NYCU Introduction to Machine Learning, Homework 2

109350008, 張詠哲

Deadline: Nov. 14, 23:59

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.

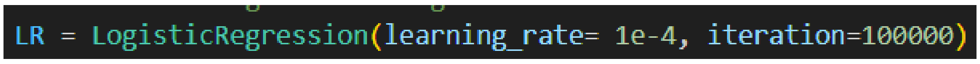
(15%) Logistic Regression

Requirements:

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

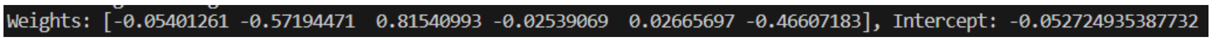
Criteria:

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



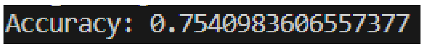
```
LR = LogisticRegression(learning_rate= 1e-4, iteration=100000)
```

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



```
Weights: [-0.05401261 -0.57194471  0.81540993 -0.02539069  0.02665697 -0.46607183], Intercept: -0.052724935387732
```

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
```

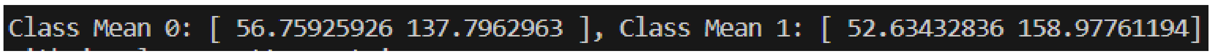
(35%) Fisher’ s Linear Discriminant (FLD)

Requirements:

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

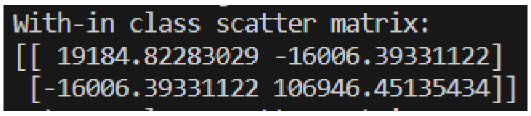
Criteria:

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



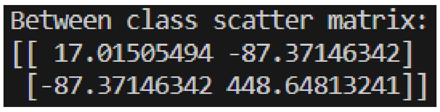
```
Class Mean 0: [ 56.75925926 137.7962963 ], Class Mean 1: [ 52.63432836 158.97761194]
```

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



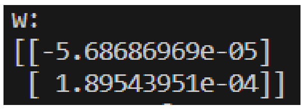
```
With-in class scatter matrix:
[[ 19184.82283029 -16006.39331122]
 [-16006.39331122 106946.45135434]]
```

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



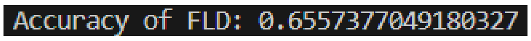
```
Between class scatter matrix:
[[ 17.01505494 -87.37146342]
 [-87.37146342 448.64813241]]
```

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



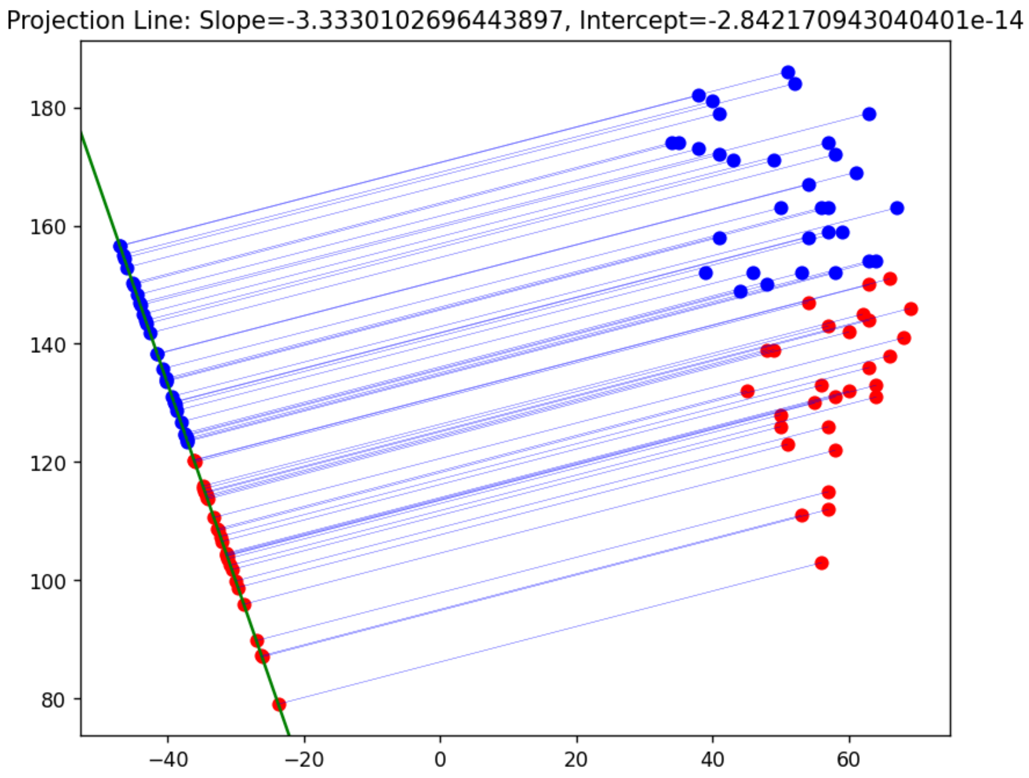
```
w:
[[-5.68686969e-05]
 [ 1.89543951e-04]]
```

8. (10%) Obtain predictions for the testing set by measuring the distance between the project ed value of the testing data and the projected means of the training data for the two classe s. 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 th e 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 b elow 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 th e first question and answer the second question respectively.

- **Difference between Sigmoid and Softmax functions:**
  - **Output range:**

    Sigmoid function:

    It is used for binary classification problems and maps the input to a range betwe en 0 and 1. It's typically used for problems where you have two classes (0 and 1).

    Softmax function:

    It is used for multi-class classification problems and maps the input to a probabi lity distribution over multiple classes. The output values of the softmax function sum up to 1, making it suitable for problems with more than two classes.

- **Scenarios for using Sigmoid and Softmax functions:**
  - **Sigmoid Function:**

  Binary Classification:

  The sigmoid function is commonly used in binary classification problems, where you have two possible outcomes (e.g., spam detection, sentiment analysis, whether an email is spam or not).

  - **Softmax Function:**

  Multi-Class Classification:

  The softmax function is used when you have more than two classes and you want to assign a probability to each class. This is common in scenarios like image classification, w here an image can belong to one of many possible categories (e.g., recognizing different an imals in a picture).

  Neural Networks:

  The softmax function is frequently used in the output layer of neural networks for mu lti-class classification tasks.

2. **(10%) In this homework, we use the cross-entropy function as the loss function for Logisti c Regression. Why can’ t we use Mean Square Error (MSE) instead? Please explain in de tail.**

- **Nature of the Problem:**

  Logistic Regression is primarily used for binary classification, where the target variab le is binary (0 or 1). MSE is typically used for regression problems where the target variab le is continuous.

- **Output Range:**

  Cross-Entropy is designed to measure the dissimilarity between the predicted probabi lities and the true binary labels (0 or 1). It ensures that the predicted probabilities are betw een 0 and 1, which is appropriate for a classification problem.

  MSE, on the other hand, measures the average squared difference between predicted and actual values. In the case of binary classification, you are essentially treating the probl em as a regression task, and the output range may not be restricted to [0, 1], which can lea d to numerical instability and poor performance.

- **Sensitivity to Outliers:**

  MSE can be sensitive to outliers, as it squares the errors. In a binary classification pro blem, a single misclassified point with a large error can significantly impact the loss, leadi ng to unstable training and suboptimal models.

  Cross-Entropy is less sensitive to outliers because it primarily penalizes the model for making incorrect predictions. It focuses on the probability distribution and is less influence d by the magnitude of errors.

- **Logarithmic Nature:**

  Cross-Entropy has a logarithmic nature, which is suitable for measuring the divergen ce between probability distributions. It aligns well with the underlying probabilistic interpr etation of logistic regression.

- **Better Handling of Imbalanced Data:**

  Cross-Entropy can better handle imbalanced datasets, as it penalizes the model more for misclassifying the minority class. It encourages the model to correctly classify the min ority class, which is often more critical in many real-world applications.

