [ 1,  1,  1],
         [ 0,  0,  0],
         [ 0,  0,  0]],

        [[ 4,  4,  4],
         [ 4,  4,  4],
         [ 4,  4,  4],
         ...,
         [ 1,  1,  1],
         [ 0,  0,  0],
         [ 0,  0,  0]],

        ...,

        [[ 4,  4,  4],
         [ 4,  4,  4],
         [ 4,  4,  4],
         ...,
         [ 0,  0,  0],
         [ 0,  0,  0],
         [ 1,  1,  1]],

        [[ 4,  4,  4],
         [ 4,  4,  4],
         [ 4,  4,  4],
         ...,
         [ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0]],

        [[ 4,  4,  4],
         [ 4,  4,  4],
         [ 4,  4,  4],
         ...,
         [ 0,  0,  0],
         [ 0,  0,  0],
         [ 0,  0,  0]]]], dtype=uint8)
```

In [14]:
```python
#  X_scale=X/255
```

In [15]:
```python
brain_class=["Tumor Not Detected","Tumor Detected"]
```

In [16]:
```python
import sklearn
```

In [17]:
```python
from sklearn.model_selection import train_test_split
```

In [18]:
```python
X=X.astype('float32')
```

In [19]:
```python
X_train,X_test,y_train,y_test=train_test_split(X,y,random_state=0)
```

In [20]:
```python
data_agumentation=keras.Sequential([
    tf.keras.layers.experimental.preprocessing.RandomFlip("horizontal",input_shape=(180,180,3)),
    tf.keras.layers.experimental.preprocessing.RandomZoom(0.1),
    tf.keras.layers.experimental.preprocessing.RandomRotation(0.1),
#     tf.keras.layers.experimental.preprocessing.RandomFlip(,input_shape=(180,180,3))
])
```

Training Of Model

In [23]:
```python
model=keras.Sequential([
#     CNN
    data_agumentation,
    keras.layers.Conv2D(16,3,input_shape=(180,180,3),activation="relu"),
    keras.layers.MaxPooling2D(),
    keras.layers.Conv2D(32,3,activation="relu"),
    keras.layers.MaxPooling2D(),
    keras.layers.Conv2D(64,3,activation="relu"),
    keras.layers.MaxPooling2D(),
    keras.layers.Flatten(),
#     layers
    keras.layers.Dense(128,activation="relu"),
    keras.layers.Dense(2,activation="softmax")
])
model.compile(optimizer="adam",loss="sparse_categorical_crossentropy",metrics=['accuracy'])
model.fit(X_train,y_train,epochs=150)
```

