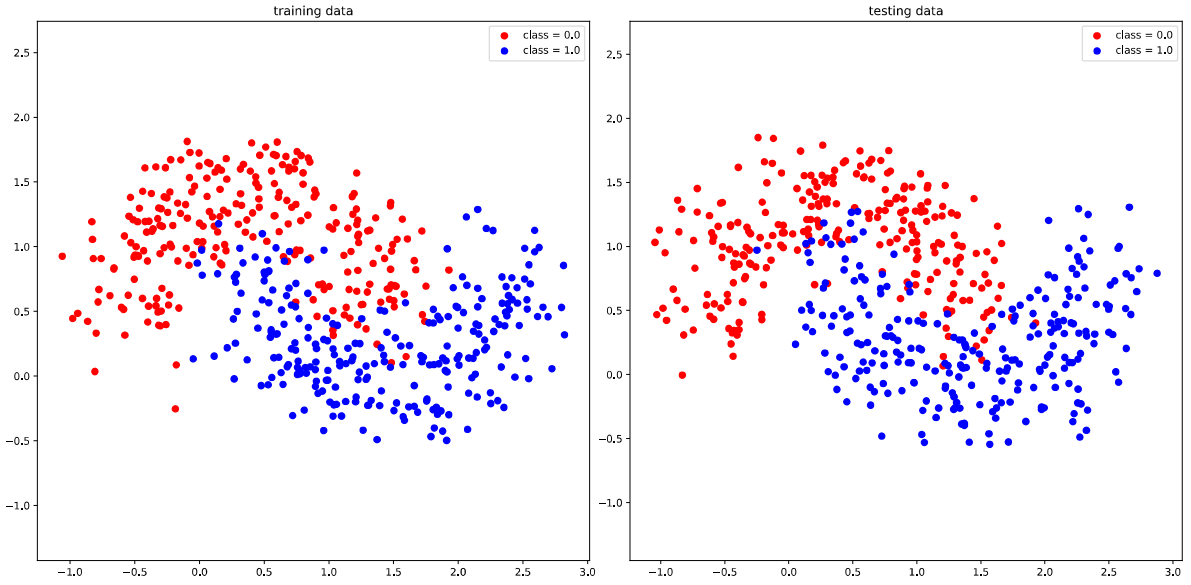
results

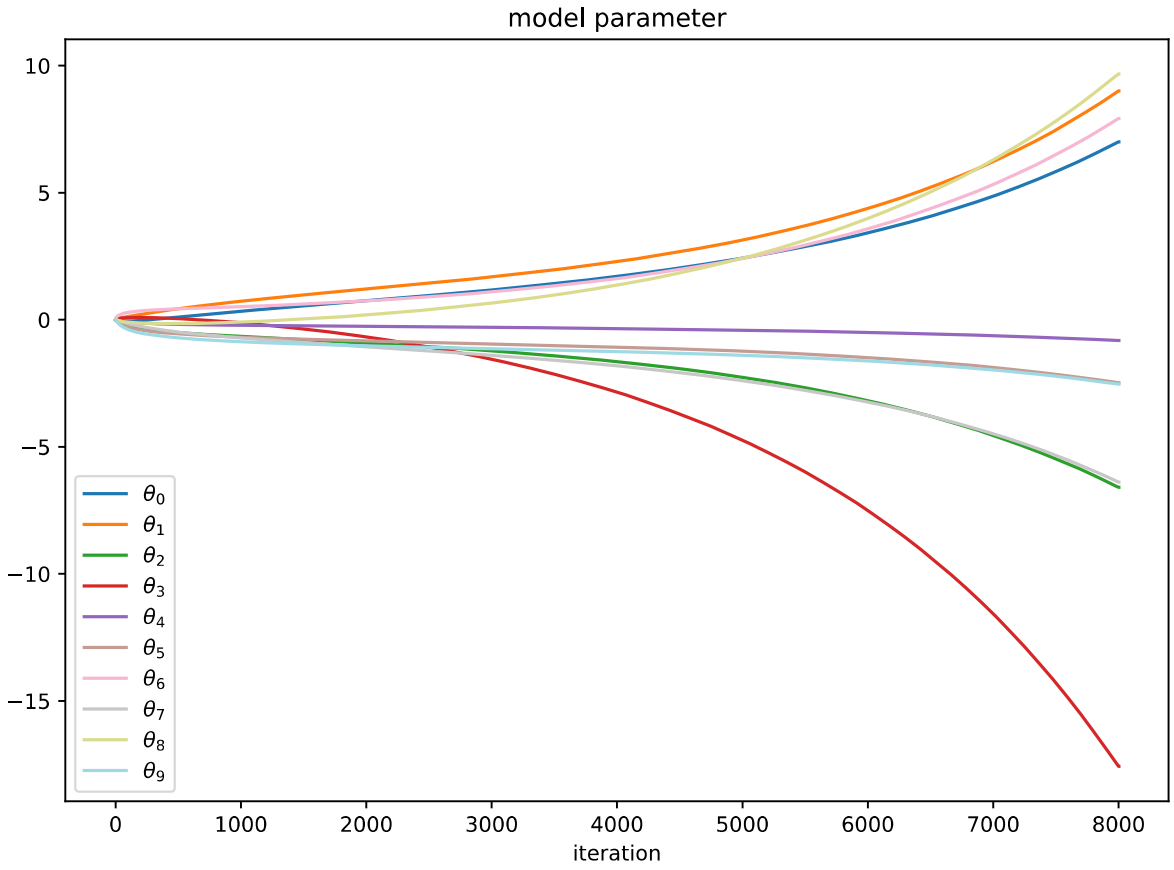
1. plot the input data (training on the left sub-figure and testing on the right sub-figure) in blue for class 0 and in red for class 1 from the file [assignment_10_data_train.csv] and [assignment_10_data_test.csv], respectively,

