Step 1: Data Preparation

Download and Extract Data:

Download the dataset from the provided Kaggle link. Extract the contents of the downloaded zip file.

Install Required Libraries:

```
!pip install numpy pandas opencv-python scikit-learn tensorflow
```

```
Requirement already satisfied: numpy in c:\users\narthana\anaconda3\
lib\site-packages (1.23.5)
Requirement already satisfied: pandas in c:\users\narthana\anaconda3\
lib\site-packages (1.5.3)
Requirement already satisfied: opencv-python in c:\users\narthana\
anaconda3\lib\site-packages (4.8.1.78)
Requirement already satisfied: scikit-learn in c:\users\narthana\
anaconda3\lib\site-packages (1.2.1)
Requirement already satisfied: tensorflow in c:\users\narthana\
anaconda3\lib\site-packages (2.13.0)
Requirement already satisfied: pytz>=2020.1 in c:\users\narthana\
anaconda3\lib\site-packages (from pandas) (2022.7)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\users\
narthana\anaconda3\lib\site-packages (from pandas) (2.8.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\
narthana\anaconda3\lib\site-packages (from scikit-learn) (2.2.0)
Requirement already satisfied: scipy>=1.3.2 in c:\users\narthana\
anaconda3\lib\site-packages (from scikit-learn) (1.10.0)
Requirement already satisfied: joblib>=1.1.1 in c:\users\narthana\
anaconda3\lib\site-packages (from scikit-learn) (1.1.1)
Requirement already satisfied: tensorflow-intel==2.13.0 in c:\users\
narthana\anaconda3\lib\site-packages (from tensorflow) (2.13.0)
Requirement already satisfied: astunparse>=1.6.0 in c:\users\narthana\
anaconda3\lib\site-packages (from tensorflow-intel==2.13.0-
>tensorflow) (1.6.3)
Requirement already satisfied: google-pasta>=0.1.1 in c:\users\
narthana\anaconda3\lib\site-packages (from tensorflow-intel==2.13.0-
>tensorflow) (0.2.0)
Requirement already satisfied: h5py>=2.9.0 in c:\users\narthana\
anaconda3\lib\site-packages (from tensorflow-intel==2.13.0-
>tensorflow) (3.7.0)
Requirement already satisfied: libclang>=13.0.0 in c:\users\narthana\
anaconda3\lib\site-packages (from tensorflow-intel==2.13.0-
>tensorflow) (16.0.0)
Requirement already satisfied: packaging in c:\users\narthana\
anaconda3\lib\site-packages (from tensorflow-intel==2.13.0-
>tensorflow) (22.0)
Requirement already satisfied: absl-py>=1.0.0 in c:\users\narthana\
anaconda3\lib\site-packages (from tensorflow-intel==2.13.0-
>tensorflow) (1.4.0)
Requirement already satisfied: six>=1.12.0 in c:\users\narthana\
anaconda3\lib\site-packages (from tensorflow-intel==2.13.0-
>tensorflow) (1.16.0)
Requirement already satisfied: opt-einsum>=2.3.2 in c:\users\narthana\
anaconda3\lib\site-packages (from tensorflow-intel==2.13.0-
>tensorflow) (3.3.0)
Requirement already satisfied: flatbuffers>=23.1.21 in c:\users\
narthana\anaconda3\lib\site-packages (from tensorflow-intel==2.13.0-
>tensorflow) (23.5.26)
Requirement already satisfied: termcolor>=1.1.0 in c:\users\narthana\
anaconda3\lib\site-packages (from tensorflow-intel==2.13.0-
>tensorflow) (2.3.0)
Requirement already satisfied: tensorboard<2.14,>=2.13 in c:\users\
narthana\anaconda3\lib\site-packages (from tensorflow-intel==2.13.0-
>tensorflow) (2.13.0)
Requirement already satisfied: wrapt>=1.11.0 in c:\users\narthana\
anaconda3\lib\site-packages (from tensorflow-intel==2.13.0-
>tensorflow) (1.14.1)
Requirement already satisfied: tensorflow-estimator<2.14,>=2.13.0 in
c:\users\narthana\anaconda3\lib\site-packages (from tensorflow-
intel==2.13.0->tensorflow) (2.13.0)
Requirement already satisfied: setuptools in c:\users\narthana\
anaconda3\lib\site-packages (from tensorflow-intel==2.13.0-
>tensorflow) (65.6.3)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in c:\users\
narthana\anaconda3\lib\site-packages (from tensorflow-intel==2.13.0-
>tensorflow) (1.56.0)
Requirement already satisfied: typing-extensions<4.6.0,>=3.6.6 in c:\
users\narthana\anaconda3\lib\site-packages (from tensorflow-
intel==2.13.0->tensorflow) (4.4.0)
Requirement already satisfied: keras<2.14,>=2.13.1 in c:\users\
