Upload the Dataset

```
from google.colab import files
uploaded = files.upload()
```

```
Choose Files P10 mnist_test.csv
  • P10 mnist_test.csv(text/csv) - 18303650 bytes, last modified: 5/14/2025 - 100% done
Saving P10 mnist_test.csv to P10 mnist_test.csv
```

Load the Dataset

Generate | 10 random numbers using numpy | Close

```
from tensorflow.keras.datasets import mnist

# Load dataset
(x_train, y_train), (x_test, y_test) = mnist.load_data()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step
```

Data Exploration

```
print("Training data shape:", x_train.shape)
print("Test data shape:", x_test.shape)
print("Sample labels:", y_train[:10])
```

```
Training data shape: (60000, 28, 28)
Test data shape: (10000, 28, 28)
Sample labels: [5 0 4 1 9 2 1 3 1 4]
```

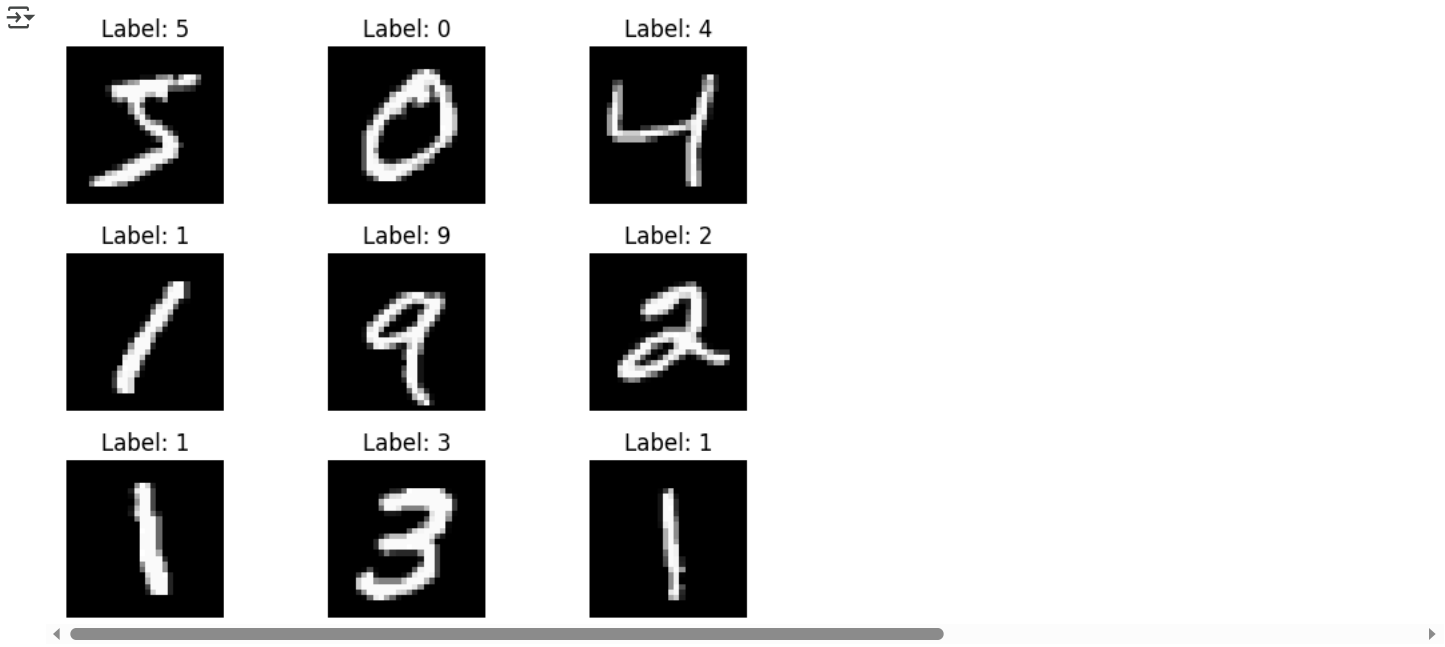
Check for Missing Values and Duplicates

```
# MNIST dataset is clean; still, we check
import numpy  as np
print("Missing values in train data:", np.isnan(x_train).sum())
print("Duplicates in train labels:", len(y_train) - len(np.unique(y_train)))
```

```
Missing values in train data: 0
Duplicates in train labels: 59990
```

Visualize a Few Features

```
import matplotlib.pyplot as plt

for i in range(9):
    plt.subplot(3, 3, i+1)
    plt.imshow(x_train[i], cmap='gray')
    plt.title(f"Label: {y_train[i]}")
    plt.axis('off')
plt.tight_layout()
plt.show()
```