```
Epoch 1/150
71/71 [==============================] - 5s 35ms/step - loss: 84.6293 - accuracy: 0.5915
Epoch 2/150
71/71 [==============================] - 2s 35ms/step - loss: 0.5735 - accuracy: 0.7214
Epoch 3/150
71/71 [==============================] - 3s 37ms/step - loss: 0.5362 - accuracy: 0.7424
Epoch 4/150
71/71 [==============================] - 3s 35ms/step - loss: 0.4895 - accuracy: 0.7643
Epoch 5/150
71/71 [==============================] - 2s 35ms/step - loss: 0.4677 - accuracy: 0.7816
Epoch 6/150
71/71 [==============================] - 3s 36ms/step - loss: 0.5004 - accuracy: 0.7518
Epoch 7/150
71/71 [==============================] - 3s 36ms/step - loss: 0.4515 - accuracy: 0.7775
Epoch 8/150
71/71 [==============================] - 3s 36ms/step - loss: 0.4751 - accuracy: 0.7772
Epoch 9/150
71/71 [==============================] - 3s 35ms/step - loss: 0.4541 - accuracy: 0.7877
Epoch 10/150
71/71 [==============================] - 2s 35ms/step - loss: 0.4565 - accuracy: 0.7777
Epoch 11/150
71/71 [==============================] - 3s 35ms/step - loss: 0.4076 - accuracy: 0.8190
Epoch 12/150
71/71 [==============================] - 2s 35ms/step - loss: 0.4169 - accuracy: 0.8182
Epoch 13/150
71/71 [==============================] - 3s 35ms/step - loss: 0.3785 - accuracy: 0.8262
Epoch 14/150
71/71 [==============================] - 2s 35ms/step - loss: 0.3507 - accuracy: 0.8528
Epoch 15/150
71/71 [==============================] - 2s 35ms/step - loss: 0.3630 - accuracy: 0.8468
Epoch 16/150
71/71 [==============================] - 2s 35ms/step - loss: 0.3817 - accuracy: 0.8303
Epoch 17/150
71/71 [==============================] - 2s 35ms/step - loss: 0.3762 - accuracy: 0.8380
Epoch 18/150
71/71 [==============================] - 2s 35ms/step - loss: 0.3513 - accuracy: 0.8601
Epoch 19/150
71/71 [==============================] - 3s 36ms/step - loss: 0.3641 - accuracy: 0.8400
Epoch 20/150
71/71 [==============================] - 3s 37ms/step - loss: 0.3119 - accuracy: 0.8684
Epoch 21/150
71/71 [==============================] - 3s 36ms/step - loss: 0.3407 - accuracy: 0.8626
Epoch 22/150
71/71 [==============================] - 3s 36ms/step - loss: 0.3055 - accuracy: 0.8769
Epoch 23/150
71/71 [==============================] - 3s 39ms/step - loss: 0.2515 - accuracy: 0.8942
Epoch 24/150
71/71 [==============================] - 3s 36ms/step - loss: 0.2869 - accuracy: 0.8877
Epoch 25/150
71/71 [==============================] - 3s 35ms/step - loss: 0.2368 - accuracy: 0.9073
Epoch 26/150
71/71 [==============================] - 3s 38ms/step - loss: 0.2346 - accuracy: 0.9075
Epoch 27/150
71/71 [==============================] - 3s 36ms/step - loss: 0.2604 - accuracy: 0.8973
Epoch 28/150
71/71 [==============================] - 3s 36ms/step - loss: 0.2201 - accuracy: 0.9141
Epoch 29/150
71/71 [==============================] - 3s 35ms/step - loss: 0.2186 - accuracy: 0.9183
Epoch 30/150
71/71 [==============================] - 3s 35ms/step - loss: 0.2170 - accuracy: 0.9114
Epoch 31/150
71/71 [==============================] - 3s 36ms/step - loss: 0.2047 - accuracy: 0.9079
Epoch 32/150
71/71 [==============================] - 3s 36ms/step - loss: 0.1962 - accuracy: 0.9266
Epoch 33/150
71/71 [==============================] - 3s 36ms/step - loss: 0.2064 - accuracy: 0.9159
Epoch 34/150
71/71 [==============================] - 3s 36ms/step - loss: 0.2040 - accuracy: 0.9290
Epoch 35/150
71/71 [==============================] - 3s 36ms/step - loss: 0.2161 - accuracy: 0.9183
Epoch 36/150
71/71 [==============================] - 3s 36ms/step - loss: 0.1831 - accuracy: 0.9345
Epoch 37/150
71/71 [==============================] - 3s 38ms/step - loss: 0.1782 - accuracy: 0.9295
Epoch 38/150
71/71 [==============================] - 3s 38ms/step - loss: 0.1847 - accuracy: 0.9331
Epoch 39/150