narthana\anaconda3\lib\site-packages (from tensorflow-intel==2.13.0-
>tensorflow) (2.13.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in
c:\users\narthana\anaconda3\lib\site-packages (from tensorflow-
intel==2.13.0->tensorflow) (0.31.0)
Requirement already satisfied: gast<=0.4.0,>=0.2.1 in c:\users\
narthana\anaconda3\lib\site-packages (from tensorflow-intel==2.13.0-
>tensorflow) (0.4.0)
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 c:\users\narthana\
anaconda3\lib\site-packages (from tensorflow-intel==2.13.0-
>tensorflow) (4.23.4)
Requirement already satisfied: wheel<1.0,>=0.23.0 in c:\users\
narthana\anaconda3\lib\site-packages (from astunparse>=1.6.0-
>tensorflow-intel==2.13.0->tensorflow) (0.38.4)
Requirement already satisfied: werkzeug>=1.0.1 in c:\users\narthana\
anaconda3\lib\site-packages (from tensorboard<2.14,>=2.13->tensorflow-
intel==2.13.0->tensorflow) (2.2.2)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0
in c:\users\narthana\anaconda3\lib\site-packages (from
tensorboard<2.14,>=2.13->tensorflow-intel==2.13.0->tensorflow) (0.7.1)
Requirement already satisfied: markdown>=2.6.8 in c:\users\narthana\
anaconda3\lib\site-packages (from tensorboard<2.14,>=2.13->tensorflow-
intel==2.13.0->tensorflow) (3.4.1)
Requirement already satisfied: requests<3,>=2.21.0 in c:\users\
narthana\anaconda3\lib\site-packages (from tensorboard<2.14,>=2.13-
>tensorflow-intel==2.13.0->tensorflow) (2.28.1)
Requirement already satisfied: google-auth<3,>=1.6.3 in c:\users\
narthana\anaconda3\lib\site-packages (from tensorboard<2.14,>=2.13-
>tensorflow-intel==2.13.0->tensorflow) (2.21.0)
Requirement already satisfied: google-auth-oauthlib<1.1,>=0.5 in c:\
users\narthana\anaconda3\lib\site-packages (from
tensorboard<2.14,>=2.13->tensorflow-intel==2.13.0->tensorflow) (1.0.0)
Requirement already satisfied: rsa<5,>=3.1.4 in c:\users\narthana\
anaconda3\lib\site-packages (from google-auth<3,>=1.6.3-
>tensorboard<2.14,>=2.13->tensorflow-intel==2.13.0->tensorflow) (4.9)
Requirement already satisfied: pyasn1-modules>=0.2.1 in c:\users\
narthana\anaconda3\lib\site-packages (from google-auth<3,>=1.6.3-
>tensorboard<2.14,>=2.13->tensorflow-intel==2.13.0->tensorflow)
(0.2.8)
Requirement already satisfied: urllib3<2.0 in c:\users\narthana\
anaconda3\lib\site-packages (from google-auth<3,>=1.6.3-
>tensorboard<2.14,>=2.13->tensorflow-intel==2.13.0->tensorflow)
(1.26.14)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in c:\users\
narthana\anaconda3\lib\site-packages (from google-auth<3,>=1.6.3-
>tensorboard<2.14,>=2.13->tensorflow-intel==2.13.0->tensorflow)
(5.3.1)
Requirement already satisfied: requests-oauthlib>=0.7.0 in c:\users\
narthana\anaconda3\lib\site-packages (from google-auth-
oauthlib<1.1,>=0.5->tensorboard<2.14,>=2.13->tensorflow-intel==2.13.0-
>tensorflow) (1.3.1)
Requirement already satisfied: idna<4,>=2.5 in c:\users\narthana\
anaconda3\lib\site-packages (from requests<3,>=2.21.0-
>tensorboard<2.14,>=2.13->tensorflow-intel==2.13.0->tensorflow) (3.4)
Requirement already satisfied: charset-normalizer<3,>=2 in c:\users\
narthana\anaconda3\lib\site-packages (from requests<3,>=2.21.0-
>tensorboard<2.14,>=2.13->tensorflow-intel==2.13.0->tensorflow)
(2.0.4)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\
narthana\anaconda3\lib\site-packages (from requests<3,>=2.21.0-
>tensorboard<2.14,>=2.13->tensorflow-intel==2.13.0->tensorflow)
(2022.12.7)
Requirement already satisfied: MarkupSafe>=2.1.1 in c:\users\narthana\
anaconda3\lib\site-packages (from werkzeug>=1.0.1-
>tensorboard<2.14,>=2.13->tensorflow-intel==2.13.0->tensorflow)
(2.1.1)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in c:\users\
narthana\anaconda3\lib\site-packages (from pyasn1-modules>=0.2.1-
>google-auth<3,>=1.6.3->tensorboard<2.14,>=2.13->tensorflow-
intel==2.13.0->tensorflow) (0.4.8)
Requirement already satisfied: oauthlib>=3.0.0 in c:\users\narthana\
anaconda3\lib\site-packages (from requests-oauthlib>=0.7.0->google-
auth-oauthlib<1.1,>=0.5->tensorboard<2.14,>=2.13->tensorflow-
intel==2.13.0->tensorflow) (3.2.2)
```

