

NYCU Introduction to Machine Learning, Homework 2

110550168, 賴御安

Part. 1, Coding (50%):

In this coding assignment, you are requested to implement Logistic Regression and Fisher's Linear Discriminant by using only Numpy. After that, train your model on the provided dataset and evaluate the performance on the testing data.

(15%) Logistic Regression

Requirements:

- Use Gradient Descent to update your model
- Use CE ([Cross-Entropy](#)) as your loss function.

Criteria:

1. (0%) Show the hyperparameters (learning rate and iteration) that you used.

```
LR = LogisticRegression(learning_rate=0.0002, iteration=300000)
```

2. (5%) Show the weights and intercept of your model.

```
Weights: [-0.06302108 -1.53906823 1.02023911 0.08383513 0.03121795 -0.67974996], Intercept: 0.735009803911418
```

3. (10%) Show the accuracy score of your model on the testing set. The accuracy score should be greater than 0.75.

```
Accuracy: 0.7540983606557377
```

(35%) Fisher's Linear Discriminant (FLD)

Requirements:

- Implement FLD to reduce the dimension of the data from 2-dimensional to 1-dimensional.

Criteria:

4. (0%) Show the mean vectors m_i ($i=0, 1$) of each class of the training set.

```
Class Mean 0: [[ 56.75925926 137.7962963 ]], Class Mean 1: [[ 52.63432836 158.97761194]]
```

5. (5%) Show the within-class scatter matrix S_W of the training set.

```
With-in class scatter matrix:  
[[ 19184.82283029 -16006.39331122]  
 [-16006.39331122 106946.45135434]]
```

6. (5%) Show the between-class scatter matrix S_B of the training set.

```
Between class scatter matrix:  
[[ 17.01505494 -87.37146342]  
 [-87.37146342 448.64813241]]
```

7. (5%) Show the Fisher's linear discriminant w of the training set.

```
w :  
[[-0.28737344  0.95781862]]
```

8. (10%) 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. Show the accuracy score on the testing set. The accuracy score should be greater than 0.65.

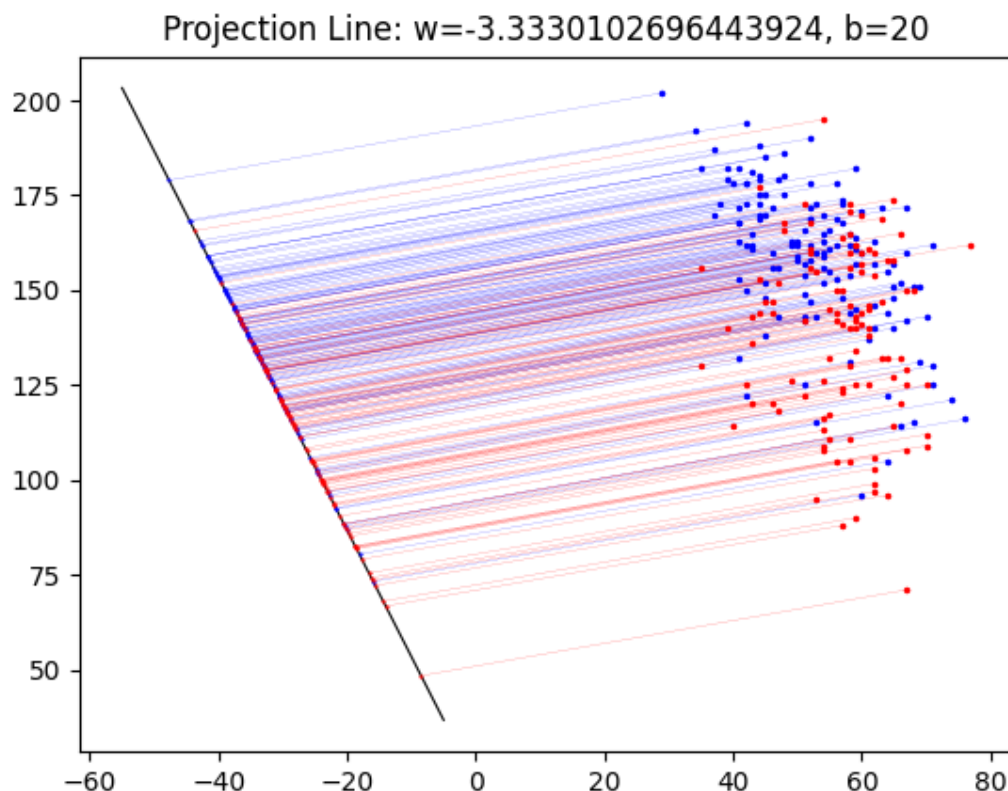
```
Accuracy of FLD: 0.6557377049180327
```

9. (10%) Plot the projection line (x-axis: age, y-axis: thalach).