71/71 [==============================] - 3s 36ms/step - loss: 0.1880 - accuracy: 0.9232
Epoch 40/150
71/71 [==============================] - 3s 36ms/step - loss: 0.1909 - accuracy: 0.9243
Epoch 41/150
71/71 [==============================] - 3s 36ms/step - loss: 0.1517 - accuracy: 0.9410
Epoch 42/150
71/71 [==============================] - 3s 41ms/step - loss: 0.1859 - accuracy: 0.9338
Epoch 43/150
71/71 [==============================] - 3s 39ms/step - loss: 0.1593 - accuracy: 0.9382
Epoch 44/150
71/71 [==============================] - 3s 36ms/step - loss: 0.1548 - accuracy: 0.9449
Epoch 45/150
71/71 [==============================] - 3s 36ms/step - loss: 0.1291 - accuracy: 0.9453
Epoch 46/150
71/71 [==============================] - 2s 35ms/step - loss: 0.1335 - accuracy: 0.9573
Epoch 47/150
71/71 [==============================] - 2s 35ms/step - loss: 0.1352 - accuracy: 0.9571
Epoch 48/150
71/71 [==============================] - 2s 35ms/step - loss: 0.1595 - accuracy: 0.9438
Epoch 49/150
71/71 [==============================] - 2s 35ms/step - loss: 0.1545 - accuracy: 0.9392
Epoch 50/150
71/71 [==============================] - 2s 34ms/step - loss: 0.1760 - accuracy: 0.9319
Epoch 51/150
71/71 [==============================] - 2s 35ms/step - loss: 0.1478 - accuracy: 0.9458
Epoch 52/150
71/71 [==============================] - 2s 35ms/step - loss: 0.0953 - accuracy: 0.9640
Epoch 53/150
71/71 [==============================] - 2s 35ms/step - loss: 0.1283 - accuracy: 0.9552
Epoch 54/150
71/71 [==============================] - 2s 35ms/step - loss: 0.1384 - accuracy: 0.9453
Epoch 55/150
71/71 [==============================] - 2s 35ms/step - loss: 0.1508 - accuracy: 0.9466
Epoch 56/150
71/71 [==============================] - 2s 35ms/step - loss: 0.1286 - accuracy: 0.9502
Epoch 57/150
71/71 [==============================] - 2s 35ms/step - loss: 0.1260 - accuracy: 0.9583 0s - loss: 0.1292 -
Epoch 58/150
71/71 [==============================] - 2s 35ms/step - loss: 0.1302 - accuracy: 0.9514
Epoch 59/150
71/71 [==============================] - 3s 38ms/step - loss: 0.1066 - accuracy: 0.9603
Epoch 60/150
71/71 [==============================] - 3s 36ms/step - loss: 0.1011 - accuracy: 0.9630
Epoch 61/150
71/71 [==============================] - 3s 39ms/step - loss: 0.1059 - accuracy: 0.9613
Epoch 62/150
71/71 [==============================] - 3s 37ms/step - loss: 0.1297 - accuracy: 0.9497
Epoch 63/150
71/71 [==============================] - 3s 37ms/step - loss: 0.1025 - accuracy: 0.9618
Epoch 64/150
71/71 [==============================] - 3s 41ms/step - loss: 0.1184 - accuracy: 0.9558
Epoch 65/150
71/71 [==============================] - 3s 38ms/step - loss: 0.1021 - accuracy: 0.9628
Epoch 66/150
71/71 [==============================] - 3s 36ms/step - loss: 0.0815 - accuracy: 0.9711
Epoch 67/150
71/71 [==============================] - 3s 44ms/step - loss: 0.1174 - accuracy: 0.9567
Epoch 68/150
71/71 [==============================] - 3s 36ms/step - loss: 0.1290 - accuracy: 0.9512
Epoch 69/150
71/71 [==============================] - 3s 36ms/step - loss: 0.0810 - accuracy: 0.9715
Epoch 70/150
71/71 [==============================] - 3s 36ms/step - loss: 0.0804 - accuracy: 0.9705ETA: 0s - los
Epoch 71/150
71/71 [==============================] - 3s 36ms/step - loss: 0.0916 - accuracy: 0.9673
Epoch 72/150
71/71 [==============================] - 3s 40ms/step - loss: 0.0933 - accuracy: 0.9675
Epoch 73/150
71/71 [==============================] - 3s 38ms/step - loss: 0.0740 - accuracy: 0.9706
Epoch 74/150
71/71 [==============================] - 3s 39ms/step - loss: 0.1110 - accuracy: 0.9636
Epoch 75/150
71/71 [==============================] - 3s 37ms/step - loss: 0.0651 - accuracy: 0.9774
Epoch 76/150
71/71 [==============================] - 3s 41ms/step - loss: 0.0704 - accuracy: 0.9688
Epoch 77/150
71/71 [==============================] - 3s 44ms/step - loss: 0.0816 - accuracy: 0.9685
