Fracture Detection Notebook - Explanation with Code Snippets

This document provides an explanation of the Python code used in the fracture detection notebook, which involves the use of deep learning techniques to classify X-ray images as fractured or non-fractured.

Each section of the notebook is explained along with code snippets.

Step 1: Dataset and Image Preparation

The notebook first loads the dataset of X-ray images and applies transformations to ensure that they are all resized to a uniform dimension (224x224 pixels). This is a common preprocessing step before feeding images into a model for classification.

```
custom_image_transform = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.ToTensor(),
])
```

Step 2: Loading the Model

The model is loaded using PyTorch's `torch.load()` function. This step assumes that a deep learning model has already been trained on the dataset and saved. The model is then set to evaluation mode using `model.eval()`, which disables any training-specific features.

```
model = torch.load('path_to_model.pth')
model.eval()
```

Step 3: Function to Predict on New Images

A custom function is defined to handle the process of loading an image, applying the necessary transformations,

feeding it into the model, and obtaining the model's predictions. It outputs a label ('fractured' or 'non-fractured')

and a probability associated with the prediction.

```
def pred_and_plot_image(model, class_names, image_path, transform):
    # Load the image from the file
    target_image = Image.open(image_path)
    target_image = transform(target_image).unsqueeze(dim=0)

    # Get predictions from the model
    target_image_pred = model(target_image)

    # Apply sigmoid to convert to probabilities
    target_image_pred_probs = torch.sigmoid(target_image_pred)

    # Convert probabilities to labels (rounding them)
    target_image_pred_label = torch.round(target_image_pred_probs)

    # Plot the image with the prediction
    plt.imshow(target_image.squeeze().permute(1, 2, 0))
            title = f"Pred: {class_names[target_image_pred_label.cpu()]} | Prob:
{target_image_pred_probs.cpu():.3f}"
    plt.title(title)
    plt.axis(False)
```

Step 4: Making a Custom Prediction

This part of the notebook allows the user to input the path of an X-ray image they want to classify.

The image is passed through the custom prediction function, and the result (whether the image contains a fracture or not)

is displayed along with the confidence score.

```
image_path = '/path_to_image/fractured/000151.png'

pred_and_plot_image(model, class_names, image_path, custom_image_transform)
```