Logistic Regression and Multi-layer Neural Network Construction Model

Xingang Liu
AI and Machine Learning Class Fall 2023
Southern University of Science and Technology
Shenzhen, China
2652173921@qq.com

***Abstract*—This document is a model and instructions for LaTeX. This and the IEEEtran.cls file define the components of your paper [title, text, heads, etc.]. *CRITICAL: Do Not Use Symbols, Special Characters, Footnotes, or Math in Paper Title or Abstract.**

***Index Terms*—Logistic regression, multi-layer neural network, machine learning, algorithm implementation**

I. Introduction

AI and machine learning has a wide range of applications in various fields of life. Building a model through some algorithms and data sets and making predictions through the model is a common process of supervised learning in machine learning. Solving classification problems is one of the important applications in this direction. Logistic regression and multi-layer neural network learning can construct linear and nonlinear prediction models to solve the problem respectively

II. Problem formulation

For binary classification problems in machine learning, the algorithm using logistic regression can develop a better prediction model through a training set that can be linearly segmented. However, the performance of logistic regression is not satisfactory for training sets that cannot be linearly segmented. In order to solve this problem, the model construction of multi-layer neural network can be carried out. By increasing the number of layers or single-layer neurons of the neural network, a nonlinear prediction model based on the training set can be built, and the model can fit the data of the training set well.

A.

III. Method and Algorithms

A. Logistic Regression

1) theory: In logistic regression algorithm, if the number of features of a single sample x is n, n parameters w are set, multiplied by each feature and added together with w0 as the weight of each feature and the bias quantity. For a single sample x, each feature of x is multiplied by the corresponding randomly initialized w, plus the randomly initialized bias quantity w0 as the quantity describing the total feature of that sample. Define sigmoid as:

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

Fig. 1. sigmoid function

For a binary classification problem that classifies the sample into class 0 or class 1, the sigmoid function is assumed as a function describing the relationship between a sample's total feature and the probability that the sample belongs to class 0. So we get the probability y that the sample belongs to class 0:

$$y(\mathbf{x}) = \sigma\left(\mathbf{w}^T\mathbf{x} + w_0\right)$$

Fig. 2. classification probability

And the probability of this sample belonging to class 1 can expresses as (1 - P(0)). Then the probability of the occurrence of the training set can be calculated as loss function.

$$\ell_{log-loss}(\mathbf{w}) = -\sum_{i=1}^{N} t^{(i)} \log p(C = 1|\mathbf{x}^{(i)}, \mathbf{w}) - \sum_{i=1}^{N} (1 - t^{(i)}) \log p(C = 0|\mathbf{x}^{(i)}, \mathbf{w})$$

Fig. 3. negative logarithm of loss to w

Then the derivative of the loss function with respect to the parameter w is obtained by:

The parameter w can then be updated:
Model training is the process of repeatedly forecasting

$$\frac{\partial \ell}{\partial w_j} = \sum_i x_j^{(i)} \left(t^{(i)} - p(C = 1 | \mathbf{x}^{(i)}; \mathbf{w}) \right)$$

Fig. 4. partial derivatives of loss function with respect to w

$$w_j^{(t+1)} \leftarrow w_j^{(t)} - \lambda \sum_i x_j^{(i)} \left(t^{(i)} - p(C = 1 | \mathbf{x}^{(i)}; \mathbf{w}) \right)$$

Fig. 5. gradient descent

and updating parameters, and the final prediction model is obtained after the termination of training.

2) practice: Generate an array containing the coordinates and class of the points shown in the following figure as a training set.

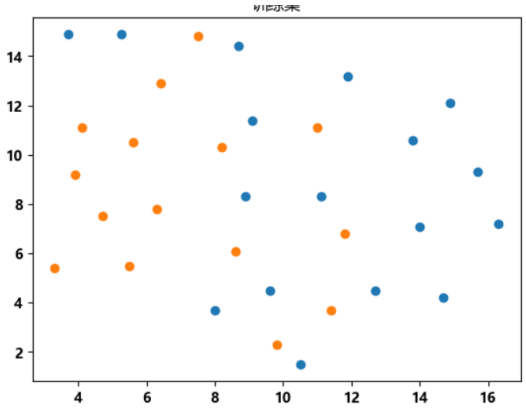


Fig. 6. training set

Write a class code for model training and storage based on logistic regression's principles. In the code part of model training, the author sets the threshold for parameter updating. When the amplitude of parameter updating exceeds the threshold, the threshold is taken as the amplitude of gradient updating to reduce the impact of gradient explosion caused by excessive learning rate. Adjust the learning rate and the condition tolerance to stop the training, observe and analyze the effect of the prediction model.