Epoch 78/150
71/71 [==============================] - 2s 34ms/step - loss: 0.0902 - accuracy: 0.9703
Epoch 79/150
71/71 [==============================] - 3s 42ms/step - loss: 0.0721 - accuracy: 0.9759
Epoch 80/150
71/71 [==============================] - 3s 38ms/step - loss: 0.1015 - accuracy: 0.9682
Epoch 81/150
71/71 [==============================] - 3s 38ms/step - loss: 0.0825 - accuracy: 0.9742
Epoch 82/150
71/71 [==============================] - 3s 38ms/step - loss: 0.0984 - accuracy: 0.9703 1s -
Epoch 83/150
71/71 [==============================] - 3s 40ms/step - loss: 0.0867 - accuracy: 0.9667
Epoch 84/150
71/71 [==============================] - 3s 37ms/step - loss: 0.0528 - accuracy: 0.9811 0s - loss: 0.0523 - acc
uracy: 0.
Epoch 85/150
71/71 [==============================] - 3s 39ms/step - loss: 0.0688 - accuracy: 0.9702
Epoch 86/150
71/71 [==============================] - 3s 37ms/step - loss: 0.1006 - accuracy: 0.9661
Epoch 87/150
71/71 [==============================] - 2s 35ms/step - loss: 0.0476 - accuracy: 0.9794
Epoch 88/150
71/71 [==============================] - 3s 36ms/step - loss: 0.0582 - accuracy: 0.9846
Epoch 89/150
71/71 [==============================] - 2s 35ms/step - loss: 0.0532 - accuracy: 0.9806
Epoch 90/150
71/71 [==============================] - 2s 35ms/step - loss: 0.0854 - accuracy: 0.9741
Epoch 91/150
71/71 [==============================] - 2s 34ms/step - loss: 0.0548 - accuracy: 0.9811
Epoch 92/150
71/71 [==============================] - 3s 36ms/step - loss: 0.0738 - accuracy: 0.9787
Epoch 93/150
71/71 [==============================] - 3s 38ms/step - loss: 0.0467 - accuracy: 0.9824
Epoch 94/150
71/71 [==============================] - 2s 35ms/step - loss: 0.0681 - accuracy: 0.9745 1s - loss: 0.0602 -  -
ETA: 0s - loss: 0.0628 - ac - ETA: 0s - loss: 0.0664 - accuracy
Epoch 95/150
71/71 [==============================] - 3s 39ms/step - loss: 0.0719 - accuracy: 0.9727
Epoch 96/150
71/71 [==============================] - 3s 35ms/step - loss: 0.0918 - accuracy: 0.9694
Epoch 97/150
71/71 [==============================] - 3s 37ms/step - loss: 0.0698 - accuracy: 0.9765
Epoch 98/150
71/71 [==============================] - 3s 36ms/step - loss: 0.0606 - accuracy: 0.9756
Epoch 99/150
71/71 [==============================] - 3s 40ms/step - loss: 0.0808 - accuracy: 0.9696
Epoch 100/150
71/71 [==============================] - 3s 37ms/step - loss: 0.0566 - accuracy: 0.9855
Epoch 101/150
71/71 [==============================] - 3s 37ms/step - loss: 0.0622 - accuracy: 0.9771
Epoch 102/150
71/71 [==============================] - 3s 35ms/step - loss: 0.1014 - accuracy: 0.9651
Epoch 103/150
71/71 [==============================] - 2s 35ms/step - loss: 0.0665 - accuracy: 0.9753
Epoch 104/150
71/71 [==============================] - 2s 35ms/step - loss: 0.0657 - accuracy: 0.9712
Epoch 105/150
71/71 [==============================] - 3s 37ms/step - loss: 0.0404 - accuracy: 0.9821
Epoch 106/150
71/71 [==============================] - 3s 38ms/step - loss: 0.0493 - accuracy: 0.9801
Epoch 107/150
71/71 [==============================] - 3s 38ms/step - loss: 0.0659 - accuracy: 0.9810
Epoch 108/150
71/71 [==============================] - 2s 35ms/step - loss: 0.0480 - accuracy: 0.9826
Epoch 109/150
71/71 [==============================] - 3s 35ms/step - loss: 0.0535 - accuracy: 0.9776
Epoch 110/150
71/71 [==============================] - 3s 37ms/step - loss: 0.0798 - accuracy: 0.9704 1s -
Epoch 111/150
71/71 [==============================] - 3s 36ms/step - loss: 0.0635 - accuracy: 0.9715
Epoch 112/150
71/71 [==============================] - 3s 38ms/step - loss: 0.0618 - accuracy: 0.9798
Epoch 113/150
71/71 [==============================] - 3s 37ms/step - loss: 0.0661 - accuracy: 0.9832
Epoch 114/150
71/71 [==============================] - 3s 39ms/step - loss: 0.0819 - accuracy: 0.9696