Import Libraries:

```
import os
import numpy as np
import cv2
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten,
Dense
from tensorflow.keras.utils import to_categorical
```

Load and Preprocess Data:

```
data_dir = "C:\\Users\\Narthana\\Downloads\\brain_tumor_dataset"
categories = ['no', 'yes']

images = []
labels = []

for category in categories:
    path = os.path.join(data_dir, category)
    label = categories.index(category)
    for img in os.listdir(path):
        img_path = os.path.join(path, img)
        img_array = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)
        img_resized = cv2.resize(img_array, (100, 100))
        images.append(img_resized)
        labels.append(label)
```

```
images = np.array(images)
labels = np.array(labels)
```

Split Data into Training and Testing Sets:

```
X_train, X_test, y_train, y_test = train_test_split(images, labels,
test_size=0.2, random_state=42)
```

Normalize Data:

```
X_train = X_train / 255.0
X_test = X_test / 255.0
```

Step 2: Build and Train the Model

Encode Labels:

```
label_encoder = LabelEncoder()
y_train = to_categorical(label_encoder.fit_transform(y_train))
y_test = to_categorical(label_encoder.transform(y_test))
```

Create the Convolutional Neural Network (CNN) Model:

```
model = Sequential([
    Conv2D(32, (3,3), activation='relu', input_shape=(100, 100, 1)),
    MaxPooling2D((2, 2)),
    Flatten(),
    Dense(64, activation='relu'),
    Dense(2, activation='softmax')
])
```

Compile the Model:

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

Train the Model:

```
model.fit(X_train, y_train, validation_data=(X_test, y_test),
epochs=10, batch_size=32)

Epoch 1/10
7/7 [==============================] - 2s 246ms/step - loss: 1.3684 -
accuracy: 0.5446 - val_loss: 0.7669 - val_accuracy: 0.7255
Epoch 2/10
7/7 [==============================] - 1s 204ms/step - loss: 0.6435 -
accuracy: 0.7327 - val_loss: 0.4209 - val_accuracy: 0.8235
Epoch 3/10
7/7 [==============================] - 1s 181ms/step - loss: 0.4538 -
accuracy: 0.8317 - val_loss: 0.4077 - val_accuracy: 0.8235
Epoch 4/10
7/7 [==============================] - 1s 189ms/step - loss: 0.3238 -
accuracy: 0.8812 - val_loss: 0.3093 - val_accuracy: 0.8824
Epoch 5/10
7/7 [==============================] - 1s 182ms/step - loss: 0.2857 -
accuracy: 0.9059 - val_loss: 0.2616 - val_accuracy: 0.9412
Epoch 6/10
7/7 [==============================] - 1s 183ms/step - loss: 0.1949 -
accuracy: 0.9356 - val_loss: 0.2111 - val_accuracy: 0.9216
Epoch 7/10
7/7 [==============================] - 1s 185ms/step - loss: 0.1274 -
accuracy: 0.9703 - val_loss: 0.2008 - val_accuracy: 0.9412
Epoch 8/10
7/7 [==============================] - 1s 208ms/step - loss: 0.0807 -
accuracy: 0.9901 - val_loss: 0.2341 - val_accuracy: 0.9216
Epoch 9/10
7/7 [==============================] - 1s 204ms/step - loss: 0.0584 -
accuracy: 0.9950 - val_loss: 0.2001 - val_accuracy: 0.9020
Epoch 10/10
7/7 [==============================] - 1s 183ms/step - loss: 0.0368 -
accuracy: 1.0000 - val_loss: 0.2707 - val_accuracy: 0.9216

<keras.src.callbacks.History at 0x2d9a72ceef0>
```