1) Plot the projection line trained on the training set and show the slope and intercept on the title (you can choose any value of intercept for better visualization).

2) Obtain the prediction of the testing set, plot and colorize them based on the prediction.

3) Project all testing data points on your projection line. Your result should look like the below image.



Part. 2, Questions (50%):

1. (5%) What's the difference between the sigmoid function and the softmax function?
In what scenarios will the two functions be used? Please at least provide one difference for the first question and answer the second question respectively.

Differences :

Sigmoid function : makes the input become in the range between 0 and 1 but the probabilities sum does not need to be 1

Softmax function : it sum up all of the values as the denominator, and the numerator will be each value, so that promise that the probability will sum up to 1

In what scenarios will the two functions be used :

Sigmoid function : used in models to classify data into two different categories (this does not require the probabilities sum to be 1)

Softmax function : used when dealing with multi-class classification problems, and can balance each class

2. (10%) In this homework, we use the cross-entropy function as the loss function for Logistic Regression. Why can't we use Mean Square Error (MSE) instead? Please explain in detail.

Since the result of logistic regression is 0 or 1, and MSE calculates the error value by getting the difference squared up, it will be harder to track the error value. While at the same time, the cross-entropy function considers the error value with log, which means that the error value grows sharply if wrong prediction happened.

3. (15%) In a multi-class classification problem, assume you have already trained a classifier using a logistic regression model, which the outputs are P_1, P_2, \dots, P_c , how do you evaluate the overall performance of this classifier with respect to its ability to predict the correct class?
 - 3.1. (5%) What are the metrics that are commonly used to evaluate the performance of the classifier? Please at least list three of them.
 - 3.2. (5%) Based on the previous question, how do you determine the predicted class of each sample?
 - 3.3. (5%) In a class imbalance dataset (say 90% of class-1, 9% of class-2, and 1% of class-3), is there any problem with using the metrics you mentioned above and how to evaluate the model prediction performance in a fair manner?

3.1. A:

(1) Accuracy

(2) Precision and Recall with Micro-averaging

(3) Precision and Recall with Macro-averaging

3.1. B:

- (1) Accuracy : correct predictions / all predictions
- (2) Precision and Recall with Micro-averaging : sum up all TP, FP/FN as as the denominator, and the sum of TP is the numerator

$$\text{Precision}_{\text{Micro-average}} = \frac{TP_A + TP_B + \dots TP_N}{TP_A + FP_A + TP_B + FP_B + \dots TP_N + FP_N}$$

$$\text{Recall}_{\text{Micro-average}} = \frac{TP_A + TP_B + \dots TP_N}{TP_A + FN_A + TP_B + FN_B + \dots TP_N + FN_N}$$

if there is N classes

TP : true and predict true

FP : false and predict true

FN : true and predict false

- (3) Precision and Recall with Macro-averaging : separated calculate the precision/recall value and take average

$$\text{Precision}_{\text{Macro-average}} = \frac{\text{Precision}_{\text{Class A}} + \text{Precision}_{\text{Class B}} + \dots \text{Precision}_{\text{Class N}}}{N}$$

$$\text{Recall}_{\text{Macro-average}} = \frac{\text{Recall}_{\text{Class A}} + \text{Recall}_{\text{Class B}} + \dots \text{Recall}_{\text{Class N}}}{N}$$

3.1. C:

- (1) Accuracy : always classify the result to class-1 can often lead to high accuracy, but this is not what we want
- (2) Precision and Recall with Micro-averaging : perform well
- (3) Precision and Recall with Macro-averaging : since it balance each class's contribution to final result, if small classes(class-2, class-3) got low precision, then it may affect the result to much, although that is not the real result

To evaluate the model prediction performance, I would say that using data from the confusion matrix, which is TP, TN, FP, FN will be a fair manner. It contains more information of the true/false and right/wrong prediction. Also, Micro-averaging is a good way, which balances every event.

4. (20%) Calculate the results of the partial derivatives for the following equations. (The first one is binary cross-entropy loss, and the second one is mean square error loss followed by a sigmoid function. σ is the sigmoid function.)

