```
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense
from tensorflow.keras.optimizers import SGD
from tensorflow.keras.models import load_model
```

```
from google.colab import files

uploaded = files.upload()

# เช็คว่าไฟล์ถูกอัปโหลดหรือไม่
for fn in uploaded.keys():
    print('User uploaded file "{name}" with length {length} bytes'.format(
        name=fn, length=len(uploaded[fn])))
```

```
# ติดตั้ง Kaggle API client
!pip install -q kaggle

# ย้ายไฟล์ kaggle.json ไปยังไดเรกทอรีที่ถูกต้อง
!mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle/
!chmod 600 ~/.kaggle/kaggle.json
```

```
!kaggle datasets download -d navoneel/brain-mri-images-for-brain-tumor-detection
!unzip -q brain-mri-images-for-brain-tumor-detection.zip -d brain_tumor_dataset
```

```
brain-mri-images-for-brain-tumor-detection.zip: Skipping, found more recently modified local copy (use --force to force download)
replace brain_tumor_dataset/brain_tumor_dataset/no/1 no.jpeg? [y]es, [n]o, [A]ll, [N]one, [r]ename:
```

```
!unzip 111.zip -d your_destination_folder
```

```
inflating: your_destination_folder/brain_tumor_dataset/yes/Y32.jpg
inflating: your_destination_folder/brain_tumor_dataset/yes/Y33.jpg
inflating: your_destination_folder/brain_tumor_dataset/yes/Y34.jpg
inflating: your_destination_folder/brain_tumor_dataset/yes/Y35.jpg
inflating: your_destination_folder/brain_tumor_dataset/yes/Y36.JPG
inflating: your_destination_folder/brain_tumor_dataset/yes/Y37.jpg
inflating: your_destination_folder/brain_tumor_dataset/yes/Y38.jpg
inflating: your_destination_folder/brain_tumor_dataset/yes/Y39.jpg
inflating: your_destination_folder/brain_tumor_dataset/yes/Y4.jpg
inflating: your_destination_folder/brain_tumor_dataset/yes/Y40.JPG
inflating: your_destination_folder/brain_tumor_dataset/yes/Y41.jpg
inflating: your_destination_folder/brain_tumor_dataset/yes/Y42.jpg
inflating: your_destination_folder/brain_tumor_dataset/yes/Y44.JPG
inflating: your_destination_folder/brain_tumor_dataset/yes/Y45.JPG
inflating: your_destination_folder/brain_tumor_dataset/yes/Y46.jpg
```

```
!pip install tensorflow
```

```
Requirement already satisfied: tensorflow in /usr/local/lib/python3.10/dist-packages (2.15.0)
Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.4.0)
Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=23.5.26 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (23.5.26)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.5.4)
Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.2.0)
Requirement already satisfied: h5py>=2.9.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.9.0)
Requirement already satisfied: libclang>=13.0.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (16.0.6)
Requirement already satisfied: ml-dtypes~=0.2.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.2.0)
Requirement already satisfied: numpy<2.0.0,>=1.23.5 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.25.2)
Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.3.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from tensorflow) (23.2)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<5.0.0dev,>=3.20.3 in /usr/local/lib/python3.10/dist-packages (from tens
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from tensorflow) (67.7.2)
Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.4.0)
Requirement already satisfied: typing-extensions>=3.6.6 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (4.10.0)
Requirement already satisfied: wrapt<1.15,>=1.11.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.14.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (0.36.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (1.62.0)
Requirement already satisfied: tensorboard<2.16,>=2.15 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.15.2)
Requirement already satisfied: tensorflow-estimator<2.16,>=2.15.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.15.0)
Requirement already satisfied: keras<2.16,>=2.15.0 in /usr/local/lib/python3.10/dist-packages (from tensorflow) (2.15.0)
Requirement already satisfied: wheel<1.0,>=0.23.0 in /usr/local/lib/python3.10/dist-packages (from astunparse>=1.6.0->tensorflow) (0.42.0)
Requirement already satisfied: google-auth<3,>=1.6.3 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.16,>=2.15->tensorflow) (2.27.0)
Requirement already satisfied: google-auth-oauthlib<2,>=0.5 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.16,>=2.15->tensorflow) (1.2.0)
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.16,>=2.15->tensorflow) (3.5.2)
Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.16,>=2.15->tensorflow) (2.31.0)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.16,>=2.15->tensorflow) (0.7.2)
Requirement already satisfied: werkzeug>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from tensorboard<2.16,>=2.15->tensorflow) (3.0.1)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from google-auth<3,>=1.6.3->tensorboard<2.16,>=2.15->tensorflov
Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.10/dist-packages (from google-auth<3,>=1.6.3->tensorboard<2.16,>=2.15->tensorflov
Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.10/dist-packages (from google-auth<3,>=1.6.3->tensorboard<2.16,>=2.15->tensorflow) (4.9)
Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/lib/python3.10/dist-packages (from google-auth-oauthlib<2,>=0.5->tensorboard<2.16,>=2.15->te
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tensorboard<2.16,>=2.15->tensorflov
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tensorboard<2.16,>=2.15->tensorflow) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tensorboard<2.16,>=2.15->tensorflow) (2.0.
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2.21.0->tensorboard<2.16,>=2.15->tensorflow) (2024
Requirement already satisfied: MarkupSafe>=2.1.1 in /usr/local/lib/python3.10/dist-packages (from werkzeug>=1.0.1->tensorboard<2.16,>=2.15->tensorflow) (2.1.5)
Requirement already satisfied: pyasn1<0.6.0,>=0.4.6 in /usr/local/lib/python3.10/dist-packages (from pyasn1-modules>=0.2.1->google-auth<3,>=1.6.3->tensorboard<
Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.10/dist-packages (from requests-oauthlib>=0.7.0->google-auth-oauthlib<2,>=0.5->tensorboarc
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense

# สร้างโมเดลแบบแรก
model1 = Sequential()
model1.add(Conv2D(32, (3, 3), activation='relu', input_shape=(64, 64, 3)))
model1.add(MaxPooling2D((2, 2)))
model1.add(Conv2D(64, (3, 3), activation='relu'))
model1.add(MaxPooling2D((2, 2)))
model1.add(Flatten())
model1.add(Dense(128, activation='relu'))
model1.add(Dense(1, activation='sigmoid'))

# สร้างโมเดลแบบที่สอง
model3 = Sequential()
model3.add(Conv2D(64, (3, 3), activation='relu', input_shape=(64, 64, 3)))
model3.add(MaxPooling2D((2, 2)))
model3.add(Conv2D(128, (3, 3), activation='relu'))
model3.add(MaxPooling2D((2, 2)))
model3.add(Conv2D(256, (3, 3), activation='relu'))
model3.add(MaxPooling2D((2, 2)))
model3.add(Flatten())
model3.add(Dense(512, activation='relu'))
model3.add(Dense(1, activation='sigmoid'))
sgd = SGD(learning_rate=0.01)



model1.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
model3.compile(optimizer='sgd', loss='binary_crossentropy', metrics=['accuracy'])
```