Step 3: Evaluate the Model

Evaluate Accuracy:

```
test_loss, test_acc = model.evaluate(X_test, y_test)
print(f'Test accuracy: {test_acc*100:.2f}%')
```

```
2/2 [==============================] - 0s 28ms/step - loss: 0.2707 -
accuracy: 0.9216
Test accuracy: 92.16%
```

Inference on New Images:

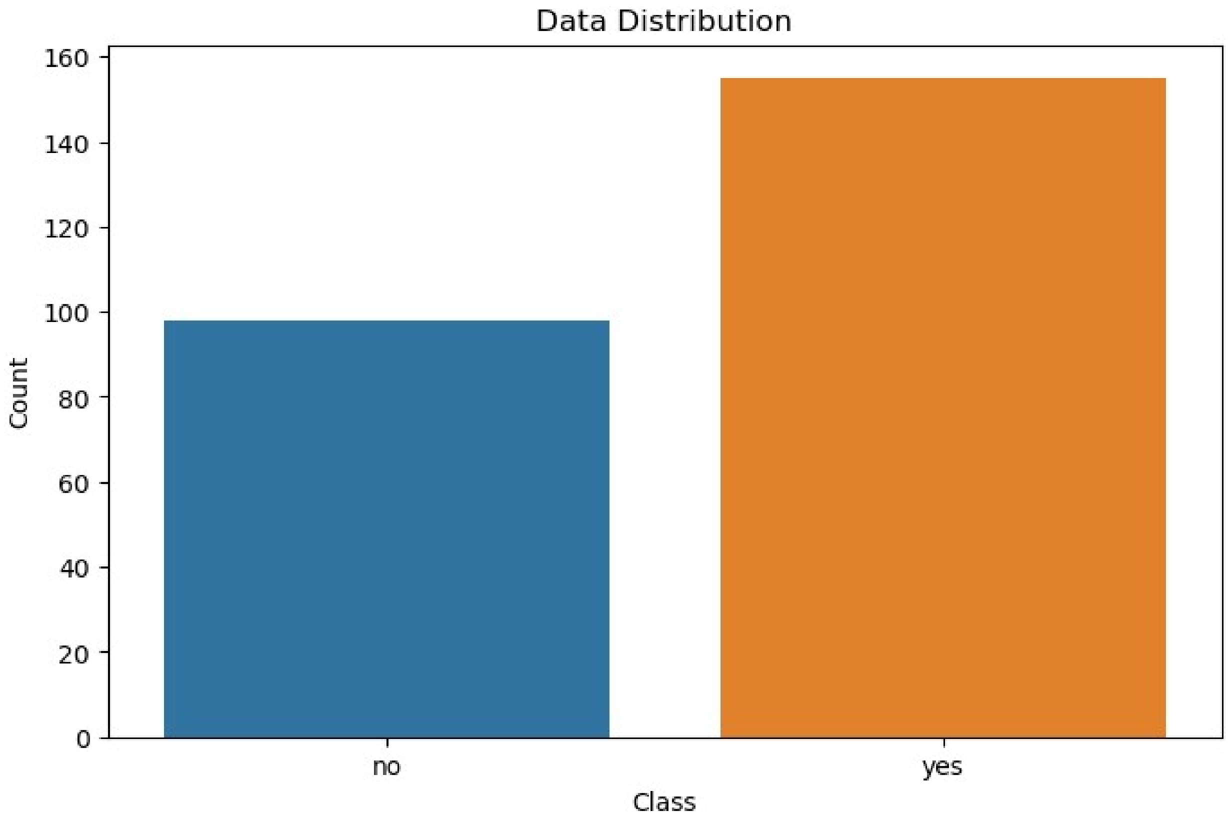
```
def predict_tumor(image_path):
    img_array = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)
    img_resized = cv2.resize(img_array, (100, 100))
    img_normalized = img_resized / 255.0
    img_reshaped = np.reshape(img_normalized, (1, 100, 100, 1))
    prediction = model.predict(img_reshaped)
    label = categories[np.argmax(prediction)]
    return label
```