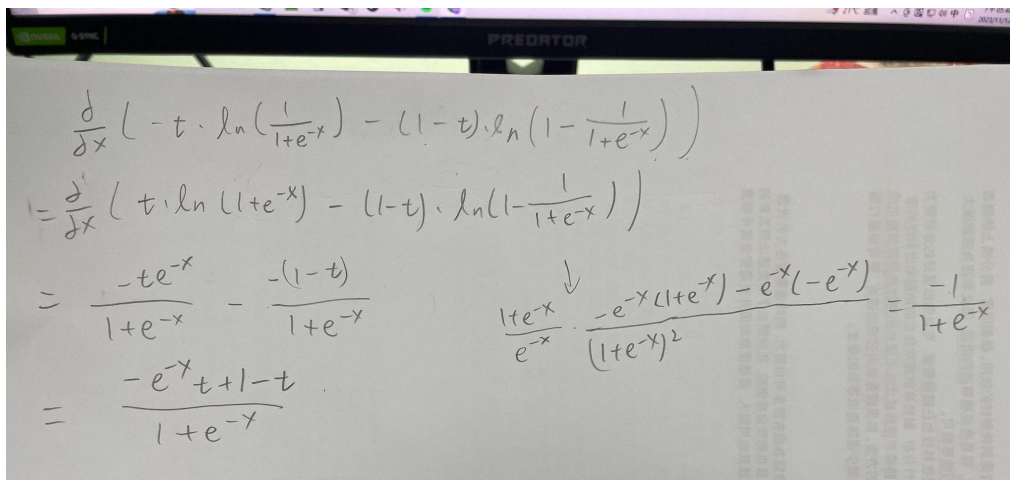
4.1. (10%)

$$\frac{\partial}{\partial x} (-t * \ln(\sigma(x)) - (1 - t) * \ln(1 - \sigma(x)))$$

4.2. (10%)

$$\frac{\partial}{\partial x} ((t - \sigma(x))^2)$$

4.1.A:



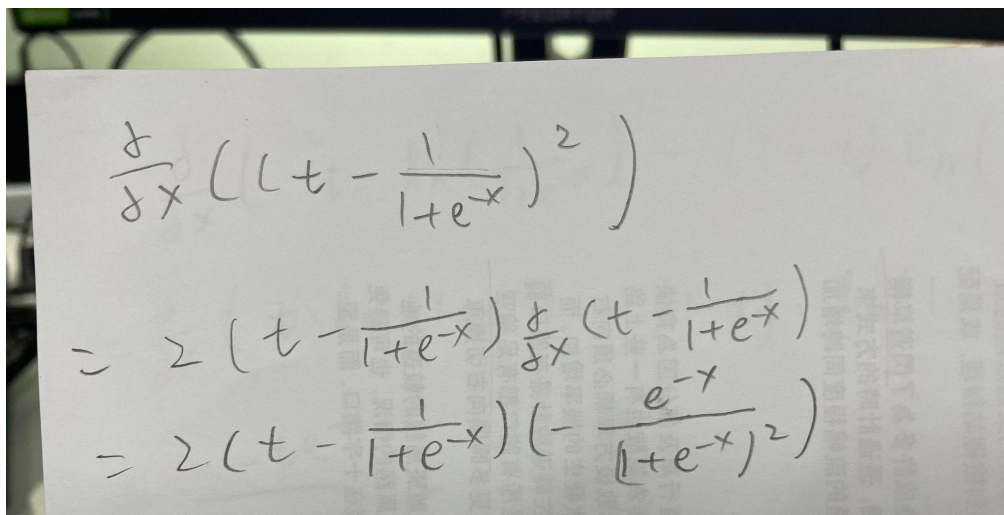
Handwritten solution for 4.1.A:

$$\begin{aligned} & \frac{\partial}{\partial x} \left(-t \cdot \ln\left(\frac{1}{1+e^{-x}}\right) - (1-t) \cdot \ln\left(1 - \frac{1}{1+e^{-x}}\right) \right) \\ &= \frac{\partial}{\partial x} \left(t \cdot \ln(1+e^{-x}) - (1-t) \cdot \ln\left(1 - \frac{1}{1+e^{-x}}\right) \right) \\ &= \frac{-te^{-x}}{1+e^{-x}} - \frac{-(1-t)}{1+e^{-x}} \\ &= \frac{-e^{-x}t + 1 - t}{1+e^{-x}} \end{aligned}$$

Intermediate step shown:

$$\frac{1+e^{-x}}{e^{-x}} \cdot \frac{-e^{-x}(1+e^{-x}) - e^{-x}(-e^{-x})}{(1+e^{-x})^2} = \frac{-1}{1+e^{-x}}$$

4.2.A:



Handwritten solution for 4.2.A:

$$\begin{aligned} & \frac{\partial}{\partial x} \left(\left(t - \frac{1}{1+e^{-x}} \right)^2 \right) \\ &= 2 \left(t - \frac{1}{1+e^{-x}} \right) \frac{\partial}{\partial x} \left(t - \frac{1}{1+e^{-x}} \right) \\ &= 2 \left(t - \frac{1}{1+e^{-x}} \right) \left(-\frac{e^{-x}}{(1+e^{-x})^2} \right) \end{aligned}$$