In [ ]:
```
plot_data(data_train, data_test)
```



1. plot the values of the model parameters θ as curves over the gradient descent iterations using different colors

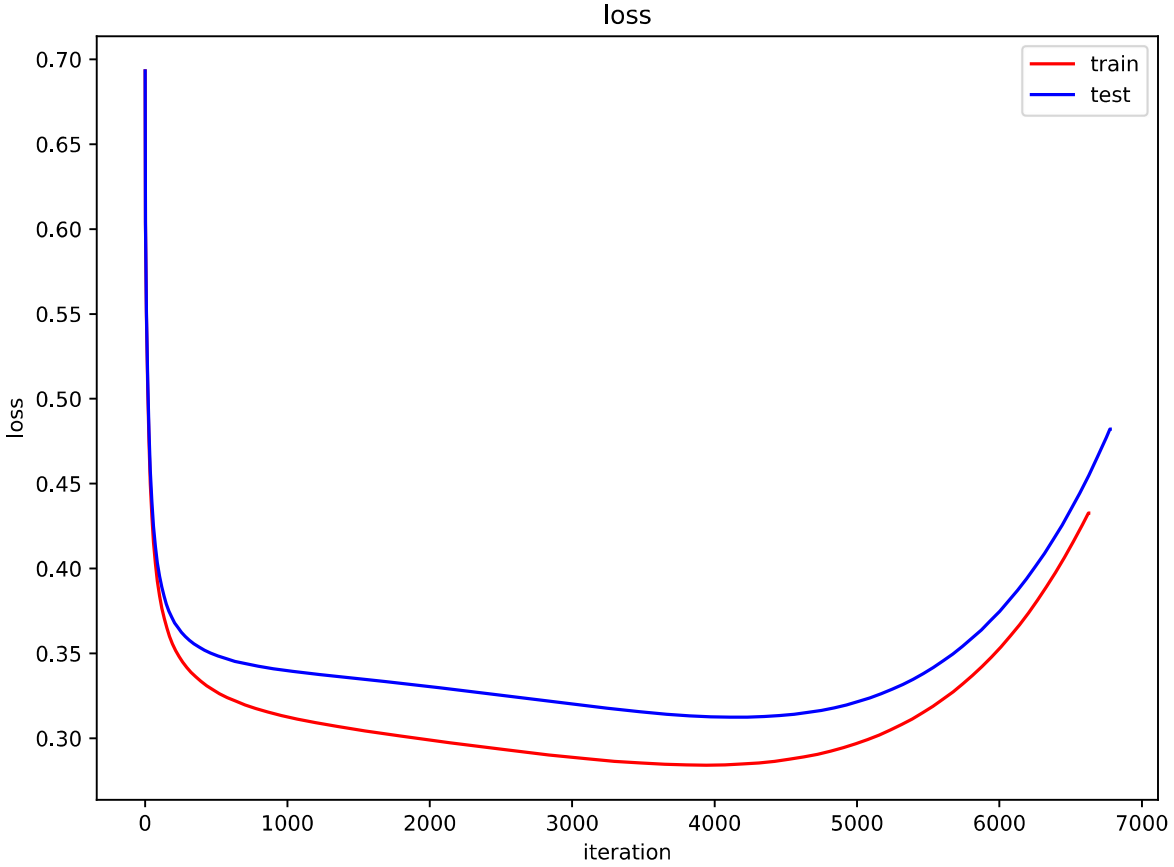
In [ ]:
```
plot_model_parameter(theta_iteration)
```