Usage:

```
result = predict_tumor("C:\\Users\\Narthana\\Downloads\\Screenshot
2023-10-22 122447.png")
print(f'Predicted class: {result}')

1/1 [==============================] - 0s 84ms/step
Predicted class: no

result = predict_tumor("C:\\Users\\Narthana\\Downloads\\Screenshot
2023-10-22 122546.png")
print(f'Predicted class: {result}')

1/1 [==============================] - 0s 29ms/step
Predicted class: yes
```

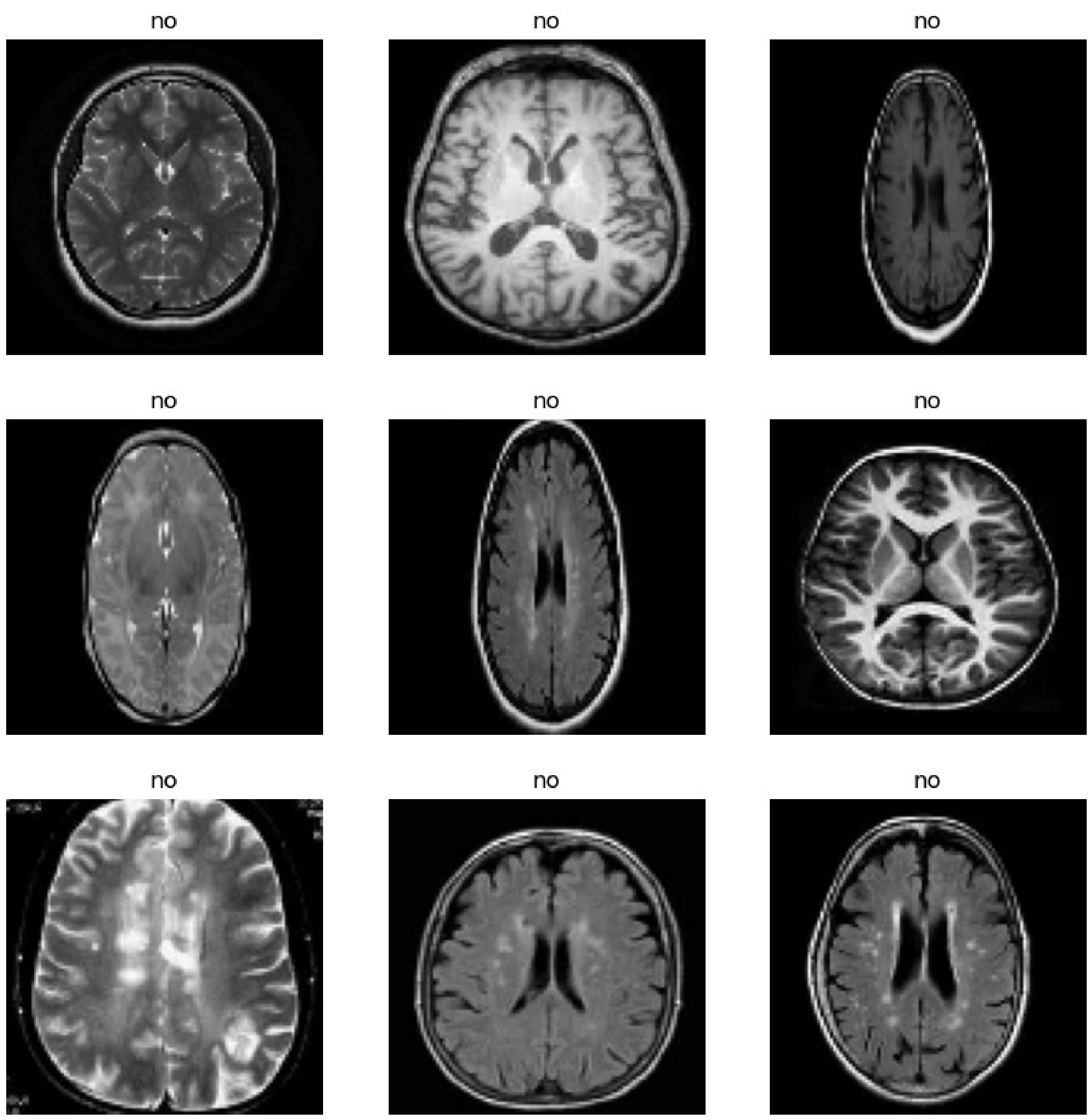
Data Visualization

```
import matplotlib.pyplot as plt
import seaborn as sns

# Count the number of samples in each class
class_counts = np.unique(labels, return_counts=True)

# Create a bar plot
plt.figure(figsize=(8, 5))
sns.barplot(x=categories, y=class_counts[1])
plt.title('Data Distribution')
plt.xlabel('Class')
plt.ylabel('Count')
plt.show()
```