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

train_generator = datagen.flow_from_directory(
    '/content/Brain_T/brain_tumor_dataset',
    target_size=(64, 64),
    batch_size=32,
    class_mode='binary',
    subset='training'
)

validation_generator = datagen.flow_from_directory(
    '/content/Brain_T/brain_tumor_dataset',
    target_size=(64, 64),
    batch_size=32,
    class_mode='binary',
    subset='validation'
)
```

```
Found 203 images belonging to 2 classes.
Found 50 images belonging to 2 classes.
```

```
history1 = model1.fit(train_generator, epochs=10, validation_data=validation_generator)
history2 = model3.fit(train_generator, epochs=10, validation_data=validation_generator)
```

```
Epoch 1/10
7/7 [==============================] - 3s 305ms/step - loss: 0.5971 - accuracy: 0.7044 - val_loss: 0.5183 - val_accuracy: 0.7400
Epoch 2/10
7/7 [==============================] - 2s 281ms/step - loss: 0.4861 - accuracy: 0.7586 - val_loss: 0.5531 - val_accuracy: 0.7400
Epoch 3/10
7/7 [==============================] - 1s 196ms/step - loss: 0.4358 - accuracy: 0.8325 - val_loss: 0.5110 - val_accuracy: 0.7400
Epoch 4/10
7/7 [==============================] - 1s 188ms/step - loss: 0.3892 - accuracy: 0.8374 - val_loss: 0.4974 - val_accuracy: 0.7400
Epoch 5/10
7/7 [==============================] - 1s 195ms/step - loss: 0.3273 - accuracy: 0.8670 - val_loss: 0.5017 - val_accuracy: 0.7800
Epoch 6/10
7/7 [==============================] - 1s 209ms/step - loss: 0.2777 - accuracy: 0.8966 - val_loss: 0.4447 - val_accuracy: 0.8000
Epoch 7/10
7/7 [==============================] - 2s 219ms/step - loss: 0.2305 - accuracy: 0.9064 - val_loss: 0.4239 - val_accuracy: 0.8000
Epoch 8/10
```

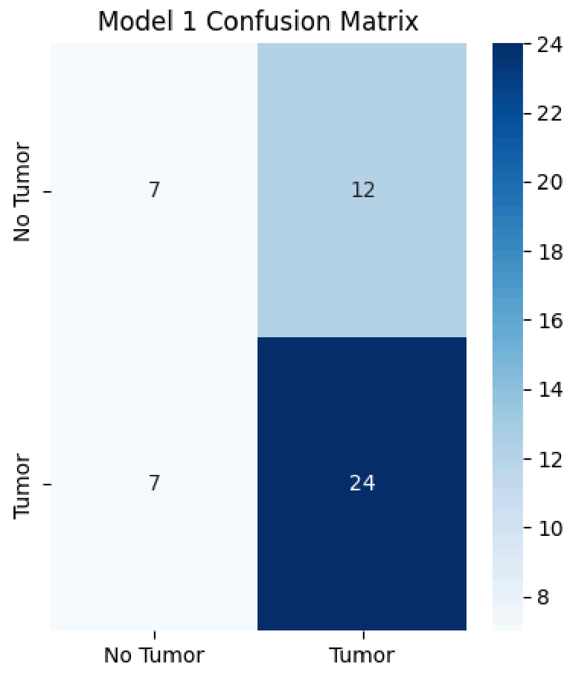
```
7/7 [==============================] - 2s 224ms/step - loss: 0.1891 - accuracy: 0.9261 - val_loss: 0.4028 - val_accuracy: 0.8400
