Deep Learning for Optical Imaging

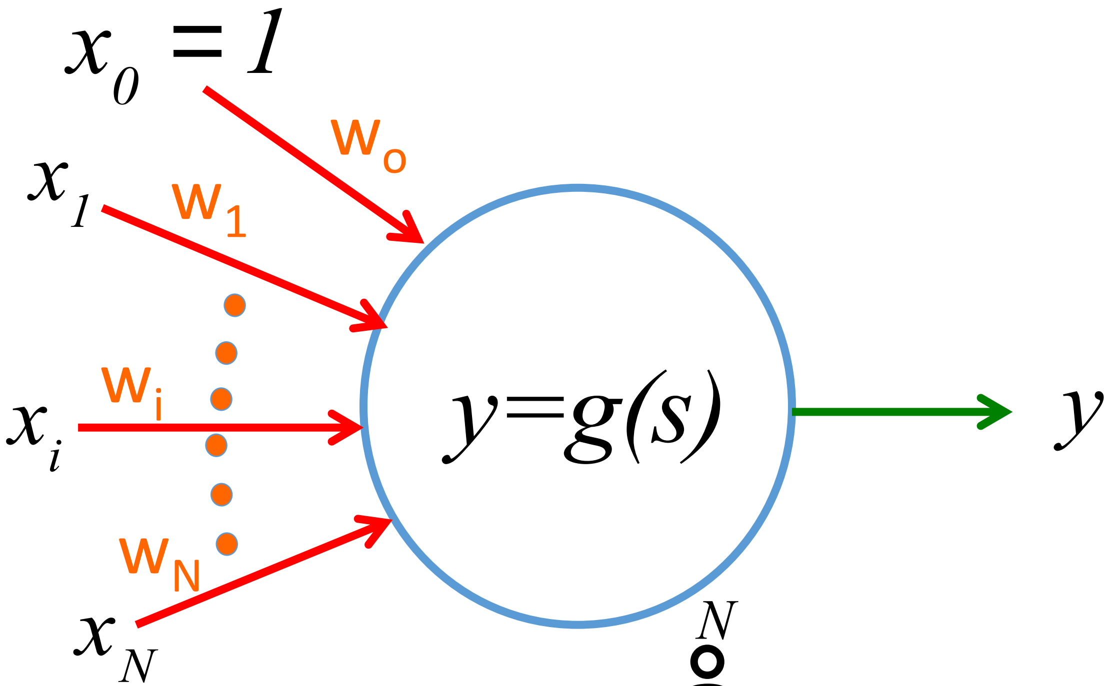
Lecture 2a

Single neuron
(continued)