Sample Images:

```
plt.figure(figsize=(10, 10))
for i in range(9):
    plt.subplot(3, 3, i+1)
    plt.imshow(images[i], cmap='gray')
    plt.title(categories[labels[i]])
    plt.axis('off')
plt.show()
```



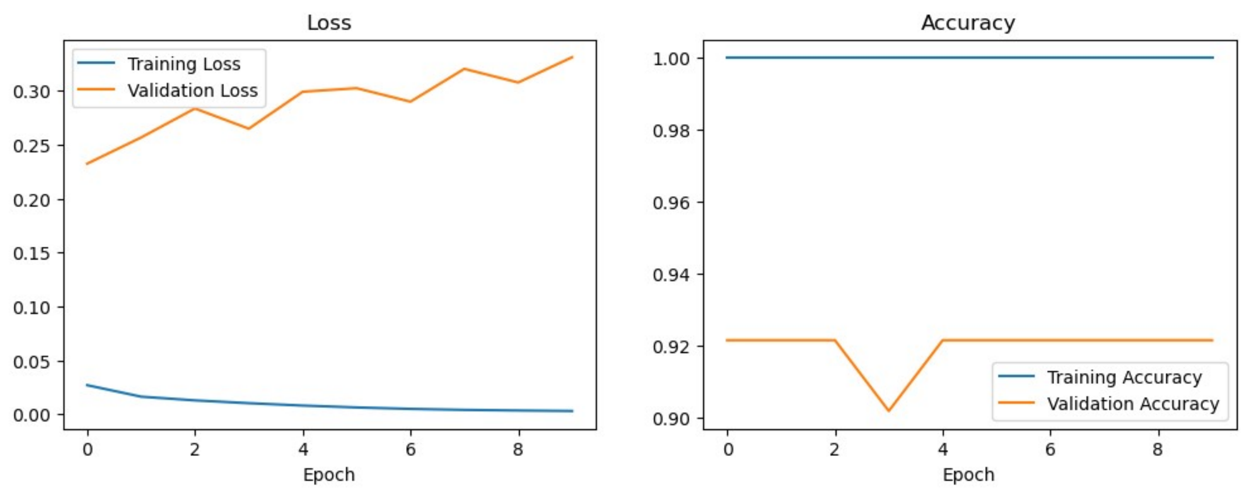
Performance Visualization

Training and Validation Loss and Accuracy:

```
history = model.fit(X_train, y_train, validation_data=(X_test,
y_test), epochs=10, batch_size=32)

plt.figure(figsize=(12, 4))
```

```python
plt.subplot(1, 2, 1)
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.legend()
plt.xlabel('Epoch')
plt.title('Loss')

plt.subplot(1, 2, 2)
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.legend()
plt.xlabel('Epoch')
plt.title('Accuracy')
plt.show()
```

