Practical 1

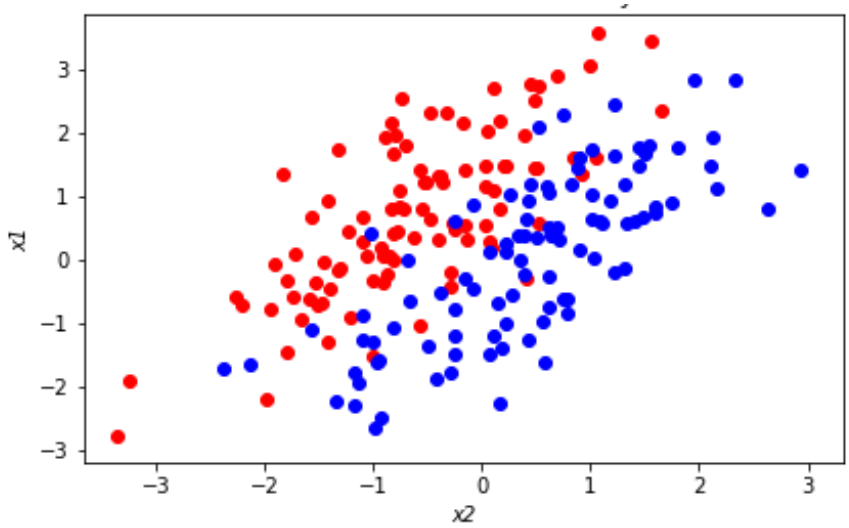
Single layer perceptron SLP
Classification

This practical will help you to understand the influence of the basic single-layer perceptron (SLP) parameters on the classification accuracy.

Use Python code *single_layer_perceptron.py*

1. Data

Data – artificially generated two-dimensional Gaussian data sets: class1 (red) and class 2 (blue).



Single-layer perceptron will be trained to discriminate between these two classes.

Eeach class is decribed by a mean vector and covariance matrix:

```
mean1 = np.array([-0.5, 1])   #  mean vector class 2
mean2 = np.array([0.5, 0])    #  covariance matrix class 2
var1=1 # variance of x1 feature
var2=2 # variance of x2 feature
cor12=0.8 #correlation coefficient between x1 and x2
```

A size of a training set is defined (equal for each class).

```
N_train=100 # number of training samples
N_test=100 # number of testing samples
```

Training set is denoted: data_train1, data_train2.
Test set is denoted: data_test1, data_test2.

Targets:
targets_train1 - zeros, targets_train2 - ones
targets_test1 - zeros, targets_test2 - ones

2. Training and testing of a single layer perceptron

Parameters for training:

```
#parameters for perceptron training
niu = 0.2 # learning Rate
epochs = 200
```

To train and test a perceptron we will use the following functions:

```
# perceptron call
perceptron = Perceptron(data_train, targets_train)

#training
mse=perceptron.train(epochs, niu)

#training errors
error_train, error_train_percent]=perceptron.errors(data_train, targets_train)

#testing errors
[error_test, error_test_percent]=perceptron.errors(data_test, targets_test)
```

The training is implemented in the method

```
def train(self, its, niu):
```

Analyze and understand the code and all methods.
Pay attention to the training gradient descent procedure:

```
def train(self, its, niu):
...

for i in range(N): #over the samples
     cost_prime = 2*(activation[i] - self.targets[i])
     #weight change for a given sample
     for j in range(p+1): #over the features
     delta_weights[j][i] = cost_prime * inputs1[i][j] *
                                   self.sigmoid_deriv(z[i])

     #average of the weight change over the learning samples and transpose
     delta_avg = np.array([np.average(delta_weights, axis=1)]).T

     #weight update
     self.weights = self.weights - niu * delta_avg
```

At the end the figures are generated to present the *mse*, classification error during training, data and decision boundary, neuron output for data, *mse* surface as a function of weights. Training and classification errors are printed in the command window.

Task 1. Analyze what effect a number of training iterations has on the *mse*, training and testing errors. Explain the results.

