HUCENROTIA LAB 機器學習

LABORATORY: Classification

NAME: 張子中	STUDENT ID#: 313605013

Objectives:

- Develop a deeper understanding of classification models, focusing on logistic regression.
- Learn the mathematical foundations of sigmoid (logistic) activation functions.
- Implement logistic regression from scratch using NumPy without relying on sklearn's built-in classifiers.
- Train models using gradient descent, analyze their convergence, and evaluate their performance.
- Apply key classification metrics, including confusion matrices, ROC curves, and AUC, to compare model effectiveness.

Part 1. Background

Classification is a fundamental machine learning task that involves predicting categorical labels. Logistic Regression uses the sigmoid function to model probabilities and is widely used for binary classification.

Tasks: In this assignment, you will implement logistic regression using only NumPy. You will get no points by simply calling *sklearn.linear_model.LogisticRegression*. Your task is to train these models on a provided dataset, evaluate their performance, and test to classify them. Compare confusion matrices, ROC curves, and AUC to understand their strengths and limitations. The dataset and sample code can be found here: https://github.com/Satriosnjya/ML-Labs.git

Classification Tasks: The goal of the classification model (Logistic Regression) is to predict y, which is a binary label (0 or 1). **What are we classifying?** We are predicting whether an individual earns more than 50K per year based on demographic and work-related factors.

- Input Features:
  - Categorical variables (age_bin, occupation_bin, education_bin, etc.), which need to be converted into numerical features using one-hot encoding before training.
- Output (Target Variable y):
  - A binary classification label (0 or 1).

Part 2. Arithmetic Instructions.

Step	Procedure
1	**Logistic Regression:** 1. (__init__) We initialize the weights and bias: $w = 0,\ b = 0$ 2. (fit) • Linear Model • Sigmoid Function (Logistic Activation Function): $\sigma(a) = \frac{1}{1+\exp(-a)}$ *(5.42)* • Update weight : $\nabla E(\mathbf{w}) = \sum_{n=1}^{N} (y_n - t_n)\phi_n$ *(5.75)*

3. (predict_proba) Uses the sigmoid function to compute probability
4. (predict) Binary Classification Decision
   Convert probabilities to class labels using a threshold $\tau = 0.5$

2 Confusion Matrix

A confusion matrix is a table used to evaluate the performance of a classification model. It consists of the following four elements:

$$CM = \begin{bmatrix} TN & FP \\ FN & TP \end{bmatrix}$$

where:

- True Positive (TP): The model correctly predicts class 1.
- True Negative (TN): The model correctly predicts class 0 .
- False Positive (FP): The model predicts 1 when the actual class is 0 .
- False Negative (FN): The model predicts 0 when the actual class is 1 .

Several other quantities can be defined as:

$$\text{Accuracy} = \frac{N_{\text{TP}}+N_{\text{TN}}}{N_{\text{TP}}+N_{\text{FP}}+N_{\text{TN}}+N_{\text{FN}}} \quad (5.29)$$

$$Precision = \frac{N_{TP}}{N_{TP}+N_{FP}} \quad (5.30)$$

$$Recall = \frac{N_{TP}}{N_{TP}+N_{FN}} \quad (5.31)$$

3 ROC Curve and AUC

The ROC curve is a plot that shows the trade-off between the True Positive Rate (TPR) and the False Positive Rate (FPR) at different classification thresholds. The Area Under the Curve (AUC) for the ROC curve measures the model's ability to distinguish between classes.

True Positive Rate (TPR), Also known as recall or sensitivity:

$$TPR = \frac{TP}{TP + FN}$$

False Positive Rate (FPR), The proportion of incorrectly predicted positives out of all actual negatives:

$$FPR = \frac{FP}{FP + TN}$$

4 F-*score*

F-score, which is the geometric mean of precision and recall:

$$F = \frac{2\,x\,precision\,x\,recall}{precision+recall} \quad (5.38)$$

$$= \frac{2\,x\,N_{TP}}{2N_{TP}+N_{FP}+N_{FN}} \quad (5.39)$$

Part 3. Data Transfer Instructions.

Step	Procedure
1	• Download dataset and example code from https://github.com/Satriosnjya/ML-Labs.git • Open colab.research.google.com • Make a New Notebook • Upload classification.csv in Colab Notebook
2	**Load Libraries** *import numpy as np* *import pandas as pd* *import matplotlib.pyplot as plt* *import seaborn as sns*

2	**Generate Data**
	# Load Dataset
	file_path = "classification.csv"
	df = pd.read_csv(file_path)

```
# Data Preparation
cat_feats = ['age_bin','capital_gl_bin','education_bin','hours_
per_week_bin','msr_bin','occupation_bin','race_sex_bin']
```

3 **Split Dataset**

```
# Split dataset into train and test
y_train = df[df['flag']=='train']['y']
x_train = df[df['flag']=='train'][cat_feats]
x_train = pd.get_dummies(x_train, columns=cat_feats, drop_first=True)

y_test = df[df['flag']=='test']['y']
x_test = df[df['flag']=='test'][cat_feats]
x_test = pd.get_dummies(x_test, columns=cat_feats, drop_first=True)
```

4 **Convert to np arrays and apply manual feature scaling**

```
# Convert to numpy arrays
x_train, y_train = x_train.values, y_train.values
x_test, y_test = x_test.values, y_test.values

# Apply manual feature scaling
mean_train = np.mean(x_train, axis=0)
std_train = np.std(x_train, axis=0)
x_train_scaled = (x_train - mean_train) / std_train
x_test_scaled = (x_test - mean_train) / std_train
```