Epoch 115/150
71/71 [==============================] - 3s 38ms/step - loss: 0.0725 - accuracy: 0.9757
Epoch 116/150
71/71 [==============================] - 2s 35ms/step - loss: 0.0588 - accuracy: 0.9820
Epoch 117/150
71/71 [==============================] - 2s 34ms/step - loss: 0.0568 - accuracy: 0.9796
Epoch 118/150
71/71 [==============================] - 3s 35ms/step - loss: 0.0320 - accuracy: 0.9914
Epoch 119/150
71/71 [==============================] - 3s 36ms/step - loss: 0.0627 - accuracy: 0.9813
Epoch 120/150
71/71 [==============================] - 3s 39ms/step - loss: 0.0530 - accuracy: 0.9821
Epoch 121/150
71/71 [==============================] - 3s 39ms/step - loss: 0.0536 - accuracy: 0.9783
Epoch 122/150
71/71 [==============================] - 3s 35ms/step - loss: 0.0718 - accuracy: 0.9745
Epoch 123/150
71/71 [==============================] - 2s 34ms/step - loss: 0.0485 - accuracy: 0.9830
Epoch 124/150
71/71 [==============================] - 2s 34ms/step - loss: 0.0534 - accuracy: 0.9807
Epoch 125/150
71/71 [==============================] - 3s 36ms/step - loss: 0.0441 - accuracy: 0.9866
Epoch 126/150
71/71 [==============================] - 3s 35ms/step - loss: 0.0463 - accuracy: 0.9841
Epoch 127/150
71/71 [==============================] - 3s 36ms/step - loss: 0.0792 - accuracy: 0.9777
Epoch 128/150
71/71 [==============================] - 3s 38ms/step - loss: 0.0828 - accuracy: 0.9724
Epoch 129/150
71/71 [==============================] - 2s 34ms/step - loss: 0.0809 - accuracy: 0.9769
Epoch 130/150
71/71 [==============================] - 3s 36ms/step - loss: 0.0549 - accuracy: 0.9823
Epoch 131/150
71/71 [==============================] - 3s 38ms/step - loss: 0.0417 - accuracy: 0.9852
Epoch 132/150
71/71 [==============================] - 2s 35ms/step - loss: 0.0386 - accuracy: 0.9859
Epoch 133/150
71/71 [==============================] - 3s 36ms/step - loss: 0.0478 - accuracy: 0.9855
Epoch 134/150
71/71 [==============================] - 3s 35ms/step - loss: 0.0503 - accuracy: 0.9815 0s - loss: 0.0501 - acc
ura
Epoch 135/150
71/71 [==============================] - 3s 37ms/step - loss: 0.0276 - accuracy: 0.9910
Epoch 136/150
71/71 [==============================] - 3s 38ms/step - loss: 0.0544 - accuracy: 0.9796
Epoch 137/150
71/71 [==============================] - 3s 35ms/step - loss: 0.0457 - accuracy: 0.9840
Epoch 138/150
71/71 [==============================] - 2s 35ms/step - loss: 0.0503 - accuracy: 0.9839
Epoch 139/150
71/71 [==============================] - 2s 35ms/step - loss: 0.0455 - accuracy: 0.9868
Epoch 140/150
71/71 [==============================] - 2s 35ms/step - loss: 0.1983 - accuracy: 0.9468
Epoch 141/150
71/71 [==============================] - 2s 35ms/step - loss: 0.1138 - accuracy: 0.9617
Epoch 142/150
71/71 [==============================] - 2s 35ms/step - loss: 0.1072 - accuracy: 0.9713
Epoch 143/150
71/71 [==============================] - 2s 35ms/step - loss: 0.0573 - accuracy: 0.9833 0s - loss: 0.0574 - acc
uracy: 0.98
Epoch 144/150
71/71 [==============================] - 2s 35ms/step - loss: 0.0681 - accuracy: 0.9792
Epoch 145/150
71/71 [==============================] - 3s 35ms/step - loss: 0.0722 - accuracy: 0.9759
Epoch 146/150
71/71 [==============================] - 2s 35ms/step - loss: 0.0490 - accuracy: 0.9855
Epoch 147/150
71/71 [==============================] - 2s 35ms/step - loss: 0.0482 - accuracy: 0.9862
Epoch 148/150
71/71 [==============================] - 2s 35ms/step - loss: 0.0446 - accuracy: 0.9891
Epoch 149/150
71/71 [==============================] - 2s 35ms/step - loss: 0.0409 - accuracy: 0.9857
Epoch 150/150
71/71 [==============================] - 2s 34ms/step - loss: 0.0819 - accuracy: 0.9739
```

Out[23]:
```
<keras.callbacks.History at 0x2142a427700>
```

In [24]:
```python
model.evaluate(X_test,y_test)
```