---

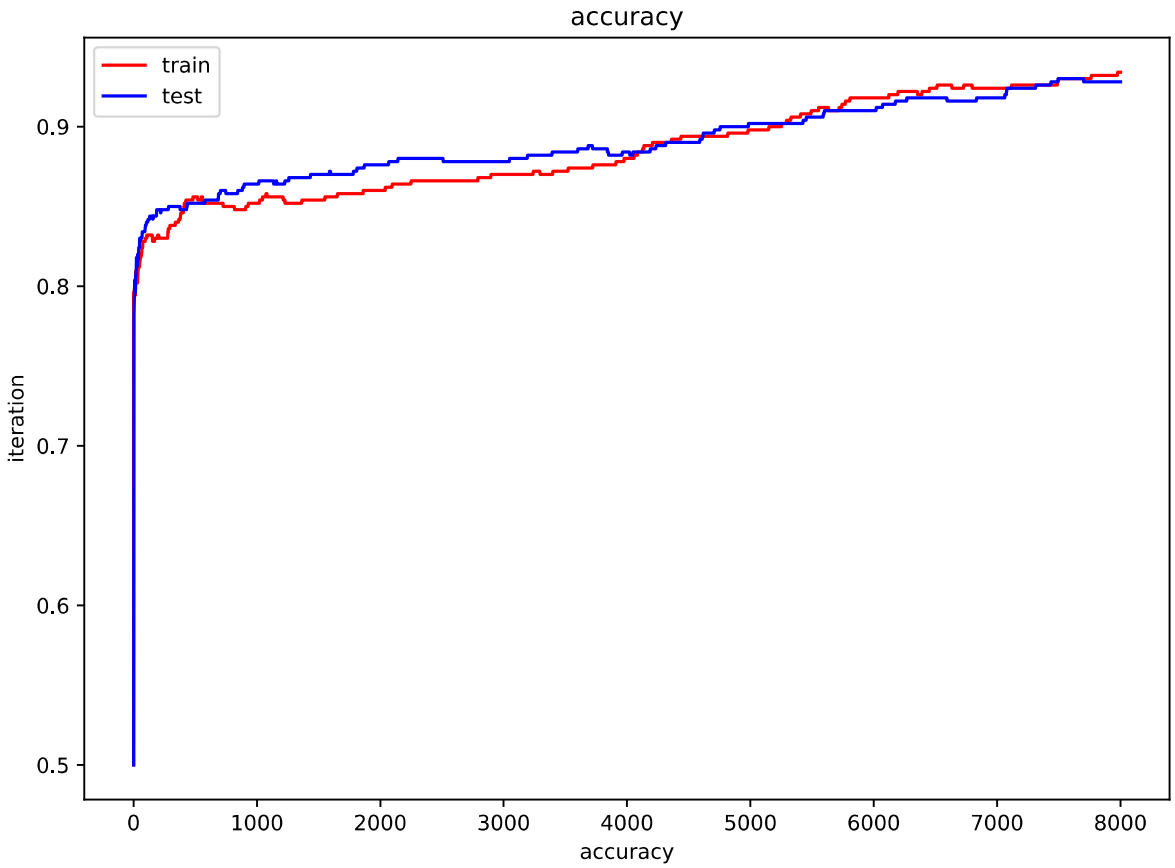
1. plot the training loss in red curve and the testing loss in blue curve over the gradient descent iterations