Identify Target and Features

```
# Features (input): pixel values of 28x28 images
print("Feature shape (x_train):", x_train.shape)

# Target (output): digit labels (0-9)
print("Target shape (y_train):", y_train.shape)
print("Unique labels (classes):", np.unique(y_train))
```

```
Feature shape (x_train): (60000, 28, 28)
Target shape (y_train): (60000,)
Unique labels (classes): [0 1 2 3 4 5 6 7 8 9]
```

Convert Categorical Columns to Numerical

```
from tensorflow.keras.utils import to_categorical

# Convert categorical integer labels to one-hot encoded vectors
y_train_encoded = to_categorical(y_train, num_classes=10)
y_test_encoded = to_categorical(y_test, num_classes=10)

# Confirm shape
print("One-hot encoded y_train shape:", y_train_encoded.shape)
print("Example (label 5):", y_train_encoded[5])
```

```
One-hot encoded y_train shape: (60000, 10)
Example (label 5): [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
```

One-Hot Encoding

```
from tensorflow.keras.utils import to_categorical

y_train_encoded = to_categorical(y_train)
y_test_encoded = to_categorical(y_test)
```

Feature Scaling

```
x_train = x_train / 255.0
x_test = x_test / 255.0
```

Train-Test Split

```
from sklearn.model_selection import train_test_split

# First, reshape and normalize image data
x_train = x_train / 255.0
x_test = x_test / 255.0

x_train_cnn = x_train.reshape(-1, 28, 28, 1)
x_test_cnn = x_test.reshape(-1, 28, 28, 1)

# One-hot encode labels
from tensorflow.keras.utils import to_categorical
y_train_encoded = to_categorical(y_train, num_classes=10)
y_test_encoded = to_categorical(y_test, num_classes=10)

# Split training data into new training and validation sets (e.g., 90% train, 10% validation)
x_train_final, x_val, y_train_final, y_val = train_test_split(
    x_train_cnn, y_train_encoded, test_size=0.1, random_state=42
)

# Print the shapes
print("Train data shape:", x_train_final.shape)
print("Validation data shape:", x_val.shape)
print("Test data shape:", x_test_cnn.shape)
```

```
Train data shape: (54000, 28, 28, 1)
Validation data shape: (6000, 28, 28, 1)
Test data shape: (10000, 28, 28, 1)
```

Model Building

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense

# Reshape for CNN
x_train_cnn = x_train.reshape(-1, 28, 28, 1)
x_test_cnn = x_test.reshape(-1, 28, 28, 1)

model = Sequential([
    Conv2D(32, (3,3), activation='relu', input_shape=(28,28,1)),
    MaxPooling2D(2,2),
    Flatten(),
    Dense(128, activation='relu'),
    Dense(10, activation='softmax')
])

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

```
/usr/local/lib/python3.11/dist-packages/keras/src/layers/convolutional/base_conv.py:107: UserWarning: Do not pass an `input_shape`/`inpu
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 26, 26, 32)	320
max_pooling2d (MaxPooling2D)	(None, 13, 13, 32)	0
flatten (Flatten)	(None, 5408)	0
dense (Dense)	(None, 128)	692,352
dense_1 (Dense)	(None, 10)	1,290

Total params: 693,962 (2.65 MB)
Trainable params: 693,962 (2.65 MB)
Non-trainable params: 0 (0.00 B)

Evaluation

```
history = model.fit(x_train_cnn, y_train_encoded, epochs=5, validation_split=0.1)

# Evaluate
test_loss, test_acc = model.evaluate(x_test_cnn, y_test_encoded)
print(f"\nTest Accuracy: {test_acc:.2f}")
```

```
Epoch 1/5
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 39s 22ms/step - accuracy: 0.4984 - loss: 1.5619 - val_accuracy: 0.9060 - val_loss: 0.3445
Epoch 2/5
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 42s 23ms/step - accuracy: 0.8827 - loss: 0.4042 - val_accuracy: 0.9243 - val_loss: 0.2680
Epoch 3/5
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 38s 22ms/step - accuracy: 0.9039 - loss: 0.3243 - val_accuracy: 0.9270 - val_loss: 0.2392
Epoch 4/5
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 40s 23ms/step - accuracy: 0.9111 - loss: 0.3005 - val_accuracy: 0.9368 - val_loss: 0.2145
Epoch 5/5
1688/1688 ━━━━━━━━━━━━━━━━━━━━ 41s 24ms/step - accuracy: 0.9192 - loss: 0.2699 - val_accuracy: 0.9425 - val_loss: 0.1961
313/313 ━━━━━━━━━━━━━━━━━━━━ 2s 8ms/step - accuracy: 0.9229 - loss: 0.2640

