**1. Data Processing:**

• The MNIST dataset is loaded using mnist.load\_data() function provided by Keras.

• The data is then preprocessed:

• Reshaping the images into a 4D tensor with shape (batch\_size, height, width, channels) where batch\_size is the number of images, and channels is 1 since these are grayscale images.

• Normalizing the pixel values to be between 0 and 1 by dividing by 255.

• One-hot encoding the labels using tf.keras.utils.to\_categorical().

**2. CNN Model Architecture:**

• The model is created using the Sequential API provided by Keras.

• Three convolutional layers are added with max-pooling layers in between:

• The first layer has 32 filters with a kernel size of (3, 3) and ReLU activation function.

• The second layer has 64 filters with a kernel size of (3, 3) and ReLU activation function.

• The third layer has 64 filters with a kernel size of (3, 3) and ReLU activation function.

• After the convolutional layers, two fully connected (Dense) layers are added:

• The first dense layer has 64 units with ReLU activation function.

• The final output layer has 10 units (one for each digit) with softmax activation function.

**3. Model Compilation:**

• The model is compiled using the Adam optimizer, categorical cross-entropy loss function, and accuracy as the metric.

**4. Model Training:**

• The model is trained using model.fit() method on the training data for 5 epochs with a batch size of 64. Validation data is provided to evaluate the model's performance on unseen data during training.

**5. Evaluation:**

• The model's performance is evaluated on the test data using model.evaluate(), and the test accuracy is printed.

**6. Prediction and Visualization:**

• Random samples of 15 images from the test set are chosen.

• The model predicts the labels for these images using model.predict().

• The original images along with their predicted labels are displayed using matplotlib.