Machine Learning
LABORATORY: Gradient Descent In Class

NAME:	STUDENT ID#:

Objectives:

- Understand and implement Stochastic Gradient Descent (SGD) using one sample at a time (Algorithm 7.1).
- Apply the sigmoid activation function to perform binary classification.
- Learn how to update model weights manually using the gradient of the loss function.
- Practice using NumPy for mathematical operations on real MNIST data.
- Evaluate classification results using accuracy and misclassified samples visualization.

Part 1. Instruction

- In this assignment, you will implement Stochastic Gradient Descent (SGD) from scratch using Algorithm 7.1 provided in class.
- The task is to build a binary classifier to distinguish whether an MNIST image corresponds to a target digit or not (e.g., "Is this a 0 or not?").
- You will complete the code template provided in the in-class assignment.
- The classifier will use the sigmoid function for binary probability output. Use only NumPy for all computations. Do not use libraries like scikit-learn or PyTorch.
- Evaluate your results using the printed accuracy and a visualization of misclassified samples.

Part 2. Arithmetic Instructions.

Algorithm 7.1: Stochastic gradient descent

Input: Training set of data points indexed by $n \in \{1, \ldots, N\}$
Error function per data point $E_n(\mathbf{w})$
Learning rate parameter η
Initial weight vector $\mathbf{w}$
Output: Final weight vector $\mathbf{w}$

$n \leftarrow 1$
repeat
$\quad \mathbf{w} \leftarrow \mathbf{w} - \eta \nabla E_n(\mathbf{w})$ // update weight vector
$\quad n \leftarrow n + 1 (\text{mod } N)$ // iterate over data
until convergence
return $\mathbf{w}$

Lecture: Prof. Hsien-I Lin
TA: Satrio Sanjaya and Muhammad Ahsan

Part 3. Code Template.

Step	Procedure
| 1 | ```python
#Load Dataset
import struct
import numpy as np
import matplotlib.pyplot as plt

# ==========Load IDX Files ==========
def load_images(filename):
    with open(filename, 'rb') as f:
        _, num, rows, cols = struct.unpack(">IIII", f.read(16))
        images=np.frombuffer(f.read(), dtype=np.uint8)
        images = images[:(len(images)//(rows * cols)) * rows * cols]
        return images.reshape(-1, rows * cols).astype(np.float32) / 255.0

def load_labels(filename):
    with open(filename, 'rb') as f:
        _, num = struct.unpack(">II", f.read(8))
        labels = np.frombuffer(f.read(), dtype=np.uint8)
        return labels[:num]
``` |
| 2 | ```python
# ========== 1. Sigmoid Function ==========
def sigmoid(z):
    # TODO: Implement sigmoid function
    pass

# ========== 2. SGD: Algorithm 7.1 ==========
def sgd_logistic(X, y, eta, max_iters):

    pass
``` |
| 3 | ```python
# ==========Show Misclassified Samples ==========
def show_misclassified(X, true_labels, pred_labels, max_show=10):
    mis_idx = np.where(true_labels != pred_labels)[0][:max_show]
    plt.figure(figsize=(10, 2))
    for i, idx in enumerate(mis_idx):
        plt.subplot(1, len(mis_idx), i + 1)
        plt.imshow(X[idx, 1:].reshape(28, 28), cmap='gray')
        plt.axis('off')
        plt.title(f"T:{true_labels[idx]} P:{pred_labels[idx]}")
    plt.suptitle("Misclassified Samples")
    plt.show()
``` |
| 4 | ```python
# ========== 3. Main ==========
if __name__ == "__main__":
    # === Load Data ===
    X_train = load_images("train-images.idx3-ubyte__")
    y_train = load_labels("train-labels.idx1-ubyte__")
    X_test = load_images("t10k-images.idx3-ubyte__")
    y_test = load_labels("t10k-labels.idx1-ubyte__")
``` |