Test Accuracy: 0.93
```

Make Predictions from New Input

```
predictions = model.predict(x_test_cnn)
predicted_labels = np.argmax(predictions, axis=1)
```

```
313/313 ━━━━━━━━━━━━━━━━━━━━ 3s 9ms/step
```

Convert to DataFrame and Encode

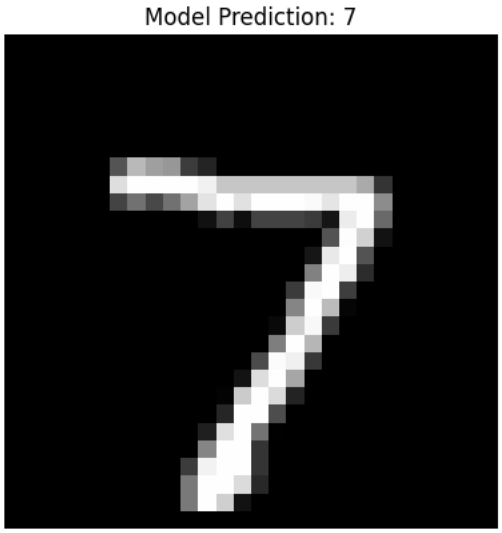
```
import pandas as pd

df_preds = pd.DataFrame({'Actual': y_test, 'Predicted': predicted_labels})
df_preds.head(10)
```

	Actual	Predicted
0	7	7
1	2	2
2	1	1
3	0	0
4	4	4
5	1	1
6	4	4
7	9	9
8	5	6
9	9	9

Next steps: Generate code with df_preds | View recommended plots | New interactive sheet

Predict the Final Grade (i.e., Digit)

```
# Show one prediction
plt.imshow(x_test[0], cmap='gray')
plt.title(f"Model Prediction: {predicted_labels[0]}")
plt.axis('off')
plt.show()
```



Deployment – Building an Interactive App

```
!pip install gradio
import gradio as gr
```

```
Requirement already satisfied: gradio in /usr/local/lib/python3.11/dist-packages (5.29.0)
Requirement already satisfied: aiofiles<25.0,>=22.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (24.1.0)
Requirement already satisfied: anyio<5.0,>=3.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (4.9.0)
Requirement already satisfied: fastapi<1.0,>=0.115.2 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.115.12)
Requirement already satisfied: ffmpy in /usr/local/lib/python3.11/dist-packages (from gradio) (0.5.0)
Requirement already satisfied: gradio-client==1.10.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (1.10.0)
Requirement already satisfied: groovy~=0.1 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.1.2)
Requirement already satisfied: httpx>=0.24.1 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.28.1)
Requirement already satisfied: huggingface-hub>=0.28.1 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.31.1)
Requirement already satisfied: jinja2<4.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (3.1.6)
Requirement already satisfied: markupsafe<4.0,>=2.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (3.0.2)
Requirement already satisfied: numpy<3.0,>=1.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (2.0.2)
Requirement already satisfied: orjson~=3.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (3.10.18)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-packages (from gradio) (24.2)
Requirement already satisfied: pandas<3.0,>=1.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (2.2.2)
Requirement already satisfied: pillow<12.0,>=8.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (11.2.1)
Requirement already satisfied: pydantic<2.12,>=2.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (2.11.4)
Requirement already satisfied: pydub in /usr/local/lib/python3.11/dist-packages (from gradio) (0.25.1)
Requirement already satisfied: python-multipart>=0.0.18 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.0.20)
Requirement already satisfied: pyyaml<7.0,>=5.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (6.0.2)
Requirement already satisfied: ruff>=0.9.3 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.11.9)
Requirement already satisfied: safehttpx<0.2.0,>=0.1.6 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.1.6)
Requirement already satisfied: semantic-version~=2.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (2.10.0)
Requirement already satisfied: starlette<1.0,>=0.40.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.46.2)
Requirement already satisfied: tomlkit<0.14.0,>=0.12.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.13.2)
Requirement already satisfied: typer<1.0,>=0.12 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.15.3)
Requirement already satisfied: typing-extensions~=4.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (4.13.2)
Requirement already satisfied: uvicorn>=0.14.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.34.2)
Requirement already satisfied: fsspec in /usr/local/lib/python3.11/dist-packages (from gradio-client==1.10.0->gradio) (2025.3.2)
Requirement already satisfied: websockets<16.0,>=10.0 in /usr/local/lib/python3.11/dist-packages (from gradio-client==1.10.0->gradio) (1
Requirement already satisfied: idna>=2.8 in /usr/local/lib/python3.11/dist-packages (from anyio<5.0,>=3.0->gradio) (3.10)
Requirement already satisfied: sniffio>=1.1 in /usr/local/lib/python3.11/dist-packages (from anyio<5.0,>=3.0->gradio) (1.3.1)
Requirement already satisfied: certifi in /usr/local/lib/python3.11/dist-packages (from httpx>=0.24.1->gradio) (2025.4.26)
Requirement already satisfied: httpcore==1.* in /usr/local/lib/python3.11/dist-packages (from httpx>=0.24.1->gradio) (1.0.9)
Requirement already satisfied: h11>=0.16 in /usr/local/lib/python3.11/dist-packages (from httpcore==1.*->httpx>=0.24.1->gradio) (0.16.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.28.1->gradio) (3.18.0)
Requirement already satisfied: requests in /usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.28.1->gradio) (2.32.3)
Requirement already satisfied: tqdm>=4.42.1 in /usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.28.1->gradio) (4.67.1)
Requirement already satisfied: hf-xet<2.0.0,>=1.1.0 in /usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.28.1->gradio) (1
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas<3.0,>=1.0->gradio) (2.9.0.
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas<3.0,>=1.0->gradio) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas<3.0,>=1.0->gradio) (2025.2)
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.11/dist-packages (from pydantic<2.12,>=2.0->gradio) (0.7
Requirement already satisfied: pydantic-core==2.33.2 in /usr/local/lib/python3.11/dist-packages (from pydantic<2.12,>=2.0->gradio) (2.33
Requirement already satisfied: typing-inspection>=0.4.0 in /usr/local/lib/python3.11/dist-packages (from pydantic<2.12,>=2.0->gradio) (0
Requirement already satisfied: click>=8.0.0 in /usr/local/lib/python3.11/dist-packages (from typer<1.0,>=0.12->gradio) (8.1.8)
Requirement already satisfied: shellingham>=1.3.0 in /usr/local/lib/python3.11/dist-packages (from typer<1.0,>=0.12->gradio) (1.5.4)
Requirement already satisfied: rich>=10.11.0 in /usr/local/lib/python3.11/dist-packages (from typer<1.0,>=0.12->gradio) (13.9.4)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->pandas<3.0,>=1.0->gradi
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3.11/dist-packages (from rich>=10.11.0->typer<1.0,>=0.12->g
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.11/dist-packages (from rich>=10.11.0->typer<1.0,>=0.12-
```

