Logistic Regression

1.The data set you have chosen and its main characteristics.

The datasets with ids 53 and 602 were selected on the uci website.The dataset with id 53 is Iris, with a small number of features, only four, and all of them are continuous and size-differentiated feature data.The dataset with id 602 is a dry bean dataset with a high number of features, 16 dimensions, and tens of thousands of samples.

ps:There is no missing data in the above two datasets.

2.The data preprocessing operations you have applied to improve data quality.

- For numerical features, there are characteristics of uneven distribution and large order of magnitude gap. Normalizing them can effectively avoid the risk of weight bias and gradient explosion during training.
- For non-numeric features, a one-hot approach is used to encode them so that they can be applied to the loss function calculation for logistic regression. However, this is prone to excessive number of features and slow computation, so pca is used for dimensionality reduction.
- The dataset target exists in multiple categories, and when predicting the binary classification, the positive and negative samples are ensured to be balanced by sample sampling so that they are around 50%.

3.The design of the different modules of your program. In describing each module, you should relate your design to the theories you have learnt in the class.

- common.py holds the hyperparameters for model training and model testing, including training epoch, iteration count, gradient descent parameters, etc.
- data.py is the module that processes the data and improves the quality of the data. After reading the data into memory through the data import method provided by uci, the data is then processed in the way mentioned in question 2, and the data is categorized into raw data, training set data, test set data, metadata, etc., and put into a folder with the name **id**.
- logisticloss.py is a loss function module and allows for multicategorical predictions.
- test.py is a model testing module that outputs the correctness or incorrectness of the predictions and the probability of belonging to a certain category.
- train.py is the model training module, which uses batch stochastic gradient descent to make the model converge more quickly.

4.The prediction results and your interpretation of the results. Plot a figure like the following one in your report:

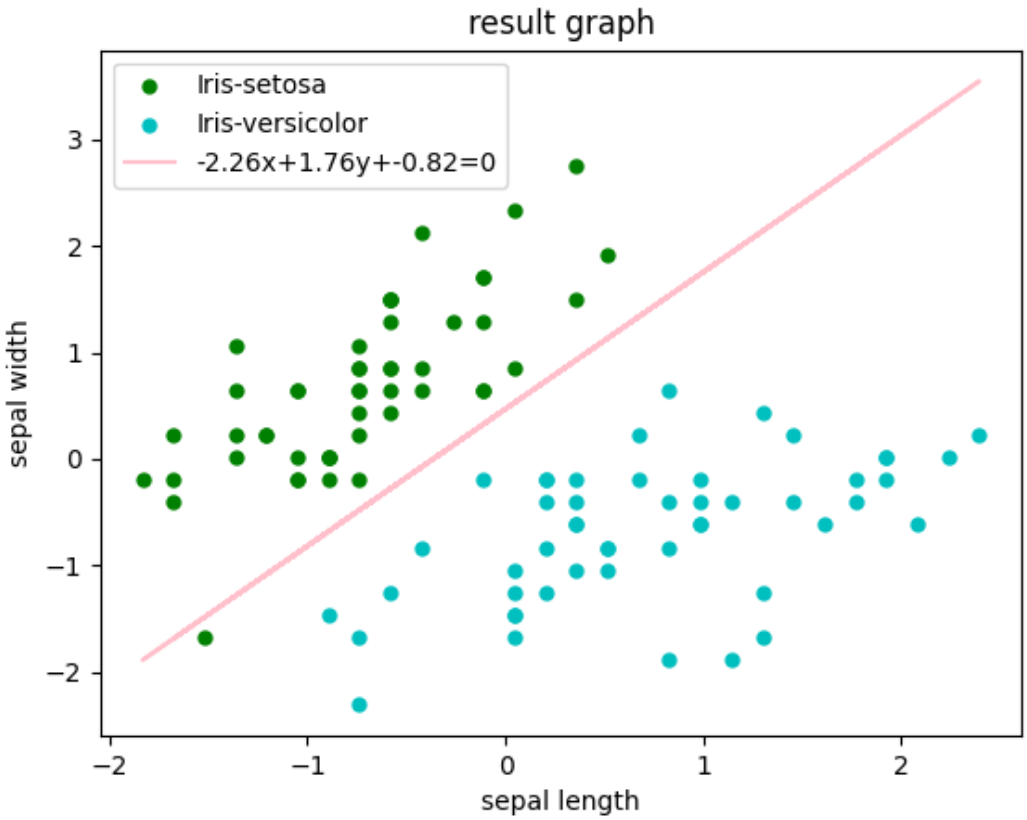
The code is set up with a training set to test set ratio of **4:1**.

4.1 Iris dataset

Here are the results of the test, in which the accuracy is 100%.

```
------------------------------
sampler 16 true value is Iris-versicolor, predict value is Iris-versicolor
probability:tensor([0.0054, 0.9946])
loss:0.005420147906988859
------------------------------
sampler 17 true value is Iris-versicolor, predict value is Iris-versicolor
probability:tensor([1.5837e-05, 9.9998e-01])
loss:1.585496102052275e-05
------------------------------
sampler 18 true value is Iris-versicolor, predict value is Iris-versicolor
probability:tensor([2.4940e-04, 9.9975e-01])
loss:0.000249476550379768
------------------------------
sampler 19 true value is Iris-versicolor, predict value is Iris-versicolor
probability:tensor([9.2568e-04, 9.9907e-01])
loss:0.0009260888327844441
------------------------------
acc: 100.00%

Process finished with exit code 0
```

Two categories, iris-setosa and iris-versicolor, were selected for categorization, and two dimensions, sepal length and sepal width, were selected for plotting.