B. Multi-Layer Neural Network

1) theory: On the basis of logistic regression algorithm, before the total features of the sample are transmitted to the output layer, several layers of hidden layers containing several units are added, and each unit is set with a parameter w number 1 more than the number of features entered into the unit. All units in the same layer use the same nonlinear function as the activation function. The total feature of each sample is input to all the units of the first hidden layer respectively, and is added with the corresponding parameter w of the unit, and then input to the activation function, and the output value of the activation function is input value of the next layer. The last layer hides the output value of the layer as input to the output layer. In the same way as logistic regression, the same loss function can be calculated. Calculate the partial derivative of the loss function with respect to each w so every w can be updated. By repeatedly predicting and updating the value of w, the model is slowly trained and the model means what value each w is.

2) practice:

a) code implementation: Generate the same array as what shows in Fig.6. The author write a class to train and preserve the well-trained model. In the class, during the process of forward passing, a method called hidden-layer is written to calculate the output and needed partial derivatives of hidden layers and another method called train-output-layer is written to calculate the probability that the sample is predicted to be class 0 and needed partial derivatives about the loss function with the input as well as weights in output layer. Parts of the two methods are showed as follows:

```
def hidden_layer(self,x,w_matrix):
    #return([y_matrix,y_derivative_of_w_matrix,y_derivative_of_x_matrix])
```

Fig. 7. hidden layer

```
def train_output_layer(self,x,t,terminate):
    #return [y,error_derivative_of_w,error_derivative_of_x]
```

Fig. 8. output layer

During the course of back passing, the author failed to use pure matrix operation but instead of some circulation methods to get the loss function's partial derivatives with respect to matrix. As a result, the codes run too slowly to train the model, because it cost more than ten minutes to train for about just ten thousand times even when the result of hidden layers timing the number of unit in each hidden layers is about just ten. So the author test the model usually when the training number less than several thousand.

b) relu to replace sigiod: In order to solve the problem of gradient disappearance caused by sigmiod, the author tries to change the activation function of hidden layer to relu function. It was later found that the negative zero operation of the relu function would cause the unit to lose its function, so the activation of the negative part was replaced by a proportional function with a slope of 0.9.

IV. Experiment Results and Analysis

A. Logistic regression

By changing the learning rate and tolerance of training, different models will be trained. The author plotted the prediction results of the model trained under different learning rate and tolerance and the changes of loss function with the training times. Parts of models' result are showed, each one with prediction of the training set showed as

follows:

When the learning rate equals 0.00002 and the tolerance equals 0.000001:

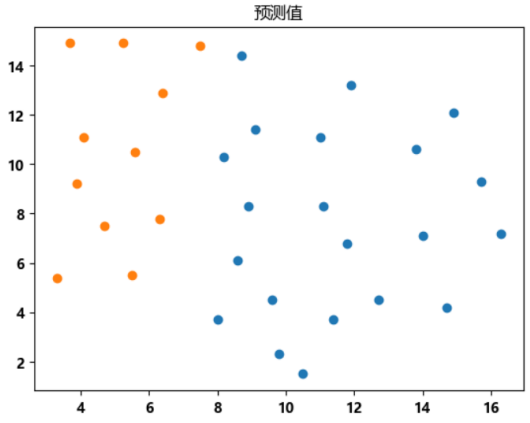


Fig. 9. predicted value

1

```
TP:9
TN:15
FP:6
FN:2
recall:0.8181818181818182
presion:0.6
F1:0.6923076923076923
```

Fig. 10. predictive performance evaluation

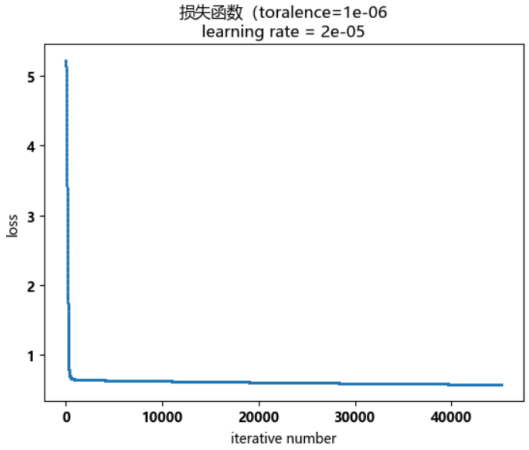


Fig. 11. loss function

When the learning rate equals 0.00002 and the tolerance equals 0.001:

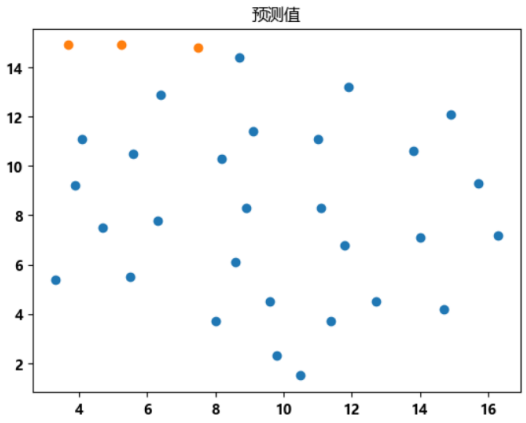


Fig. 12. predicted value

```
TP:1
TN:15
FP:14
FN:2
recall:0.3333333333333333
presion:0.06666666666666667
F1:0.1111111111111111
```

Fig. 13. predictive performance evaluation

The plots show that When the tolerance is relatively small, the model is sufficiently trained to obtain the prediction result of linearized segmentation according to the logisitc regression expectation. However, when the tolerance increases, the training times of the model will decrease, which will easily lead to the poor effect of the obtained model.

When keeping the tolerance at a suitable value and changing the learning rate, the result is summmarized as follows: When the learning rate is reduced, the training times will increase

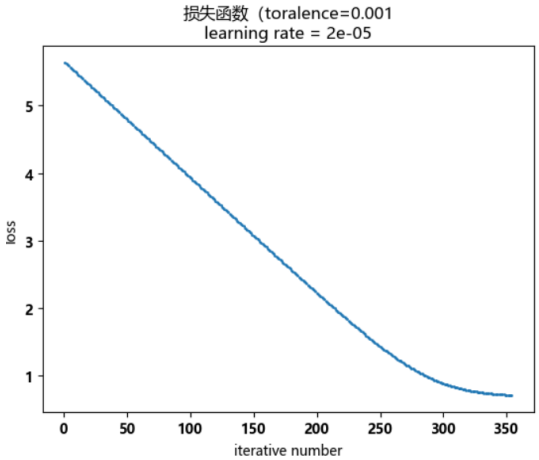


Fig. 14. loss function

and the model effect will not change much. When the learning rate is increased to 0.02, the training times are shortened and the loss function appears to jump, but the overall trend is still downward, with little difference in model effect. When the learning rate continues to increase, The loss function starts to diverge and the model can't get theoretical prediction effects.This is because the learning rate determines the range of each parameter update. Too large a learning rate is likely to cause the parameter to be updated to a monotonic interval in which the loss function drops to a certain minimum value, and the reduction of learning rate will reduce the variation of each parameter update. In turn, more training sessions are needed to reach the minimum point.

B. Multi-Layer Neural Network

1) Failed to write the program successfully: The author writes a class named "MLP" which can set the number of hidden layers and the number of units in each layer:

```
class MLP:
    def __init__(self,hidden_layer_number,the_number_of_units_in_each_layer,feature_number,learning_rate,terminate_num):
```

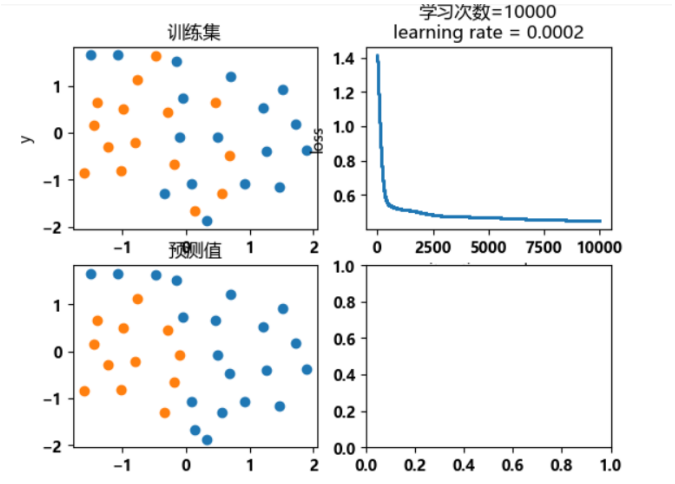
Fig. 15. initialize method of class "MLP"

When the number of hidden layers is set to 0, that is, only the output layer, this class can train a binary classification model with the same prediction principle and prediction effect as the "logistic regression" mentioned above. However, when the number of hidden layers is not 0, the class fails to train the theoretical nonlinear prediction model. The author try hard but eventually fail to fix the problems. Next paragraph will show the result of the model test and analyze its reasons.