Epoch 9/10
7/7 [==============================] - 1s 194ms/step - loss: 0.1348 - accuracy: 0.9507 - val_loss: 0.4165 - val_accuracy: 0.8600
Epoch 10/10
7/7 [==============================] - 1s 192ms/step - loss: 0.0889 - accuracy: 0.9655 - val_loss: 0.5325 - val_accuracy: 0.8200
Epoch 1/10
7/7 [==============================] - 6s 771ms/step - loss: 0.6797 - accuracy: 0.6158 - val_loss: 0.6571 - val_accuracy: 0.6200
Epoch 2/10
7/7 [==============================] - 4s 541ms/step - loss: 0.6633 - accuracy: 0.6108 - val_loss: 0.6450 - val_accuracy: 0.6200
Epoch 3/10
7/7 [==============================] - 5s 708ms/step - loss: 0.6564 - accuracy: 0.6108 - val_loss: 0.6339 - val_accuracy: 0.6200
Epoch 4/10
7/7 [==============================] - 4s 548ms/step - loss: 0.6499 - accuracy: 0.6108 - val_loss: 0.6280 - val_accuracy: 0.6200
Epoch 5/10
7/7 [==============================] - 4s 541ms/step - loss: 0.6436 - accuracy: 0.6108 - val_loss: 0.6230 - val_accuracy: 0.6200
Epoch 6/10
7/7 [==============================] - 5s 768ms/step - loss: 0.6398 - accuracy: 0.6108 - val_loss: 0.6213 - val_accuracy: 0.6200
Epoch 7/10
7/7 [==============================] - 4s 567ms/step - loss: 0.6371 - accuracy: 0.6108 - val_loss: 0.6177 - val_accuracy: 0.6200
Epoch 8/10
7/7 [==============================] - 5s 597ms/step - loss: 0.6340 - accuracy: 0.6108 - val_loss: 0.6132 - val_accuracy: 0.6200
Epoch 9/10
7/7 [==============================] - 4s 532ms/step - loss: 0.6295 - accuracy: 0.6108 - val_loss: 0.6081 - val_accuracy: 0.6200
Epoch 10/10
7/7 [==============================] - 4s 616ms/step - loss: 0.6289 - accuracy: 0.6158 - val_loss: 0.6044 - val_accuracy: 0.6200
```

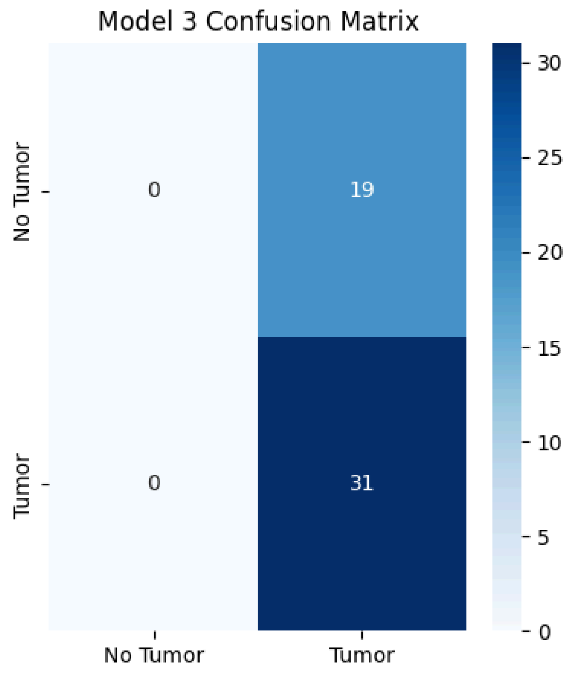
```
score1 = model1.evaluate(validation_generator)
score2 = model3.evaluate(validation_generator)