```
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests->huggingface-hub>=0.28
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests->huggingface-hub>=0.28.1->gr
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.11/dist-packages (from markdown-it-py>=2.2.0->rich>=10.11.0->typer<1
```

Create a Prediction Function

```
def predict_digit(image):
    import numpy as np
    img = image.reshape(1, 28, 28, 1)
    img = img / 255.0
    pred = model.predict(img)
    return np.argmax(pred)
```

Create the Gradio Interface

```
# Step 1: Install Gradio (if not already installed)
!pip install gradio --quiet

# Step 2: Import libraries
import gradio as gr
import numpy as np
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense
from tensorflow.keras.utils import to_categorical
from PIL import Image

# Step 3: Load and preprocess MNIST data
(x_train, y_train), _ = mnist.load_data()
x_train = x_train.reshape(-1, 28, 28, 1) / 255.0
y_train_cat = to_categorical(y_train)

# Step 4: Build and train model
model = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)),
    MaxPooling2D((2, 2)),
    Flatten(),
    Dense(128, activation='relu'),
    Dense(10, activation='softmax')
])
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.fit(x_train, y_train_cat, epochs=1)

# Step 5: Prediction function
def predict_digit(img):
    img = img.convert('L').resize((28, 28))         # Convert to grayscale and resize
    img_array = 255 - np.array(img)                 # Invert image (white bg to black bg)
    img_array = img_array / 255.0                   # Normalize
    img_array = img_array.reshape(1, 28, 28, 1)     # Reshape for model
    prediction = model.predict(img_array)
    return f"Predicted Digit: {np.argmax(prediction)}"

# Step 6: Gradio interface (NO deprecated args)
gr.Interface(
    fn=predict_digit,
    inputs=gr.Image(type="pil", image_mode="L"),
    outputs="text",
    title="Handwritten Digit Recognition",
    description="Draw a digit (0-9) and get the predicted result!"
).launch(share=True)
```

```
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 42s 22ms/step - accuracy: 0.9191 - loss: 0.2858
Colab notebook detected. To show errors in colab notebook, set debug=True in launch()
* Running on public URL: https://fe373b3f96e634c3ea.gradio.live

This share link expires in 1 week. For free permanent hosting and GPU upgrades, run `gradio deploy` from the terminal in the working dir
```