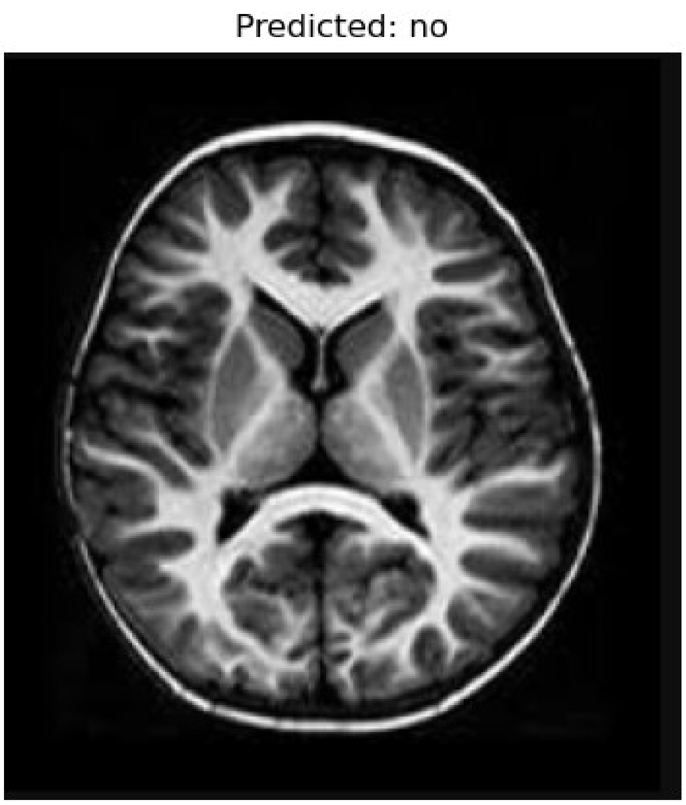
```
Epoch 1/10
7/7 [==============================] - 1s 196ms/step - loss: 0.0270 -
accuracy: 1.0000 - val_loss: 0.2324 - val_accuracy: 0.9216
Epoch 2/10
7/7 [==============================] - 1s 182ms/step - loss: 0.0164 -
accuracy: 1.0000 - val_loss: 0.2567 - val_accuracy: 0.9216
Epoch 3/10
7/7 [==============================] - 1s 181ms/step - loss: 0.0129 -
accuracy: 1.0000 - val_loss: 0.2836 - val_accuracy: 0.9216
Epoch 4/10
7/7 [==============================] - 1s 178ms/step - loss: 0.0103 -
accuracy: 1.0000 - val_loss: 0.2649 - val_accuracy: 0.9020
Epoch 5/10
7/7 [==============================] - 1s 175ms/step - loss: 0.0081 -
accuracy: 1.0000 - val_loss: 0.2991 - val_accuracy: 0.9216
Epoch 6/10
7/7 [==============================] - 1s 175ms/step - loss: 0.0064 -
accuracy: 1.0000 - val_loss: 0.3024 - val_accuracy: 0.9216
Epoch 7/10
7/7 [==============================] - 1s 168ms/step - loss: 0.0050 -
accuracy: 1.0000 - val_loss: 0.2899 - val_accuracy: 0.9216
Epoch 8/10
7/7 [==============================] - 1s 175ms/step - loss: 0.0041 -
accuracy: 1.0000 - val_loss: 0.3203 - val_accuracy: 0.9216
Epoch 9/10
7/7 [==============================] - 1s 181ms/step - loss: 0.0036 -
accuracy: 1.0000 - val_loss: 0.3077 - val_accuracy: 0.9216
Epoch 10/10
7/7 [==============================] - 1s 175ms/step - loss: 0.0031 -
accuracy: 1.0000 - val_loss: 0.3311 - val_accuracy: 0.9216
```



Inference and Results Visualization

Predictions on New Images:

```
def visualize_predictions(image_path):
    result = predict_tumor(image_path)
    img_array = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)
    plt.figure(figsize=(5, 5))
    plt.imshow(img_array, cmap='gray')
    plt.title(f'Predicted: {result}')
    plt.axis('off')
    plt.show()

# Example usage
visualize_predictions("C:\\Users\\Narthana\\Downloads\\Screenshot 
2023-10-22 122447.png")

1/1 [==============================] - 0s 19ms/step
```



```
def visualize_predictions(image_path):
    result = predict_tumor(image_path)
    img_array = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)
    plt.figure(figsize=(5, 5))
    plt.imshow(img_array, cmap='gray')
    plt.title(f'Predicted: {result}')
    plt.axis('off')
    plt.show()

# Example usage
visualize_predictions("C:\\Users\\Narthana\\Downloads\\Screenshot
2023-10-22 122546.png")

1/1 [==============================] - 0s 22ms/step
```

Predicted: yes