In [ ]:
```
plot_loss_curve(loss_iteration_train, loss_iteration_test)
```



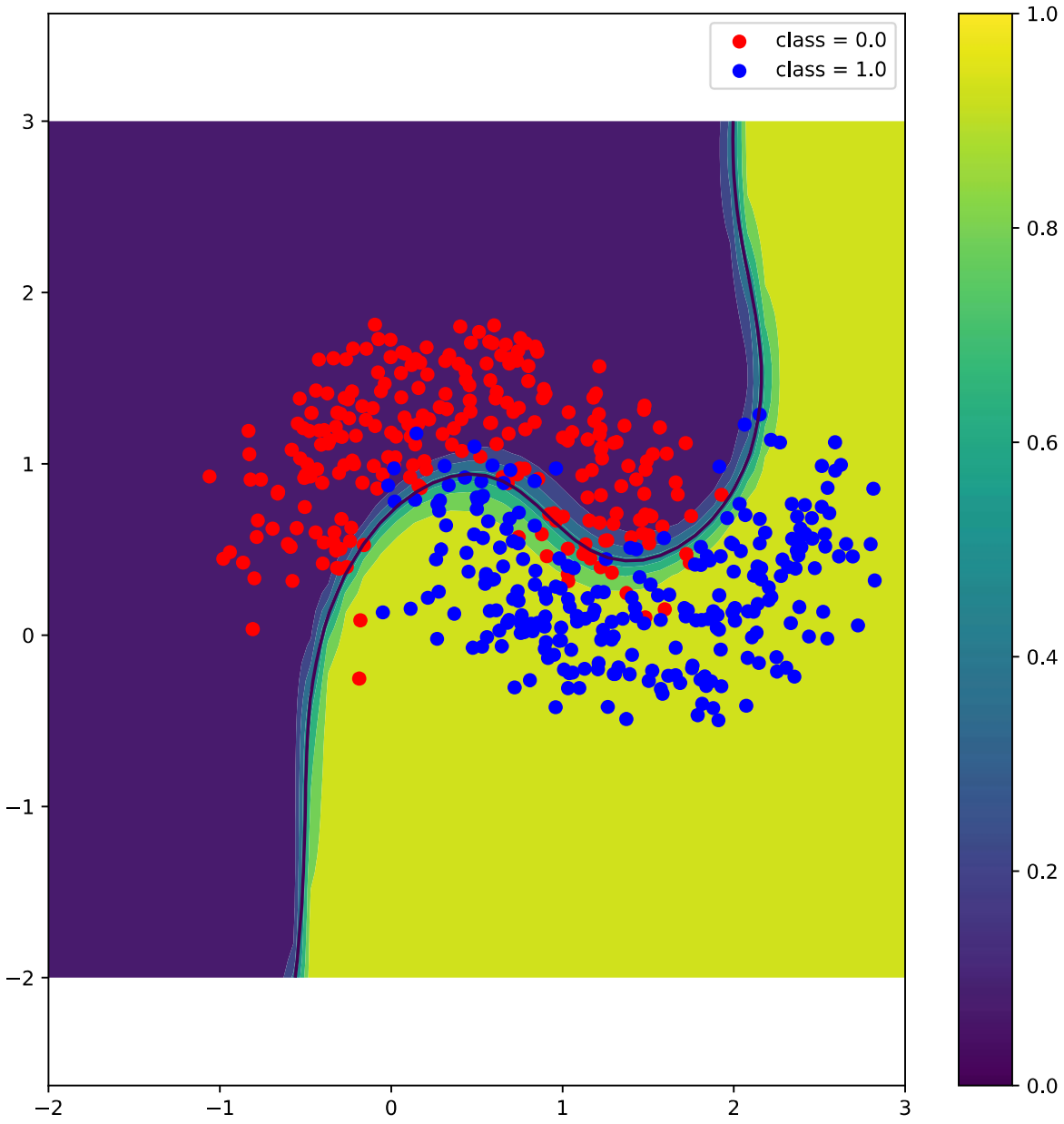
1. plot the training accuracy in red curve and the testing accuracy in blue curve over the gradient descent iterations

In [ ]:
```
plot_accuracy_curve(accuracy_iteration_train, accuracy_iteration_test)
```



1. plot the classifier using the prediction values in the color coding scheme ranges from blue (class 0) to red (class 1) with the training data

