NYCU Introduction to Machine Learning, Homework 2

111550129, 林彥亨

Part. 1, Coding (60%):

(25%) Logistic Regression w/ Gradient Descent Method

1. (5%) Show the hyperparameters (learning rate and iteration, etc) that you used and the weights and intercept of your model.

```
LR = LogisticRegression(
    learning_rate=0.05,
    num_iterations=1500,
)
```

```
2025-10-22 11:15:47.694 | INFO     | __main__:main:252 - LR: Weights: [
-0.41421757  0.10320723  0.49694739  0.09949418  0.14894617], Intercep:
 -1.549974448695558
```

2. (5%) Show the AUC of the classification results on the testing set.

```
2025-10-22 11:15:47.694 | INFO     | __main__:main:253 -
LR: Accuracy=0.8333, AUC=0.8500
```

3. (15%) Show the accuracy score of your model on the testing set

```
2025-10-22 11:15:47.694 | INFO     | __main__:main:253 -
LR: Accuracy=0.8333, AUC=0.8500
```

(25%) Fisher Linear Discriminant, FLD

4. (5%) Show the mean vectors m_i (i=0, 1) of each class, the within-class scatter matrix *Sw,* and the between-class scatter matrix *Sb* of the training set.

```
2025-10-22 11:15:47.702 | INFO     | __main__:main:275 - FLD:
 m0=[ 0.35994138 -0.04560139], m1=[0.32519126 0.04435118] of
cols=['27', '30']
2025-10-22 11:15:47.703 | INFO     | __main__:main:276 - FLD:

Sw=
[[41.93041055 15.7202037 ]
 [15.7202037  37.25186904]]
2025-10-22 11:15:47.703 | INFO     | __main__:main:277 - FLD:

Sb=
[[ 0.00120757 -0.00312586]
 [-0.00312586  0.00809147]]
```

5. (5%) Show the Fisher's linear discriminant *w* of the training set.

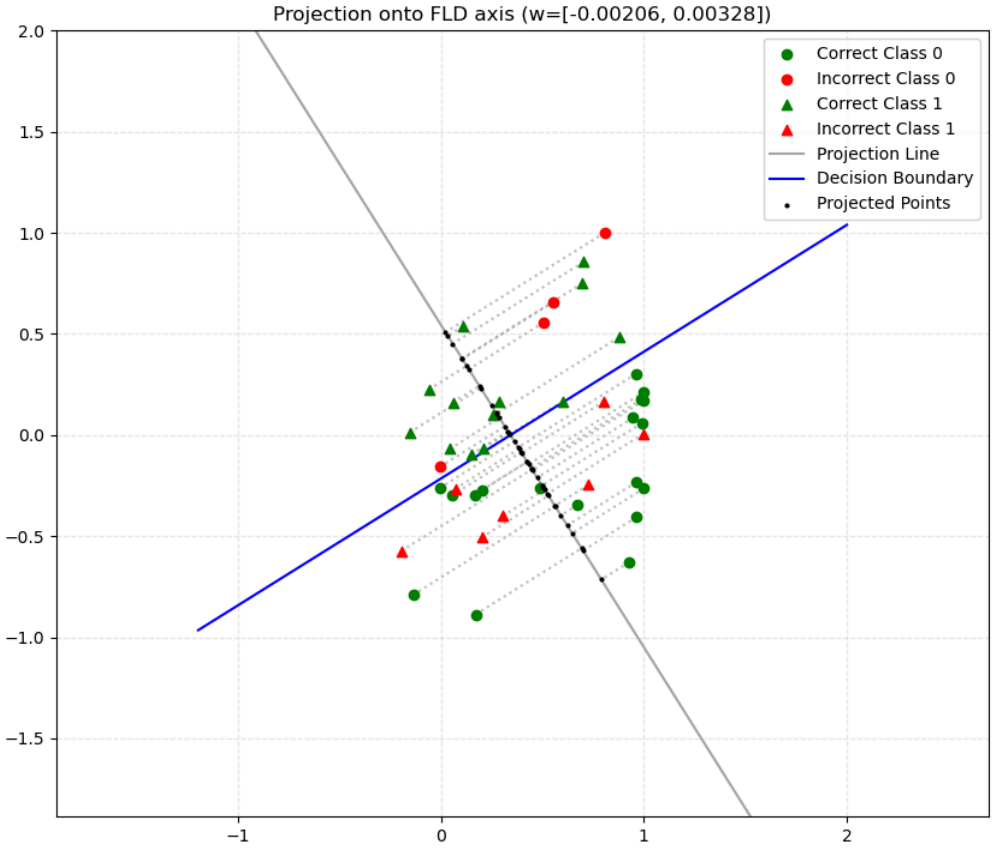
```
2025-10-22 11:15:47.703 | INFO     | __main__:main:278 - FLD:

w=
[-0.00205997  0.00328402]
```

6. (15%) Show the accuracy score on the testing set. Also, plot/obtain predictions for the testing set by measuring the distance between the projected value of the testing data

and the projected means of the training data for the two classes (**Please check the slide for color, shape, and other plotting requirements**).

```
2025-10-22 11:15:47.703 | INFO     | __main__:main:279 - FLD:
 Accuracy=0.7381
```