Grading & Submission Instructions

Assignment (70%):

1. (20%) Implement your Logistic Regression model from scratch
    a. Train the model using
        i. learning rate = 0.01, and iteration =3000
        ii. learning rate = 0.0001, and iteration =5000
        iii. learning rate = 0.001, and iteration =1000
2. Compute Classifier Accuracy
    a. (10%) Compute True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN).
    b. (10%) Construct a confusion matrix heatmap for logistic regression.
    c. (10%) Compute the evaluation for Precision, Recall, and Accuracy.
3. Generate and Analyze the ROC Curve.
    a. (7%) Compute TPR and FPR for different threshold.
    b. (7%) Plot the ROC curve and show the AUC score.
    c. (6%) Compute the F-*score*.

Question (30%):

1. (7.5%) Given your confusion matrix, analyze the types of errors made by the model. Identify which type of error (False Positives or False Negatives) is more common in your dataset. Why might this be the case?
2. (7.5%) Discuss the impact of different learning rates and iterations on the convergence of logistic regression. How does hyperparameter tuning affect performance?
   ***Provide the results of ROC curves for different hyperparameters.**

3. (7.5%) Discuss the **activation functions: sigmoid** for logistic. How do the activation functions influence decision boundaries?
4. (7.5%) Define and explain the significance of Confusion Matrices, ROC Curves, and AUC in evaluating classification models. How do they contribute to model selection?

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. Name your files correctly:
   a. Report: StudentID_Lab2.pdf
   b. Code: StudentID_Lab2.py or StudentID_Lab2.ipynb
4. 1 day late: 10% deduction from total score.
5. Plagiarism is **strictly prohibited**. Submitting copied work from other students will result in penalties.

Code Results and Answer:

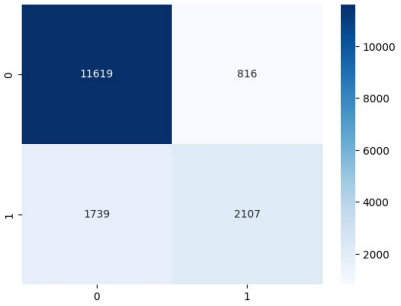
A1:

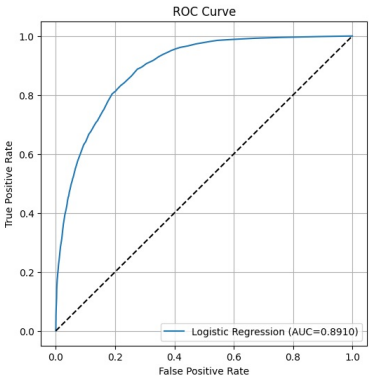
Parameters: learning_rate=0.01, iterations=3000, threshold=0.5
False Positives(model predict 1, but actually 0): 816
False Negatives(model predict 0, but actually 1): 1739
There are more FP than FN, maybe because:

Class imbalance:

```
y_train distribution:
class 0: 24720 samples
class 1: 7841 samples

y_test distribution:
class 0: 12435 samples
class 1: 3846 samples
```

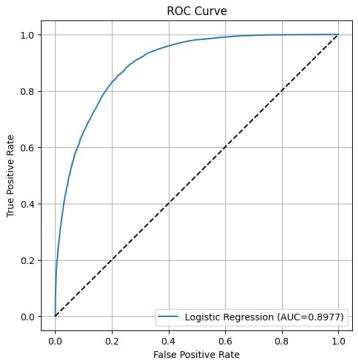


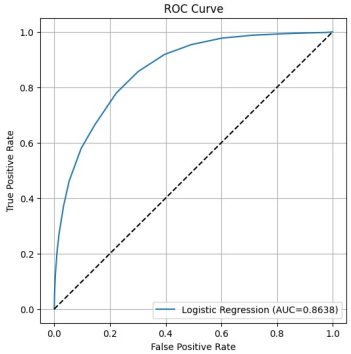


A2:

learning_rate=0.1, iterations=3000, threshold=0.5 -> AUC 0.897
learning_rate=0.0001, iterations=3000, threshold=0.5 -> AUC 0.863

With a larger learning rate, the model tends to capture the data distribution at a faster pace, although it may risk overshooting the optimal solution.





A3: Sigmoid can transform model output to probability, then we can set a threshold to decide which class to assign.

Sigmoid outputs probabilities, but the decision boundary remains linear because it's based on the linear equation
w^T x + b = 0.

A4:

Confusion Matrix:
True Positives (TP): correctly predicted positive cases
True Negatives (TN): correctly predicted negative cases
False Positives (FP): predicted positive but actually negative
False Negatives (FN): predicted negative but actually positive

It helps calculate key performance metrics such as accuracy, precision, recall, and F1-score. These metrics reflect different aspects of model performance and are useful for comparing models depending on the application context (e.g., medical diagnosis may prefer high recall).

The ROC curve plots the True Positive Rate against the False Positive Rate at different classification thresholds. It shows the trade-off between sensitivity and specificity. A better model will have a curve that rises steeply toward the top-left corner.

Higher AUC means the model is better at ranking positive samples higher than negative ones. It ranges from 0 to 1:

- AUC = 1.0: perfect classifier
- AUC = 0.5: random guessing

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