3. (15%) In a multi-class classification problem, assume you have already trained a classifier using a logistic regression model, which the outputs are P1, P2, ... Pc, how do you evaluat e the overall performance of this classifier with respect to its ability to predict the correct c lass?

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.

There are some commonly used metrics:

- **Accuracy:**

  Accuracy is a straightforward metric that measures the proportion of correctly classifi ed instances out of the total number of instances. While it is easy to understand, it ma y not be suitable for imbalanced datasets.

- Precision:
  Measures the proportion of true positive predictions among all instances predicted as positive. It assesses the classifier's ability to make accurate positive predictions.

- F1-Score:
  It is the harmonic mean of precision and recall and provides a balance between the tw o. It's useful when you want to consider both false positives and false negatives.

- Confusion Matrix:
  A confusion matrix provides a comprehensive view of the classifier's performance by showing the number of true positives, true negatives, false positives, and false negati ves for each class.

3.2. (5%) Based on the previous question, how do you determine the predicted class of each sample?

To determine the predicted class for each sample in a multi-class classification pro blem, you can choose the class with the highest predicted probability. In logistic re gression, the model outputs probabilities for each class, such as P1, P2, ..., Pc. The class with the highest probability is considered the predicted class for a given samp le. It can be expressed as:
Predicted Class = argmax(P1, P2, ..., Pc)

3.3. (5%) In a class imbalance dataset (say 90% of class-1, 9% of class-2, and 1% of cl ass-3), is there any problem with using the metrics you mentioned above and how to evaluate the model prediction performance in a fair manner?

- Accuracy:
  Accuracy is not a suitable metric for imbalanced datasets, especially when one class s ignificantly outweighs the others. In the scenario you mentioned (90% class-1, 9% cl ass-2, and 1% class-3), a model that simply predicts class-1 for every instance would achieve a high accuracy of 90%. However, this doesn't indicate good model performa nce because it fails to capture the minority classes.

- Precision:
  Precision is the ratio of true positive predictions to the total predicted positive instanc es. In an imbalanced dataset, a high precision can be achieved by avoiding false posit

ives, but it may come at the expense of high false negatives. For minority classes, pre cision can be artificially inflated at the cost of recall.

- F1-Score:
  The F1-score is the harmonic mean of precision and recall. Similar to precision, it ca n be biased in favor of the majority class because of the imbalance. It may not provid e a fair assessment of the model's performance for the minority classes.