print("Model 1 - Validation Accuracy:", score1[1])
print("Model 2 - Validation Accuracy:", score2[1])
```

```
2/2 [==============================] - 0s 63ms/step - loss: 0.5325 - accuracy: 0.8200
2/2 [==============================] - 1s 137ms/step - loss: 0.6044 - accuracy: 0.6200
Model 1 - Validation Accuracy: 0.8199999928474426
Model 2 - Validation Accuracy: 0.6200000047683716
```

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

# ทำนายคลาสจากชุดข้อมูลทดสอบ
y_pred1 = (model1.predict(validation_generator) > 0.5).astype("int32")
y_pred3 = (model3.predict(validation_generator) > 0.5).astype("int32")

# ดึงคลาสจริงจากชุดข้อมูลทดสอบ
y_true = validation_generator.classes

# สร้าง confusion matrix
cm1 = confusion_matrix(y_true, y_pred1)
cm3 = confusion_matrix(y_true, y_pred3)

# แสดง confusion matrix
plt.figure(figsize=(15, 5))

plt.subplot(1, 3, 1)
sns.heatmap(cm1, annot=True, fmt="d", cmap="Blues", xticklabels=["No Tumor", "Tumor"], yticklabels=["No Tumor", "Tumor"])
plt.title("Model 1 Confusion Matrix")

plt.subplot(1, 3, 3)
sns.heatmap(cm3, annot=True, fmt="d", cmap="Blues", xticklabels=["No Tumor", "Tumor"], yticklabels=["No Tumor", "Tumor"])
plt.title("Model 3 Confusion Matrix")

plt.show()
```

```
2/2 [==============================] - 0s 39ms/step
2/2 [==============================] - 0s 94ms/step
```





```
# บันทึกโมเดลลงไปยังไฟล์
model1.save('brain_tumor_model1.h5')
```