(10%) Code Check and Verification

7. (10%) Lint the code and show the PyTest results.

```
(ml) heng@heng:~/Desktop/ml_hw2$ flake8 main.py
(ml) heng@heng:~/Desktop/ml_hw2$
```

```
(ml) heng@heng:~/Desktop/ml_hw2$ pytest ./test_main.py -s
============================= test session starts ==============================
platform linux -- Python 3.11.13, pytest-8.4.2, pluggy-1.6.0
rootdir: /home/heng/Desktop/ml_hw2
plugins: anyio-4.11.0
collected 2 items

test_main.py (395, 2) (395,)
Iteration 0: Loss= 0.7181
Iteration 3000: Loss= 0.2379
Iteration 6000: Loss= 0.1920
Iteration 9000: Loss= 0.1710
Iteration 12000: Loss= 0.1581
Iteration 15000: Loss= 0.1491
Iteration 18000: Loss= 0.1422
Iteration 21000: Loss= 0.1366
Iteration 24000: Loss= 0.1320
Iteration 27000: Loss= 0.1281
2025-10-22 10:44:36.016 | INFO     | test_main:test_logistic_regression:35 - accuracy=0.9517
.(395, 2) (395,)
2025-10-22 10:44:36.024 | INFO     | test_main:test_fld:45 - accuracy=0.8759
.

============================== 2 passed in 3.52s ===============================
```

Part. 2, Questions (40%):

1. (15%)

(⋆) Using (4.57) and (4.58), derive the result (4.65) for the posterior class probability in the two-class generative model with Gaussian densities, and verify the results (4.66) and (4.67) for the parameters $\mathbf{w}$ and w_0.

$$\begin{aligned} p(\mathcal{C}_1|\mathbf{x}) &= \frac{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1) + p(\mathbf{x}|\mathcal{C}_2)p(\mathcal{C}_2)} \\ &= \frac{1}{1+\exp(-a)} = \sigma(a) \end{aligned} \tag{4.57}$$

$$a = \ln \frac{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x}|\mathcal{C}_2)p(\mathcal{C}_2)} \tag{4.58}$$

$$p(\mathcal{C}_1|\mathbf{x}) = \sigma(\mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0) \tag{4.65}$$

$$\mathbf{w} = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) \tag{4.66}$$

$$w_0 = -\frac{1}{2}\boldsymbol{\mu}_1^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_2^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_2 + \ln \frac{p(\mathcal{C}_1)}{p(\mathcal{C}_2)}. \tag{4.67}$$

Ans:

(1) Assumptions and Starting Point

We begin with the definition of the posterior probability (4.57) and the log-odds a(4.58):

$$p(C_1|x) = \frac{1}{1+exp(-a)} = \sigma(a) \qquad (4.57)$$

$$a = ln\frac{p(x|C_1)p(C_1)}{p(x|C_2)p(C_2)} \qquad (4.58)$$

We assume the class-conditional densities are multivariate Gaussian distributions with class-specific means μ_k but a shared covariance matrix Σ:

$$p(x|C_k) = N(x|\mu_k, \Sigma) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} exp(\frac{-1}{2}(x-\mu_k)^T\Sigma^{-1}(x-\mu_k))$$

(2) Rewrite a

Using the properties of logarithms, we can rewrite **a** from (4.58) as:

$$\begin{aligned} a &= ln\frac{p(x|C_1)p(C_1)}{p(x|C_2)p(C_2)} = ln[p(x|C_1)p(C_1)] - ln[p(x|C_2)p(C_2)] \\ &= ln\,p(x|C_1) + ln\,p(C_1) - ln\,p(C_1) - ln\,p(x|C_2) - ln\,p(C_2) \\ &= [ln\,p(x|C_1) - ln\,p(x|C_2)] + ln\,\frac{p(C_1)}{p(C_2)} \end{aligned}$$

Now, we take the natural logarithm of the Gaussian density:

$$ln\, p(x|C_k) = ln[\frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}}] - \frac{1}{2}(x-\mu_k)^T\Sigma^{-1}(x-\mu_k)$$

Let $C = ln[(2\pi)^{-D/2}|\Sigma|^{-1/2}]$. Since the covariance Σ is shared, this normalization term C is identical for both classes.