Lecture: Prof. Hsien-I Lin
TA: Satrio Sanjaya and Muhammad Ahsan

```
# === Choose binary classification target digit ===
TARGET_DIGIT = 0    # TODO: Fill in (0 to 9)

y_train_bin = np.where(y_train == TARGET_DIGIT, 1, 0)
y_test_bin = np.where(y_test == TARGET_DIGIT, 1, 0)

# === Add bias term ===
X_train  =  np.hstack([np.ones((X_train.shape[0],  1)),
X_train])
    X_test  =  np.hstack([np.ones((X_test.shape[0],  1)),
X_test])

# === Set parameters ===
#eta = _____          # TODO: Learning rate
#max_iters = _____    # TODO: Number of SGD iterations

# === Train ===
#w = sgd_logistic(X_train, y_train_bin, eta, max_iters)

# === Predict ===
#pred_probs = ____
#preds = (pred_probs >= 0.5).astype(int)

# === Evaluate ===
#accuracy = np.mean(preds == y_test_bin)
#print(f"Accuracy: {accuracy:.4f}")

# === Show Misclassified Samples ===
#show_misclassified(X_test, y_test_bin, preds)
```
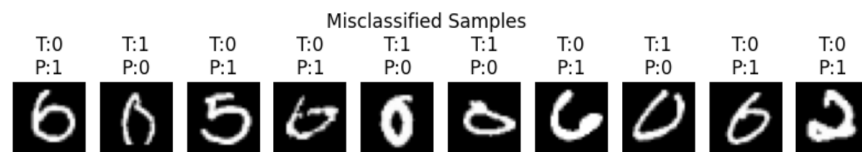
| 5 | #Example Output: |

```
[INFO] Header: 60000 images, 28x28
[INFO] Loading 60000 images based on file size
[INFO] Loading 60000 labels based on file size
[INFO] Header: 10000 images, 28x28
[INFO] Loading 10000 images based on file size
[INFO] Loading 10000 labels based on file size

[INFO] Binary classification: '0' vs not-0

Test Accuracy (is 0 or not): 0.9927
```


Misclassified Samples

Grading Assignment & Submission (30% Max)

Implementation:

1. (15%) Implement the Stochastic Gradient Descent (SGD) algorithm based on Algorithm 7.1.
2. (10%) The model runs successfully without errors, Trains using the provided MNIST dataset, and output the test accuracy.
3. (5%) Set the class of binary classification to the last digit of your student ID. (e.g., if your ID ends in 7, use the class '7'). Displays at least 5 misclassified test images along with their true (T:) and predicted (P:) labels, as shown in the "Example Output" in the last pages of this document.

Lecture: Prof. Hsien-I Lin
TA: Satrio Sanjaya and Muhammad Ahsan

Submission :

1. Report: Answer all conceptual questions. Include screenshots of your results in the last pages of this PDF File.
2. Code: Submit your complete Python script in either .py or .ipynb format.
3. Upload both your report and code to the E3 system (**Labs4 In Class Assignment**). Name your files correctly:
   a. Report: StudentID_Lab4_InClass.pdf
   b. Code: StudentID_Lab4_ InClass.py or StudentID_Lab4_InClass.ipynb
4. Deadline: 16:20 PM
5. Plagiarism is **strictly prohibited**. Submitting copied work from other students will result in penalties.
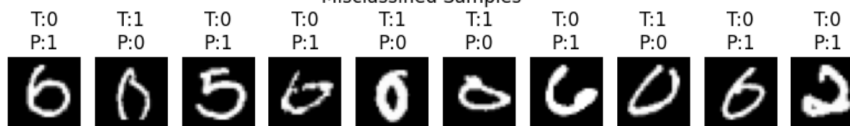
Example Output (Just for reference):


```
[INFO] Header: 60000 images, 28x28
[INFO] Loading 60000 images based on file size
[INFO] Loading 60000 labels based on file size
[INFO] Header: 10000 images, 28x28
[INFO] Loading 10000 images based on file size
[INFO] Loading 10000 labels based on file size

[INFO] Binary classification: '0' vs not-0

Test Accuracy (is 0 or not): 0.9927
```

Code Results and Answer:


```
Training binary classifier for digit 4...
Iteration 10/30 completed
Iteration 20/30 completed
Iteration 30/30 completed
Accuracy: 0.9816
```

Lecture: Prof. Hsien-I Lin
TA: Satrio Sanjaya and Muhammad Ahsan