Mean_1 = [0.8, 0]
Mean_2 = [0.5,0]

Class observed
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1.]
Class predicted
[1 0 0 0 1 0 0 0 1 1 1 0 1 0 1 0 0 0 1 1 1 1 0 1 1 1 1 0 0 0 0 1 0 0 0 0
 0 0 1 0 0 1 1 0 0 0 0 0 1 0 0 1 0 1 0 0 0 0 0 0 1 1 0 0 0 1 1 1 1 0 0 0
 0 0 1 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0 1 0 1 1 0 1 1 1 1 0 0 1
 0 0 1 1 1 1 1 0 1 0 0 1 1 1 1 1 1 1 0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0 1
 1 1 0 0 1 0 1 0 1 1 1 0 0 1 1 1 0 1 0 1 1 1 1 0 1 0 1 0 1 0 0 1 0 0 1 0 1
 0 1 1 1 0 1 0 1 0 1 1 1 1 0 0]
Training errors: 73
Training errors: 36.5 %
Class observed
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1.]
Class predicted
[0 0 0 0 1 0 0 1 0 0 0 0 0 1 0 0 0 1 0 1 1 1 0 1 0 0 1 0 0 1 0 0 0 1 0 0
 0 1 1 0 0 1 0 0 0 1 0 1 1 1 0 1 0 0 1 0 0 0 1 1 0 0 1 1 0 0 0 0 1 0 1 0 1
 1 1 1 1 0 0 0 0 0 1 0 0 1 0 1 1 0 0 0 1 0 0 0 0 0 1 0 1 1 0 1 1 0 0 0 1 0
 0 0 1 1 0 0 0 1 0 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 1 1 0 0 0 1 1 1 1 1 0 0 1
 1 1 0 0 0 0 1 0 0 1 0 1 1 1 0 0 1 1 0 1 1 1 1 0 0 0 0 1 1 1 1 1 1 1 0 0 1
 1 0 1 0 0 0 1 1 1 0 1 1 1 1 0]
Test errors: 78
Test errors: 39.0 %

Note: the new random data set will be generated in every experiment.You can add additional procedure to work with the same data (optional).It will enable you to compare the results in a correct way as you will work with the same data set.

Task 2. Analyze what effect a learning rate niu has on the *mse*, training and testing errors. Explain the results (niu>0).

When using Speed rate at = [0.6]

Class observed
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1.]
Class predicted
[0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 1 1 1 0 0 0 0 0 1 0 0 1 1 1 1
 1 0 0 1 1 1 0 0 1 1 0 0 0 0 1 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 1 1 0 1 0 0 0
 0 1 0 0 1 0 1 0 0 1 0 0 0 0 1 1 0 0 0 0 1 0 0 1 0 0 1 0 1 0 0 0 0 0 0 1 1
 1 1 1 1 0 1 1 1 1 1 0 1 0 1 0 0 0 1 1 1 1 1 0 0 0 1 0 1 0 1 1 1 0 1 1 1 1
 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 0 0 1 1 0 0 1 1 0 1 0 1 1 0 1 1 1 1 1 1 1 1
 1 1 1 1 0 1 1 0 0 0 1 1 1 1 1]
Training errors: 68
Training errors: 34.0 %
Class observed
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1.]
Class predicted
[1 1 1 0 0 1 0 1 0 0 0 0 1 0 0 1 0 0 0 0 0 0 1 0 0 0 1 1 0 0 1 1 1 1 0 0
 1 1 1 1 1 0 0 1 1 0 0 1 0 0 0 1 0 1 0 1 0 0 0 0 0 1 0 1 0 0 0 1 1 0 1 0 0
 0 0 1 1 1 0 0 0 0 0 0 1 0 1 0 0 1 0 1 0 0 0 0 1 1 1 0 1 0 0 0 1 0 1 0 0 1
 1 0 0 1 1 1 1 1 1 1 0 1 1 0 0 0 1 1 0 1 1 0 1 1 1 1 1 0 1 1 0 1 1 1 1 0 1
 0 1 1 0 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 0 1 1 1 0 1 1 0 1 0 1 1 1 0
 1 1 1 1 0 1 0 1 1 1 1 1 1 1 0]
Test errors: 70
Test errors: 35.0 %

Task 3. Analyze what effect a training set size has on the *mse*, training and testing errors. Explain the results.