2) Result shows: By changing the number of hidden layers and , different models will be trained. The author plotted the prediction results of the model trained under different learning rate and tolerance and the changes of loss function with the training times. Parts of models' result are showed as follows:
Although the loss function can be reduced by increasing the number of layers and units, the prediction perform does not show an obvious trend of nonlinearized prediction.
After the activation function of the hidden layer is replaced by relu function, although loss function decrease faster, the problem of slow decline of loss function is still not solved well. Under the training times set during the test, the model's prediction of the sample still fails to show obvious nonlinearization.

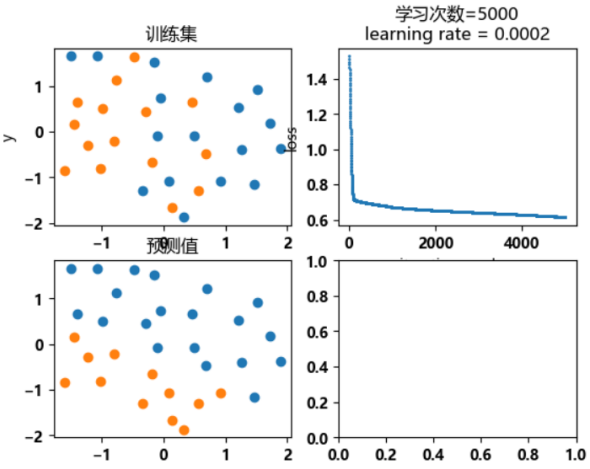
3) Reason Analysis: Based on the above test results, the reasons leading to the failure to train the nonlinear model of multi-layer neural network are analyzed:

a) : Because a normal linear prediction model can be constructed when the number of hidden layers is set to 0, the problem of inferred code mainly lies in the updating of hidden layer parameters in back-propagation: error in the partial derivatives calculation of loss function with respect to the hidden layer input, or error in the code writing of the data save and call when iterating back to update.




```
2层神经网络，隐藏层1层，10个神经元，sigmod函数作为激活函数
训练次数: 10000 learning rate: 0.0002
训练集的错误个数: 7.0
对于训练集的预测准确率: 0.78125
TP:10 TN:15 FP:5 FN:2
recall:0.8333333333333334 presion:0.6666666666666666
F1 score:0.7407407407407408
```

Fig. 16. One hidden layers and 10 units with training number 50000




```
4层神经网络，3个隐藏层，每层3个神经元
learning_rate = 0.0002 终止次数 = 5000
训练集的错误个数: 11.0
对于训练集的预测准确率: 0.65625
TP:8 TN:13 FP:7 FN:4
recall:0.6666666666666666 presion:0.5333333333333333
F1 score:0.5925925925925926
```

Fig. 17. Three hidden layers and 3 units each with training number 5000

b) : Because the code uses too many loops instead of matrix operations, the training takes a long time. The authors stopped before training the model enough times during the test. At the same time, the hidden layer uses sigmoid function as the activation function, and the gradient will disappear during the training process. Therefore, stopping training before the model training is sufficient can result in a situation where the correct model is not observed.

c) : After changing the activation function of the hidden layer to relu function, the ideal prediction model still fails to be obtained under the test training times, which may cause the gradient to disappear because the output layer uses sigmoid as a function.

V. Conclusion and Future Problems

A. Conclusion

a) : The learning rate directly affects the quality of the model. When training the model, it is necessary to set a moderate learning rate to avoid gradient explosion, which makes the model fail to reach the prediction function and avoid too slow training speed.

b) : tolerance determines the termination condition of model training. Setting a moderate tolerance can get a well-performing training model without consuming too much time.

c) : The excessive use of the for loop structure will seriously slow down the running speed of the code, and the for loop structure should be replaced by the integrated matrix operation in numpy when the code is written.

B. Future Problems

a) : Optimize code to improve operational efficiency.

b) : Collect some real samples to make training sets and test sets, and use the code to build predictive models to further evaluate the practicability of the code.

c) : Try to use other classification algorithms and compare their advantages and disadvantages with logistic regression binary classification and multi-layer neural network binary classification.