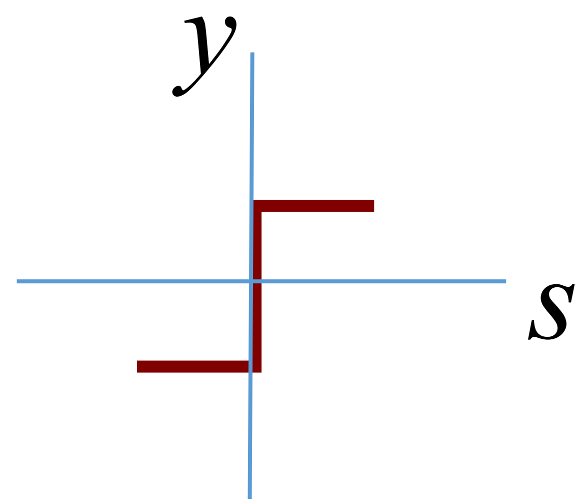
Outline

1. Perceptron review
2. Adaline
3. Assessment with the handwritten digits database
4. Adaline and regression

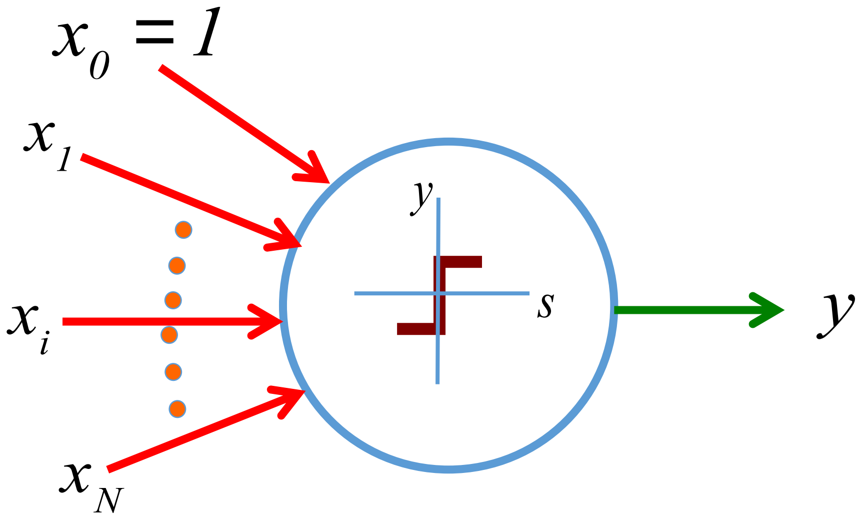
Single Neuron



$$s = \sum_{i=0}^{N} w_i x_i$$



Perceptron

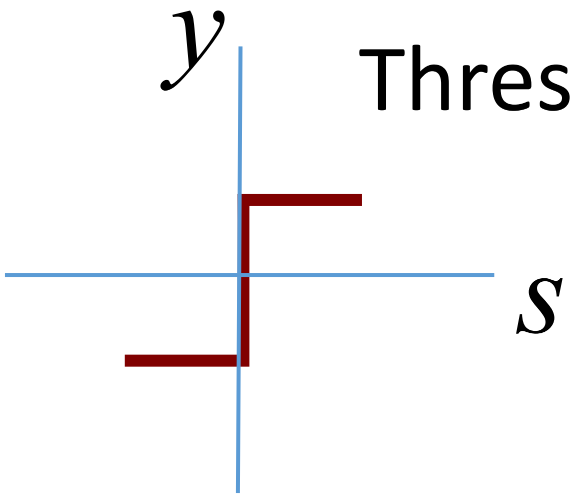


$$Training\ set\!:\left\{x^m, y^m\right\} \quad m = 1, M \quad y^m = \pm 1$$

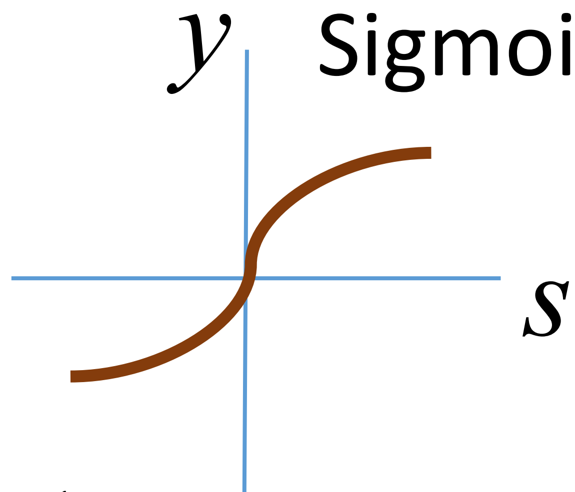
$$w_i^{t+1} = w_i^t + \mathrm{D}w_i^t \qquad \mathrm{D}w_i^t = \begin{cases} 0 & if\ \ y^m\left(\sum_{i=1}^{N} w_i x_i^m + w_0\right) \geq 0 \\ y^m x_i^m & otherwise \end{cases}$$