When using **Training set size = [50]**

Class observed
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1.]
Class predicted
[0 0 1 0 0 0 1 0 0 0 0 0 1 0 0 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 0 0 0 0 1 0 1
0 1 0 0 1 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 1 1 0 1 1 0 1 0 1 0 0 0 0 1 1 0 0
0 1 1 1 1 0 0 0 0 1 1 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1]
Training errors: 39
Training errors: 39.0 %   For class 1
Class observed
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
1. 1. 1. 1. 1. 1. 1. 1.]
Class predicted
[0 0 1 1 0 0 0 0 0 0 1 0 1 0 1 0 0 1 0 0 1 0 0 1 0 1 1 1 0 0 0 0 1 1 1 0 0
0 1 1 0 0 0 0 0 0 0 0 1 1 0 1 0 0 1 0 1 1 0 0 1 0 1 0 1 0 0 1 0 1 0 0 1 0
0 0 0 0 0 0 0 0 1 0 0 0 1 1 0 1 0 1 1 0 0 0 0 0 0 1 0 1 0 1 1 0 1 1 1 1 0
0 1 0 0 1 1 0 1 0 0 1 0 0 0 1 1 1 1 0 1 1 1 1 1 0 0 1 1 1 0 0 0 1 0 1 0 1
1 1 0 1 1 1 1 1 1 1 1 1 1 0 0 1 1 0 1 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1
1 1 0 1 1 1 1 0 1 0 0 0 1 1 1]
Test errors: 67
Test errors: 33.5 %   class-B
.
Task 4. Analyze what effect a distance between the classes has on the *mse*, training and testing errors. Distance between the classes is presented as Mahalanobis distance. The larger the distance, the better class separability is.

Change the mean values and correlation coefficient of the classes:

```
mean1 = np.array([-0.5, 1])  #  mean vector class 2
mean2 = np.array([0.5, 0])   #  covariance matrix class 2
var1=1 # variance of x1 feature
var2=2 # variance of x2 feature
cor12=0.8 #correlation coefficient between x1 and x2
```

You can also make the class covariance matrices cov1 and cov2 not equal.

Mean = [0.8, 1] and Covariance = [0.5,1]

Class observed
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1.]
Class predicted
[1 0 1 0 0 1 1 0 0 1 0 1 0 0 0 0 0 0 0 1 1 0 0 1 0 0 1 1 0 0 0 0 1 0 0 1 1
 0 1 0 0 0 1 1 0 0 0 0 0 0 1 0 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 0 0 1 1
 1 1 0 1 1 1 1 0 0 0 1 1 1 0 0 1 1 1 0 0 0 1 1 0 0 0]
Training errors: 35
Training errors: 35.0 %
Class observed
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1.]
Class predicted
[0 0 1 1 1 0 0 1 0 0 0 0 1 0 1 1 1 1 1 0 1 0 1 0 0 0 0 1 0 0 0 1 0 0 1 0 0
 0 0 1 1 0 0 1 0 0 1 1 0 0 0 0 0 0 1 0 1 0 0 1 1 0 1 1 1 1 1 1 0 0 1 0 1 0
 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 0 1 1 0 0
 1 1 0 1 0 0 0 1 0 0 0 0 1 1 1 1 1 1 0 1 1 0 0 0 0 1 1 1 0 1 1 0 0 1 1 0 1 0
 1 1 0 1 0 0 1 1 1 0 1 0 0 0 1 1 1 0 0 1 0 1 0 1 0 1 1 1 0 1 0 1 0 0 1 1 0
 1 1 1 0 1 0 0 1 1 0 0 0 1 0 1]
Test errors: 91
Test errors: 45.5 %
Mahalanobis distance between classes: 0.3