- Confusion Matrix:
  While the confusion matrix provides detailed information about true positives, true ne gatives, false positives, and false negatives, it doesn't inherently address the imbalanc e issue. it can identify where the model is making errors, but it doesn't automatically balance the evaluation.

To evaluate the model prediction performance in a fair manner in the presence of c lass imbalance, there are some alternatives:

- Precision-Recall Curve:
  Plotting the precision-recall curve allows you to assess the trade-off between precisio n and recall at different classification thresholds. It is particularly useful for imbalanc ed datasets because it focuses on the performance of positive (minority) classes.

- Area Under the Receiver Operating Characteristic (ROC-AUC):
  ROC AUC measures the model's ability to distinguish between positive and negative instances, regardless of class imbalance. It can provide a more balanced view of mod el performance.

- Weighted Metrics:
  Weighted versions of precision, recall, and F1-score can be used to give more import ance to minority classes. These metrics assign different weights to each class based o n their prevalence.

- Macro-Averaging and Micro-Averaging:
  Macro-averaging and micro-averaging of precision, recall, and F1-score provide a mo re comprehensive evaluation by considering overall performance across all classes, n ot just focusing on the majority class.

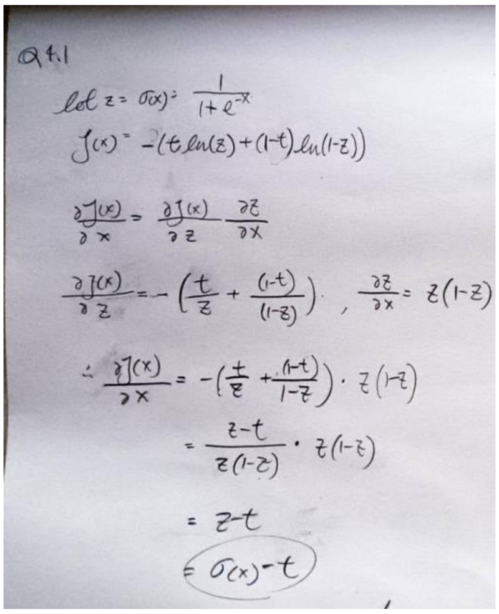
- Sampling Techniques:

Resampling techniques such as oversampling the minority class or undersampling the majority class to balance the dataset before evaluation.

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%)

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



Q4.1

let $z = \sigma(x) = \frac{1}{1+e^{-x}}$

$J(x) = -(t\,ln(z) + (1-t)\,ln(1-z))$

$\frac{\partial J(x)}{\partial x} = \frac{\partial J(x)}{\partial z}\frac{\partial z}{\partial x}$

$\frac{\partial J(x)}{\partial z} = -\left(\frac{t}{z} + \frac{(1-t)}{(1-z)}\right)$, $\frac{\partial z}{\partial x} = z(1-z)$

$\therefore \frac{\partial J(x)}{\partial x} = -\left(\frac{t}{z} + \frac{(1-t)}{1-z}\right) \cdot z(1-z)$

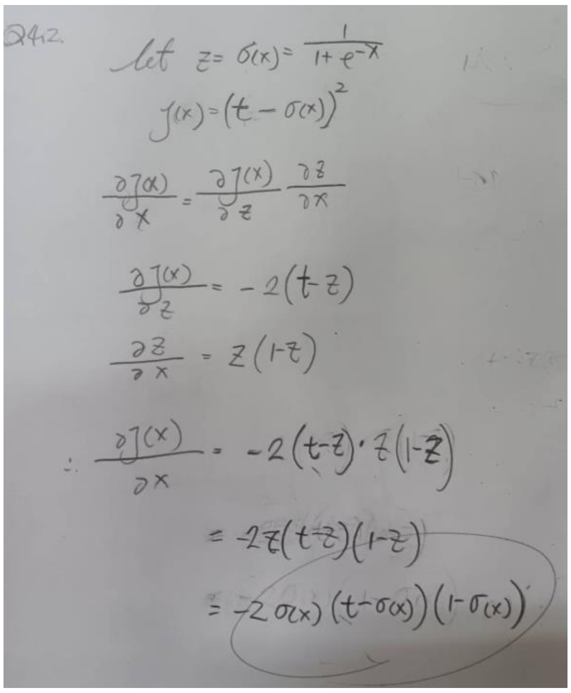
$= \frac{z-t}{z(1-z)} \cdot z(1-z)$

$= z-t$

$= \sigma(x) - t$

4.2. (10%)

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



Q4.2.

let $z = \sigma(x) = \frac{1}{1+e^{-x}}$

$J(x) = (t-\sigma(x))^2$

$\frac{\partial J(x)}{\partial x} = \frac{\partial J(x)}{\partial z}\frac{\partial z}{\partial x}$

$\frac{\partial J(x)}{\partial z} = -2(t-z)$

$\frac{\partial z}{\partial x} = z(1-z)$

$\therefore \frac{\partial J(x)}{\partial x} = -2(t-z) \cdot z(1-z)$

$= -2z(t-z)(1-z)$

$= -2\sigma(x)(t-\sigma(x))(1-\sigma(x))$