Lecture 1 Notes： Linear Regression Logistic Regression

“Machine Learning and Deep Learning” Lab Course (Lecture 1)

—— Linear Regression Model (PyTorch Implementation + HuggingFace Datasets)

I. Equipment and Materials

Students should prepare:

1. Personal computer (Windows / macOS / Linux)
2. Python 3.10+
3. PyTorch (CPU or GPU)
4. Jupyter Notebook or VSCode
5. HuggingFace Datasets library (for loading datasets)
6. Two datasets must be completed in this experiment:
    - **California Housing (HuggingFace dataset)**
    - **House Prices (Kaggle) → use HuggingFace mirror version, no Kaggle account needed**

II. Experimental Principles

(1) Linear Regression Model

Linear regression fits linear relationships in data:

$\hat{y} = w^\top x + b$

Mean Squared Error (MSE) measures the fitting quality:

$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$

Training uses **PyTorch autograd + optimizers**.

---

(2) Experimental Datasets

① California Housing (HuggingFace Official Dataset)

HuggingFace link:

https://huggingface.co/datasets/scikit-learn/california-housing

> The dataset contains 8 features for predicting median house prices.

---

② House Prices

No Kaggle account is required.

HuggingFace link:

https://huggingface.co/datasets/stanfordaka/house-prices

Students must complete:

- Missing value handling
- Numerical feature extraction
- Categorical feature encoding (OneHot or Embedding, choose one)
- Convert to PyTorch Tensors

III. Learning Objectives

Students must master:

(1) Mathematical principles of linear regression

- Model expression
- Loss function (MSE)
- Gradient descent principles
- Feature dimension, normalization, and model stability

(2) End-to-end implementation of linear regression using PyTorch

Includes:

- Data loading (HuggingFace)
- Data preprocessing (NumPy / Pandas)
- Model construction (nn.Linear)

- Loss function (nn.MSELoss)
- Optimizer (torch.optim.SGD / Adam)
- Training loop (forward → loss → backward → update)
- Loss curve plotting
- Model performance evaluation (MSE / RMSE)

(3) Analysis and prediction based on two datasets

Must complete:

- California Housing: full linear regression training
- House Prices: including missing value handling, feature engineering, and training

---

IV. Experimental Tasks

Task A: California Housing Linear Regression (Basic)

Students are required to:

1. Load data using HuggingFace
2. Standardize data using NumPy / PyTorch
3. Build the linear model: model = torch.nn.Linear(in_features=8, out_features=1)
4. Use MSE Loss
5. Use SGD or Adam optimizer
6. Write complete training code
7. Plot training Loss curve
8. Evaluate RMSE on the test set
9. Provide simple analysis: which features might be important?

Task B: House Prices Kaggle Dataset

Students are required to:

1. Load train.csv via HuggingFace
2. Handle missing values
3. Choose one preprocessing method:
    - **Option 1: OneHot encode all categorical features**
    - **Option 2: Build an Embedding for each categorical feature**
4. Build a PyTorch linear regression model
5. Complete full training pipeline (forward/backward/update)

6. Evaluate RMSE on training/validation sets
7. Write an error analysis
8. (Optional) Visualize the top 10 most important features

Task C: Titanic (Classification, Basic, Logistic Regression)

Students must complete:

1. Load Titanic dataset using HuggingFace
2. Data cleaning:
   - Missing values (age, cabin)
   - Categorical features (sex, passenger class) → OneHot
3. Build a PyTorch logistic regression model
4. Use BCE loss
5. Train & plot Loss curve
6. Evaluate on validation set:
   - Accuracy
   - Precision
   - Recall
   - F1 score
7. Provide simple analysis of important features

Task D: SMS Spam (Classification, Advanced, Logistic Regression + Text)

Students must complete:

1. Load the dataset (Ham/Spam)
2. Text preprocessing:
   - Cleaning (lowercase, remove punctuation, remove stop words)
   - Build Bag-of-Words or TF-IDF (CountVectorizer or HuggingFace tokenizer)
3. Convert to PyTorch Tensors
4. Build logistic regression model
5. Use BCEWithLogitsLoss
6. Train & plot Loss curve
7. Output Accuracy / F1
8. Analyze which words contribute the most to “spam”