Figure.1 – Class 1

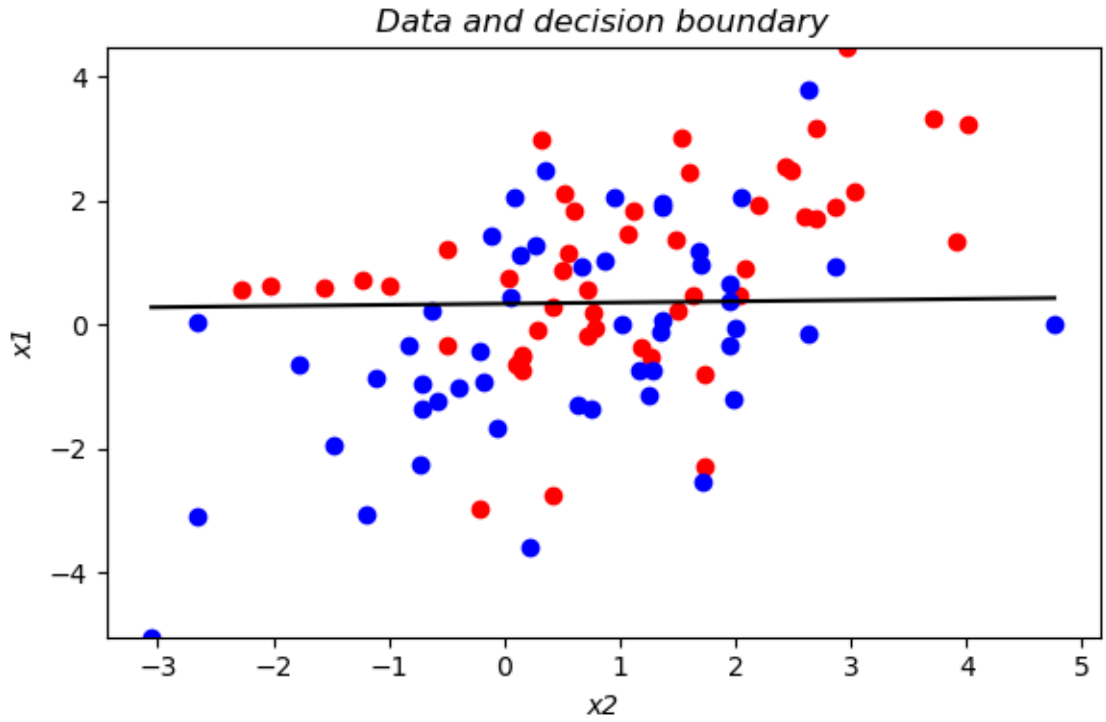
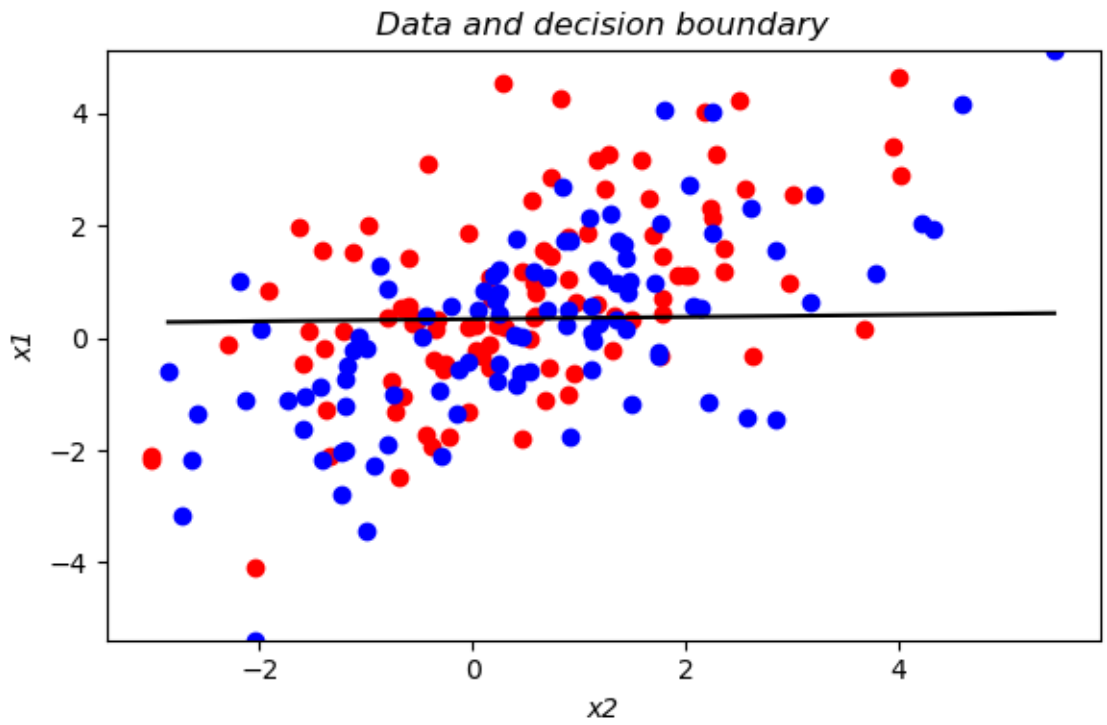
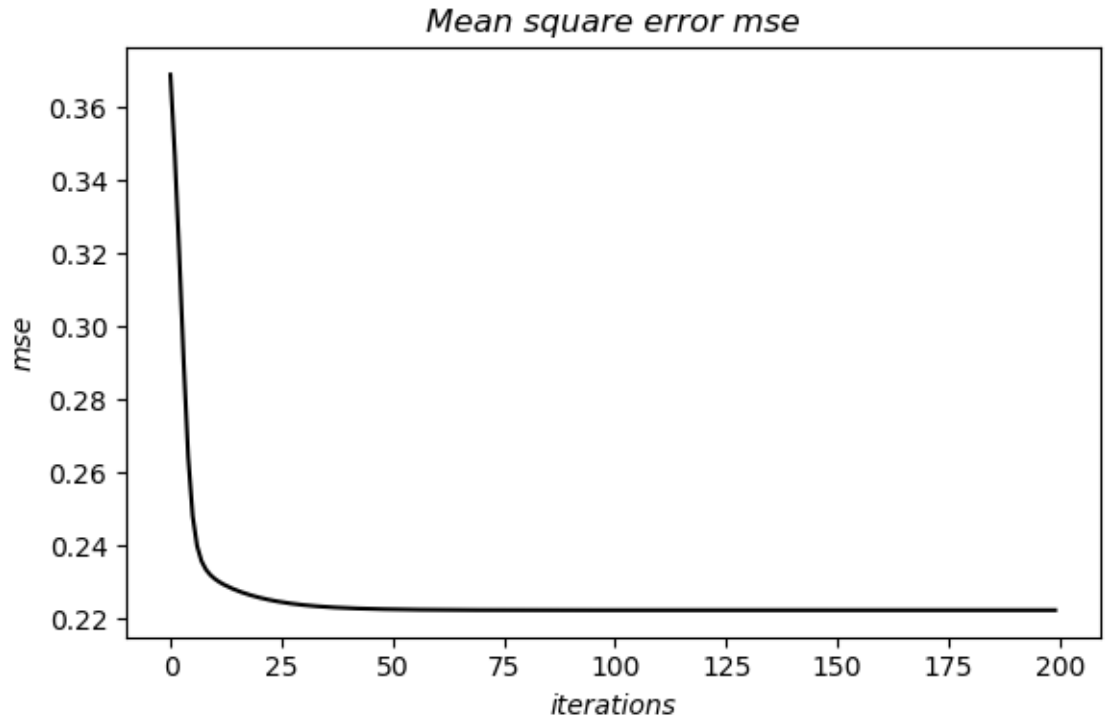


Figure 2 – Class 2





Task 5 (advanced). Calculate the training and test errors for each class.

Task 6 (not obligatory). Classify your own data.

Analyse the results and present in a written report.