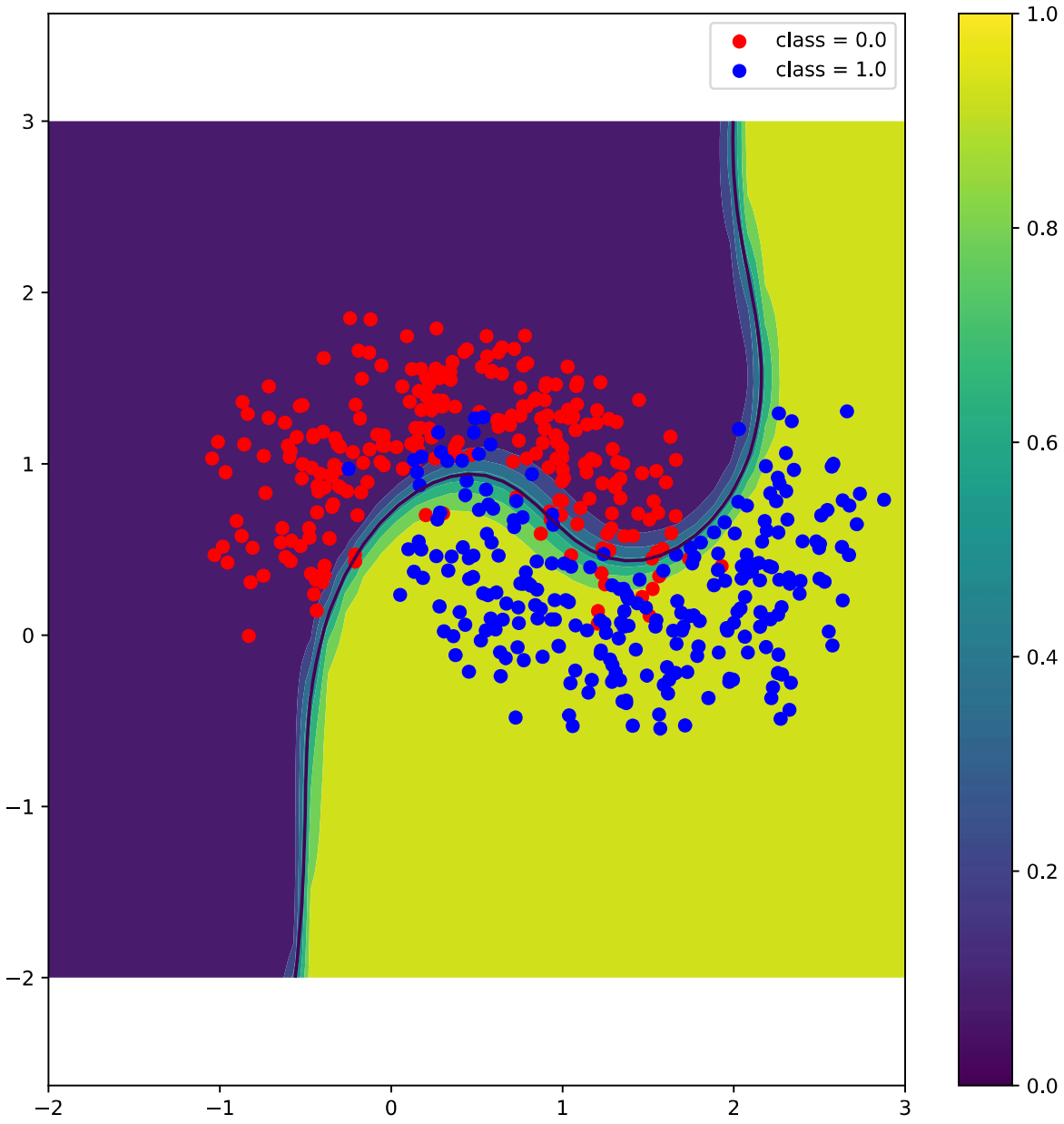
In [ ]:
```
plot_classifier(data_train, theta_optimal)
```



---

1. plot the classifier using the prediction values in the color coding scheme ranges from blue (class 0) to red (class 1) with the testing data

In [ ]:
```
plot_classifier(data_test, theta_optimal)
```



1. print out the final training accuracy and the final testing accuracy in number with 5 decimal places (e.g. 0.98765)

In [ ]:
```
print('accuract(train): {0:6.5f}'.format(accuracy_train))
print('accuracy(test) : {0:6.5f}'.format(accuracy_test))
```

```
accuract(train): 0.93400
accuracy(test) : 0.92800
```

In [ ]:
```

```