```
24/24 [==============================] - 1s 14ms/step - loss: 0.1662 - accuracy: 0.9640
```

Out[24]:
```
[0.16615791618824005, 0.9639999866485596]
```

In [25]:
```python
pred=model.predict(X_test)
pred[3]
```

Out[25]:
```
array([0.00104911, 0.99895084], dtype=float32)
```

In [26]:
```python
pred=[np.argmax(i) for i in pred]
```

In [27]:
```python
y_test[10:15]
```

Out[27]:
```
array([1, 0, 0, 0, 1])
```

In [28]:
```python
pred[10:15]
```

Out[28]:
```
[1, 0, 0, 0, 1]
```

In [29]:
```python
def predict(predict,index):
    if predict[index]==0:
        return "no_tumor"
    else:
        return "yes _tumor"
```

In [30]:
```python
def true(true,index):
    if true[index]==0:
        return "no_tumor"
    else:
        return "yes _tumor"
```

In [31]:
```python
for i in range(0,5):
    print(predict(pred,i))
```

```
no_tumor
no_tumor
yes _tumor
yes _tumor
yes _tumor
```

In [32]:
```python
for i in range(0,5):
    print(true(y_test,i))
```

```
no_tumor
no_tumor
yes _tumor
yes _tumor
yes _tumor
```

In [33]:
```python
from sklearn.metrics import classification_report,confusion_matrix
```

Classification Report For Yes And No

In [36]:
```python
print(classification_report(y_test,pred))
```