(3) Simplifying the Log-Likelihood Difference

We compute the difference in the log-likelihoods:

$$\begin{aligned}
&ln\, p(x|C_1) - ln\, p(x|C_2) \\
&= [C - \frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1)] - [C - \frac{1}{2}(x-\mu_2)^T\Sigma^{-1}(x-\mu_2)] \\
&= [\frac{-1}{2}(x^T-\mu_1{}^T)\Sigma^{-1}(x-\mu_1)] + [\frac{1}{2}(x^T-\mu_2{}^T)\Sigma^{-1}(x-\mu_2)] \\
&= [\frac{-1}{2}(x^T\Sigma^{-1}x - x^T\Sigma^{-1}\mu_2 - \mu_2{}^T\Sigma^{-1}x + \mu_2{}^T\Sigma^{-1}\mu_2)] + \\
&[\frac{1}{2}(x^T\Sigma^{-1}x - x^T\Sigma^{-1}\mu_2 - \mu_2{}^T\Sigma^{-1}x + \mu_2{}^T\Sigma^{-1}\mu_2)] \\
&= [\frac{-1}{2}(x^T\Sigma^{-1}x - 2x^T\Sigma^{-1}\mu_1 + \mu_1{}^T\Sigma^{-1}\mu_1)] + \\
&[\frac{-1}{2}(x^T\,\Sigma^{-1}x - 2x^T\Sigma^{-1}\mu_2 + \mu_2{}^T\Sigma^{-1}\mu_2)] \\
&= (x^T\Sigma^{-1}\mu_1 - x^T\Sigma^{-1}\mu_2) - \frac{1}{2}\mu_1{}^T\Sigma^{-1}\mu_1 + \frac{1}{2}\mu_2{}^T\Sigma^{-1}\mu_2 \\
&= x^T\Sigma^{-1}(\mu_1-\mu_2) - \frac{1}{2}\mu_1{}^T\Sigma^{-1}\mu_1 + \frac{1}{2}\mu_2{}^T\Sigma^{-1}\mu_2
\end{aligned}$$

(4) Verifying W and w_0

$$w = \Sigma^{-1}(\mu_1-\mu_2) \qquad (4.66)$$

$$w_0 = \frac{-1}{2}\mu_1{}^T\Sigma^{-1}\mu_1 + \frac{1}{2}\mu_2{}^T\Sigma^{-1}\mu_2 + ln\,\frac{p(C_1)}{p(C_2)} \qquad (4.67)$$

Now we substitute this result back into the full expression for a:

$$a = [x^T\Sigma^{-1}(\mu_1-\mu_2)] - \frac{1}{2}\mu_1{}^T\Sigma^{-1}\mu_1 + \frac{1}{2}\mu_2{}^T\Sigma^{-1}\mu_2 + ln\,\frac{p(C_1)}{p(C_2)}$$

We want to show this is of the form $a = w^Tx + w_0$.

The term $[x^T\Sigma^{-1}(\mu_1-\mu_2)]$ is a scalar, so it is equal to its own transpose:

$$x^T\Sigma^{-1}(\mu_1-\mu_2) = (x^T\Sigma^{-1}(\mu_1-\mu_2))^T = (\mu_1-\mu_2)^T(\Sigma^{-1})^Tx = (\mu_1-\mu_2)^T(\Sigma^{-1})x = w^Tx$$

So, we can write a as:

$$a = [x^T\Sigma^{-1}(\mu_1-\mu_2)] - \frac{1}{2}\mu_1{}^T\Sigma^{-1}\mu_1 + \frac{1}{2}\mu_2{}^T\Sigma^{-1}\mu_2 + ln\,\frac{p(C_1)}{p(C_2)}$$

$= (w^T x + w_0)$

(5) Conclusion

We have successfully shown that the log-odds a (4.58) is a linear function of x, namely $a = w^T x + w_0$, where w and w_0 are given by (4.66) and (4.67).

Substituting this result for a back into the posterior probability equation (4.57), $p(C_1|x) = \sigma(a)$, we immediately obtain:

$$p(C_1|x) = \sigma(w^T x + w_0) \qquad (4.65).$$

2. (10%)

(a) Give one real-world situation where you would prefer Logistic Regression (LR) over the Perceptron, and explain why.

(b) Is Logistic Regression actually used for regression (predicting a continuous value)? If not, state what task it really solves and why the name includes “regression.”

Ans:

(a) Predicting the probability of a customer defaulting on a loan. The Logistic Regression provides a continuous output between 0 and 1, interpreted as a probability. Since a bank needs to know the likelihood of default (e.g., 5% vs. 45%), not just a simple "Default/No Default" answer.

(b) No, Logistic Regression is used for binary classification. The term "regression" is included because the model's fundamental structure involves estimating the parameters weight and bias of that initial linear regression to best fit the data.

3. (15%)

(a) Why is feature scaling (e.g., standardization or normalization) important in Logistic Regression? Explain two reasons.

(b) If feature scaling is not applied in Logistic Regression, list three problems that may occur. Briefly explain.

Ans:

(a)

1. Make gradient descent converge faster.
   -> Make all features have a similar range (mean ≈ 0, standard deviation ≈ 1), resulting in a smoother, more symmetric cost surface and faster convergence.

2. Ensures all features contribute fairly to the model's prediction.
   -> Each variable contributes proportionally to the decision boundary, allowing the model to learn meaningful and comparable weights across all features.

(b)

1. Slow or Unstable Convergence -> When features have very different scales, the cost function becomes highly skewed.
2. Biased Weight Magnitudes -> The model uses regularization to prevent overfitting, unscaled features lead to a distorted penalty.
3. Poor Regularization Performance -> The unscaled features can lead to the model effectively ignoring features with small magnitudes.