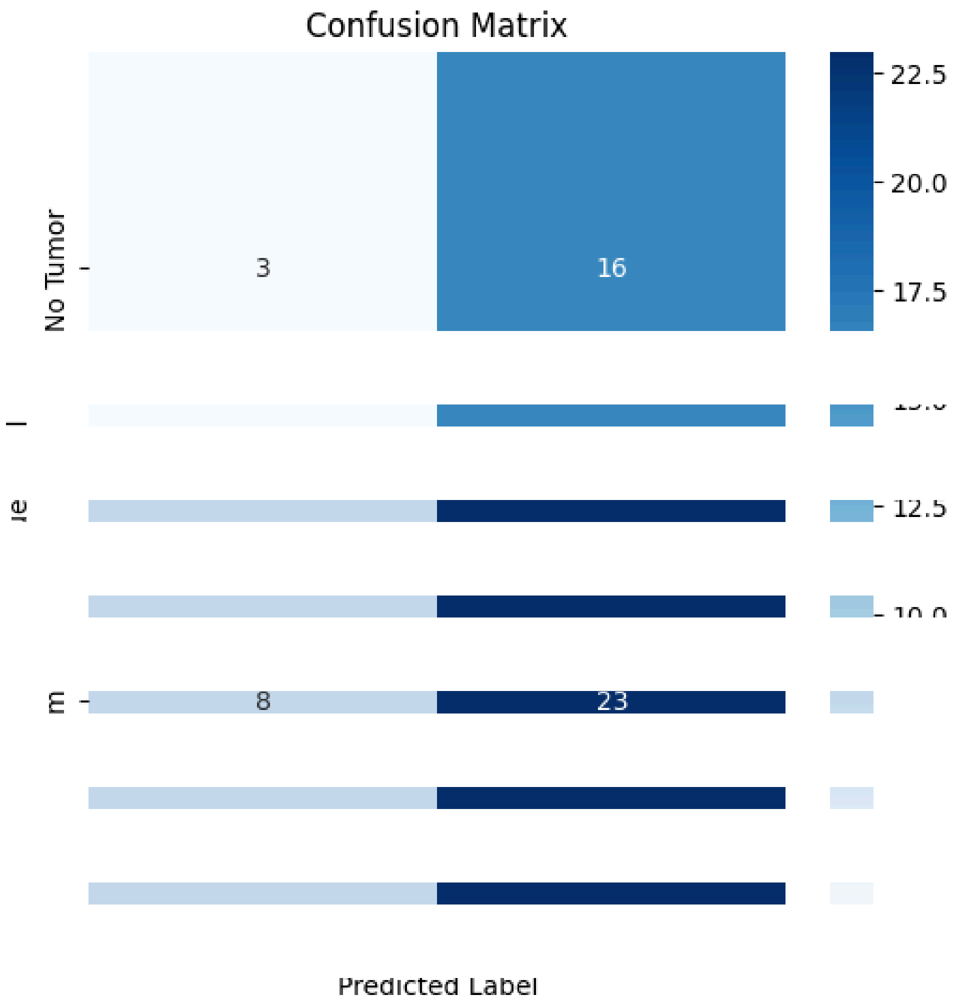
```
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3103: UserWarning: You are saving your model as an HDF5 file via `model.save()`. This file format i
  saving_api.save_model(
```

```
from google.colab import files

# ดาวน์โหลดไฟล์โมเดล
files.download('brain_tumor_model1.h5')
```

```
loaded_model = load_model('brain_tumor_model1.h5')
predictions = (loaded_model.predict(validation_generator) > 0.5).astype("int32")
```

```
2/2 [==============================] - 0s 53ms/step
```

```
cm = confusion_matrix(y_true, predictions)

# แสดงผลลัพธ์เป็นภาพ
plt.figure(figsize=(6, 6))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=["No Tumor", "Tumor"], yticklabels=["No Tumor", "Tumor"])
plt.title("Confusion Matrix")
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.show()
```


Confusion Matrix
No Tumor
3
16
8
23
22.5
20.0
17.5
12.5
Predicted Label