```
              precision    recall  f1-score   support

           0       0.98      0.95      0.97       419
           1       0.94      0.98      0.96       331

    accuracy                           0.96       750
   macro avg       0.96      0.97      0.96       750
weighted avg       0.96      0.96      0.96       750
```

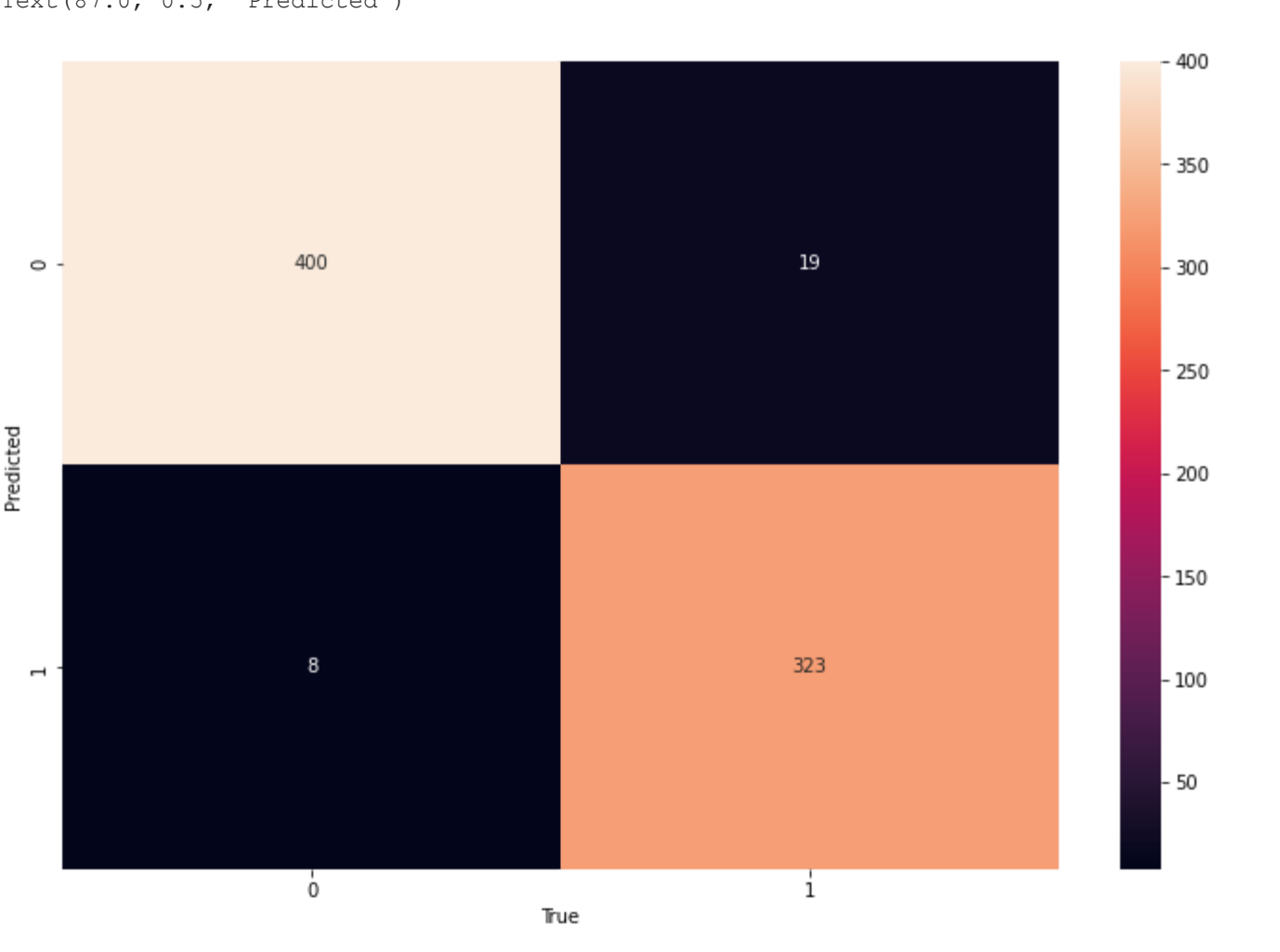
In [37]:
```python
cm=confusion_matrix(y_test,pred)
```

In [38]:
```python
import seaborn as sm
import matplotlib.pyplot as plt
```

Confusion Matrix

In [39]:
```python
plt.figure(figsize=(12,8))
sm.heatmap(cm,annot=True,fmt="d")
plt.xlabel("True")
plt.ylabel("Predicted")
```

Out[39]: Text(87.0, 0.5, 'Predicted')



In [37]:
```python
model.summary()
```

Model: "sequential_1"

Layer (type)	Output Shape	Param #
sequential (Sequential)	(None, 180, 180, 3)	0
conv2d (Conv2D)	(None, 178, 178, 16)	448
max_pooling2d (MaxPooling2D)	(None, 89, 89, 16)	0
conv2d_1 (Conv2D)	(None, 87, 87, 32)	4640
max_pooling2d_1 (MaxPooling2	(None, 43, 43, 32)	0
conv2d_2 (Conv2D)	(None, 41, 41, 64)	18496
max_pooling2d_2 (MaxPooling2	(None, 20, 20, 64)	0
flatten (Flatten)	(None, 25600)	0
dense (Dense)	(None, 128)	3276928
dense_1 (Dense)	(None, 2)	258

Total params: 3,300,770
Trainable params: 3,300,770
Non-trainable params: 0

In [40]:
```python
def predict_input_image(img):
    img_4d=img.reshape(-1,180,180,3)
    prediction=model.predict(img_4d)[0]
    return {brain_class[i]: float(prediction[i]) for i in range(2)}
```

In [41]:
```python
image = gr.inputs.Image(shape=(180,180))
```

In [42]:
```python
label = gr.outputs.Label(num_top_classes=2)
```

Project Demonstration Using Gradio

In [ ]:
```python
gr.Interface(fn=predict_input_image, inputs=image, outputs=label,interpretation='default').launch(debug='True')
```

```
Running on local URL:  http://127.0.0.1:7860/

To create a public link, set `share=True` in `launch()`.
```