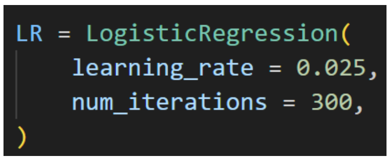
NYCU Introduction to Machine Learning, Homework 2

[111550113], [謝詠晴]

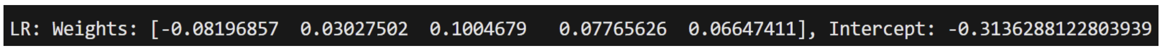
Part. 1, Coding (60%):

(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.

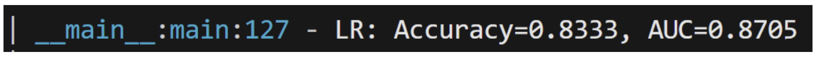


```
LR = LogisticRegression(
    learning_rate = 0.025,
    num_iterations = 300,
)
```



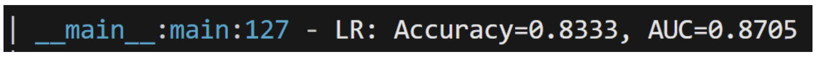
```
LR: Weights: [-0.08196857  0.03027502  0.1004679   0.07765626  0.06647411], Intercept: -0.3136288122803939
```

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



```
| __main__:main:127 - LR: Accuracy=0.8333, AUC=0.8705
```

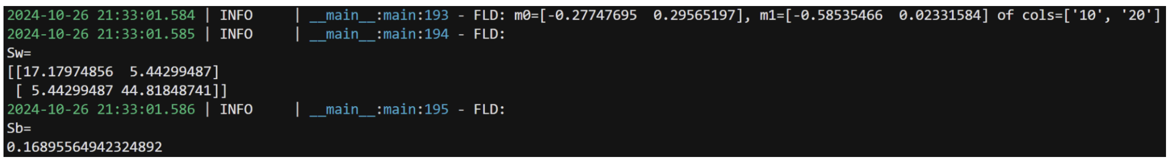
3. (15%) Show the accuracy score of your model on the testing set.



```
| __main__:main:127 - LR: Accuracy=0.8333, AUC=0.8705
```

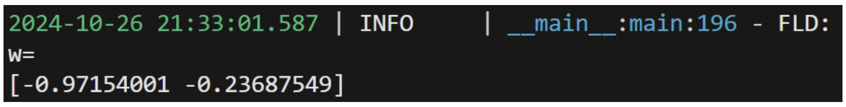
(25%) Fisher Linear Discriminant, FLD

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.



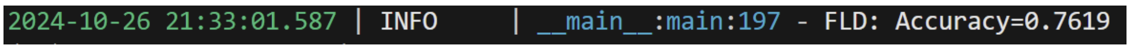
```
2024-10-26 21:33:01.584 | INFO     | __main__:main:193 - FLD: m0=[-0.27747695  0.29565197], m1=[-0.58535466  0.02331584] of cols=['10', '20']
2024-10-26 21:33:01.585 | INFO     | __main__:main:194 - FLD:
Sw=
[[17.17974856  5.44299487]
 [ 5.44299487 44.81848741]]
2024-10-26 21:33:01.586 | INFO     | __main__:main:195 - FLD:
Sb=
0.16895564942324892
```

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

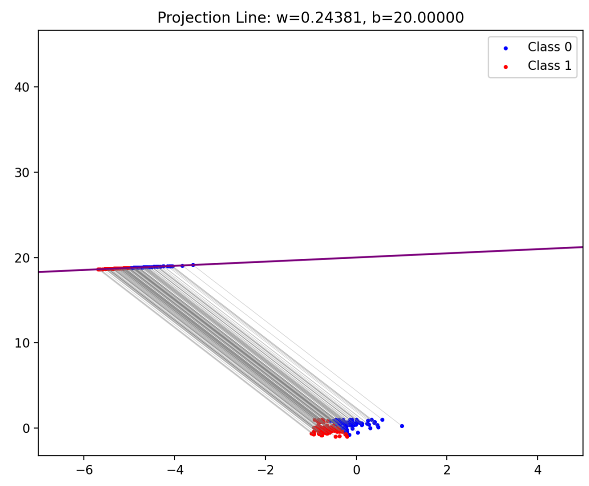


```
2024-10-26 21:33:01.587 | INFO     | __main__:main:196 - FLD:
w=
[-0.97154001 -0.23687549]
```

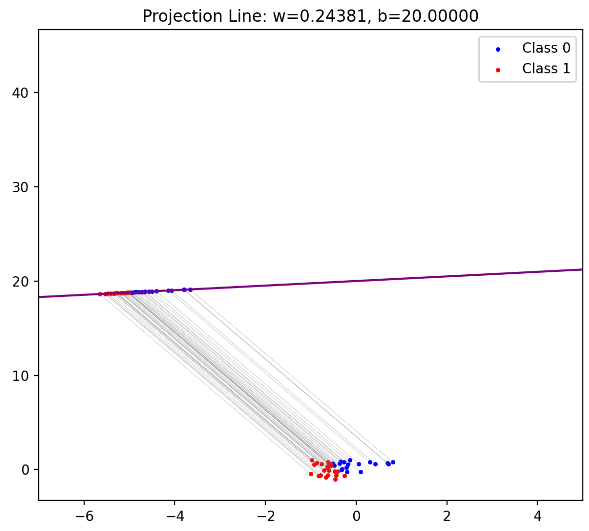
6. (15%) Obtain predictions for the testing set by measuring the distance between the projected value of the testing data and the projected means of the training data for the two classes. (Also, plot for training data). Show the accuracy score on the testing set.



```
2024-10-26 21:33:01.587 | INFO     | __main__:main:197 - FLD: Accuracy=0.7619
```