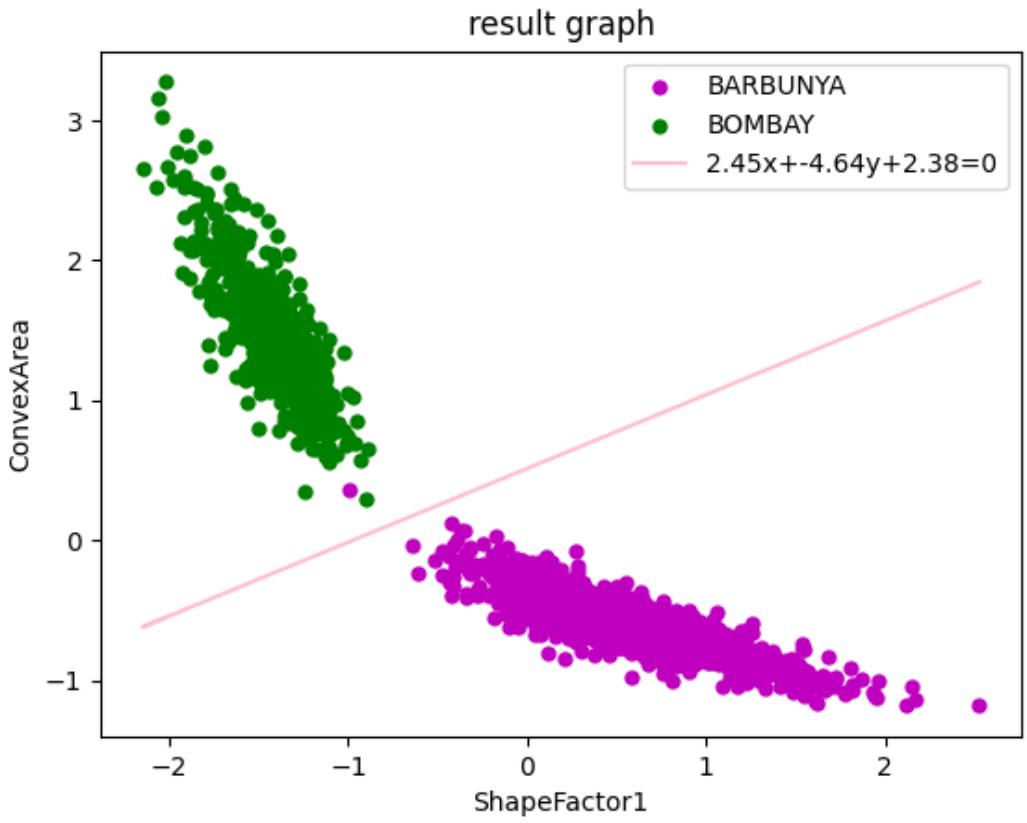


4.2 Dry bean dataset

Here are the results of the test, in which the accuracy is 99.52%.

```
sampler 205 true value is BOMBAY, predict value is BOMBAY
probability:tensor([4.7554e-07, 1.0000e+00])
loss:4.768372718899627e-07

------------------------------
sampler 206 true value is BOMBAY, predict value is BOMBAY
probability:tensor([1.0318e-04, 9.9990e-01])
loss:0.00010318096610717475

------------------------------
sampler 207 true value is BOMBAY, predict value is BOMBAY
probability:tensor([1.1480e-05, 9.9999e-01])
loss:1.150376283476362e-05

------------------------------
sampler 208 true value is BOMBAY, predict value is BOMBAY
probability:tensor([6.9776e-06, 9.9999e-01])
loss:6.9737675403303e-06

------------------------------
acc: 99.52%

Process finished with exit code 0
```

Two categories,BARBUNYA and BOMBAY, were selected for categorization, and two dimensions, shapeFactor1 and convexarea, were selected for plotting.



5.Limitations and possible improvements of the program.

- Code lacks handling of missing or ambiguous data.

- The categorical hyperplane for graphing is $\boldsymbol{w}^T\hat{\boldsymbol{x}}+\boldsymbol{b}=\boldsymbol{0}$, subject to $\hat{\boldsymbol{x}}=\boldsymbol{g}(\boldsymbol{x})$, $\boldsymbol{x}\in\mathcal{D}$. If $\boldsymbol{g}$ is a nonlinear mapping, it is difficult to find a norm like $\boldsymbol{w}\rightarrow\boldsymbol{h}(\boldsymbol{w})$,$\boldsymbol{b}\rightarrow\boldsymbol{t}(\boldsymbol{b})$ mapping back to the original space
- Since the model training is simple and there are not a large number of hyperparameters, there is no division of the validation set