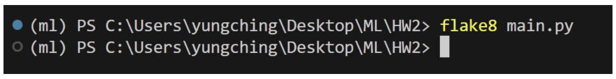
[training set]



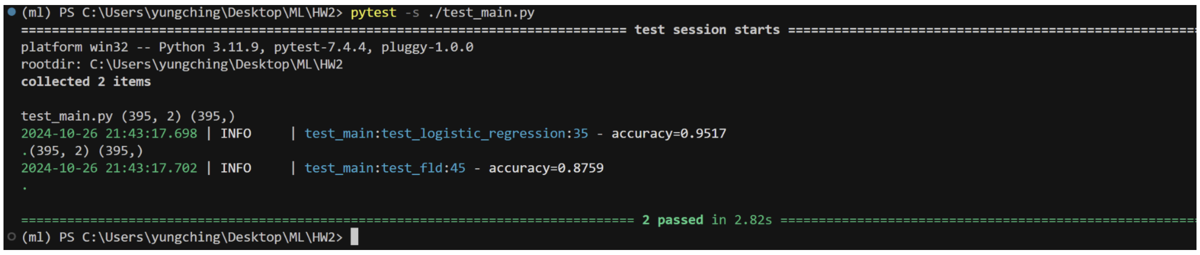
[testing set]

(10%) Code Check and Verification

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



```
● (ml) PS C:\Users\yungching\Desktop\ML\HW2> flake8 main.py
○ (ml) PS C:\Users\yungching\Desktop\ML\HW2> █
```



```
● (ml) PS C:\Users\yungching\Desktop\ML\HW2> pytest -s ./test_main.py
============================================================================== test session starts =====================================================
platform win32 -- Python 3.11.9, pytest-7.4.4, pluggy-1.0.0
rootdir: C:\Users\yungching\Desktop\ML\HW2
collected 2 items

test_main.py (395, 2) (395,)
2024-10-26 21:43:17.698 | INFO     | test_main:test_logistic_regression:35 - accuracy=0.9517
.(395, 2) (395,)
2024-10-26 21:43:17.702 | INFO     | test_main:test_fld:45 - accuracy=0.8759
.

=============================================================================== 2 passed in 2.82s ======================================================
○ (ml) PS C:\Users\yungching\Desktop\ML\HW2> █
```

Part. 2, Questions (40%):

1. (10%)
   - Is logistic 'regression' used for regression problems?
   - If not, what task is it primarily used? (without any additional techniques and modification); If yes, how can it be implemented?
   - Why are we using the logistic function in such a task? (list two reasons)
   - If there are multi-class, what should we use to substitute it?

   ANS:
   No, logistic regression is not used for regression problems. Instead, it is used for classification problems.
   The logistic function is used because:
   1) It can map a real valued number into a probability between 0 and 1 , making it ideal for classification tasks since it represents the likelihood of a data point belonging to a specific class.
   2) The curve of the logistic function has a threshold, which naturally fits the needs of binary classification. The values greater than the threshold can be classified as one class.

   For multi-class problems, we can use softmax function instead of sigmoid function.

2. (15%) When a trained classification model shows exceptionally high precision but unusually low recall and F1-score, what potential issues might arise? How can these issues be resolved? List at least three solutions.
   Potential issues:
   High precision but low recall and F1-score means that the ability of the model to identify positive samples accurately is perfect but it still misses many samples that are positive. The model may have a high threshold or the dataset may contain more negative samples than positive ones. Also, the model might be overfitting to features which have a strong relationship with the positive class.
   Solutions:
   1) Lower the decision threshold can let the model classify more samples as positive, thus recall may increase.
   2) If there are more negative samples than positive ones, we can try oversampling the minority class to ensure the training data is balanced, avoiding any unfairness.
   3) We can modify the loss function or evaluation metrics to shift the model's focus from solely optimizing precision to prioritizing recall or F1-score.

3. (15%) In this homework, we use Cross-Entropy as the loss function for Logistic Regression. Can we use Mean Square Error (MSE) instead? Why or why not? Please explain in detail.
   MSE is more suitable for continuous value predictions because it calculates the differences between predicted and true value. On the other hand, cross-entropy measures the differences between two probability distributions, which is a more

meaningful interpretation in classification cases. If using MSE with logistic regression, the gradients may become very small and result in slower convergence. When the predicted probabilities are close to the true label, the updates value made to the weights may be really small, causing the model to fail to converge. With cross-entropy, it penalizes incorrect confident predictions more strongly than MSE.