Activation Functions

Threshold

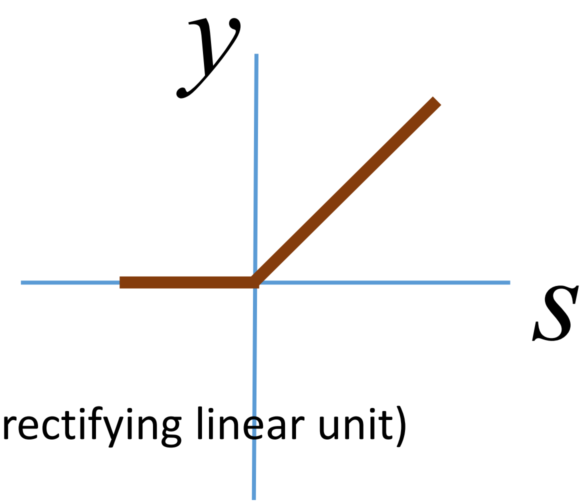


Sigmoid



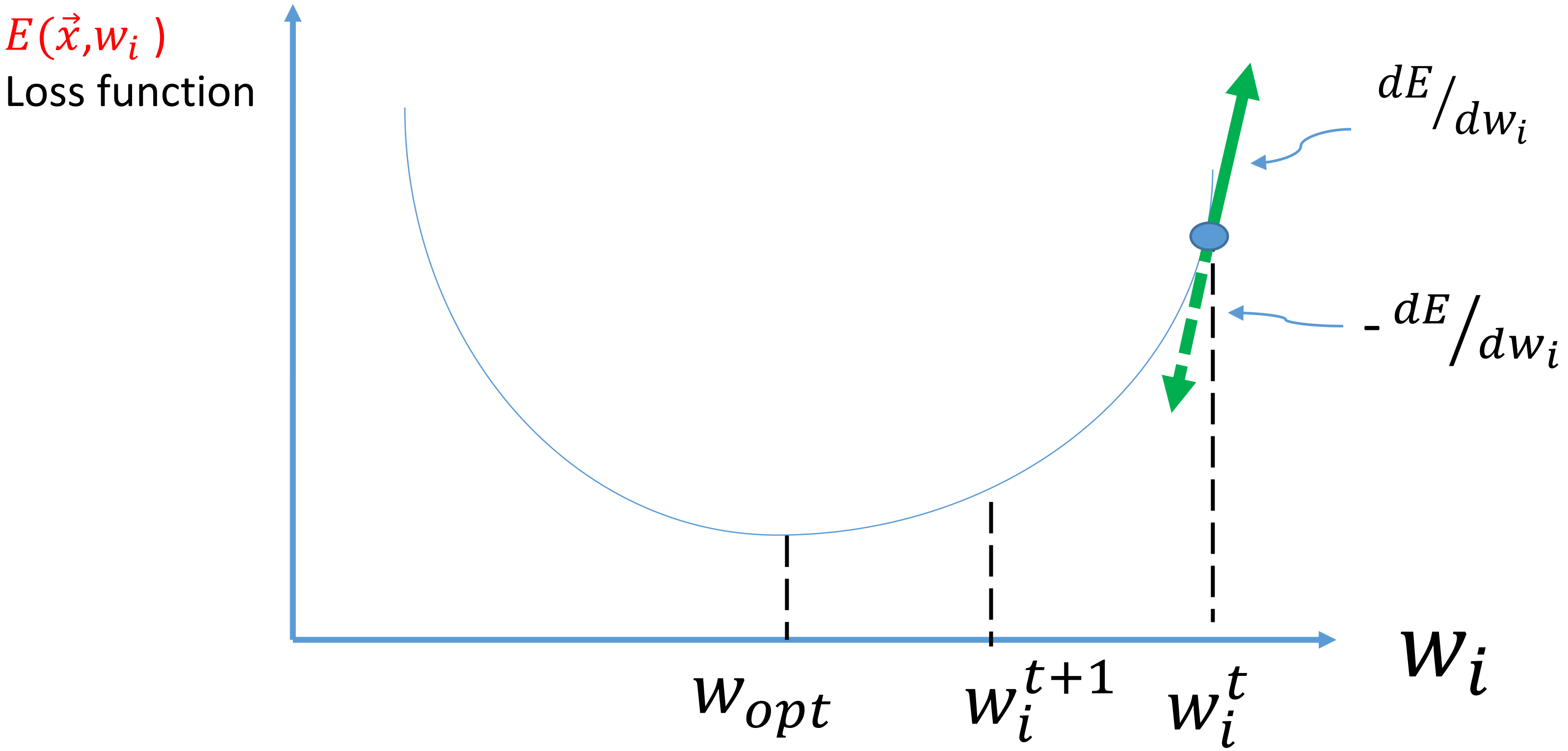
$$g(s) = \frac{1}{1 + e^{-\alpha s}}$$

ReLu



(rectifying linear unit)

Optimization: Steepest decent



$$w_i^{t+1} = w_i^{t+1} + \Delta w_i$$

$$\Delta w_i \sim - {}^{dE}/_{dw_i}$$

MSE loss (energy) function

$\boxed{Training\ set.\ \ (x_i^m, y^m)}$

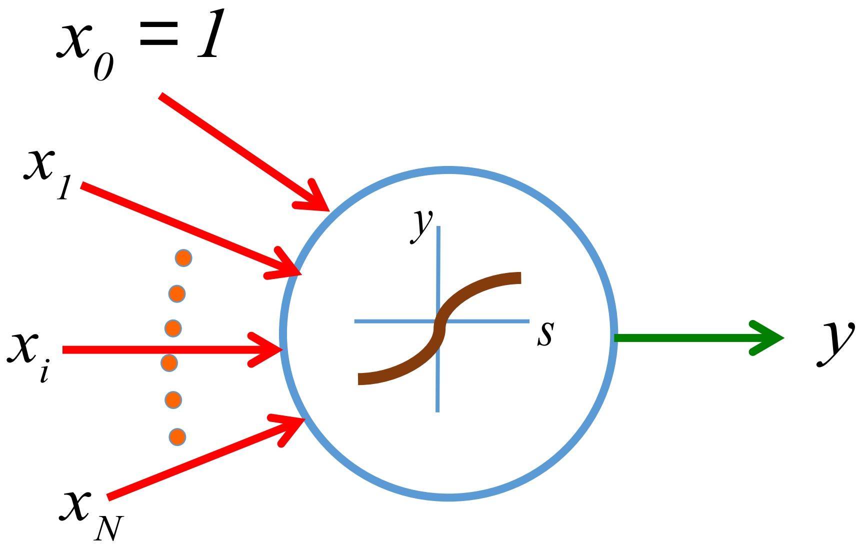
$$\bar{y}^m = g\{\textstyle\sum_i^N w_i\, x_i^m\}$$

$$E(x,w) = \sum_m^M (y^m - \bar{y}^m)^2$$

$$dE/_{dw_i} = 2\ \textstyle\sum_m^M (y^m - \bar{y}^m)\ \frac{d\bar{y}^m}{dw_i}$$

$$= 2\ \underbrace{\textstyle\sum_m^M (y^m - \bar{y}^m)}_{s} \frac{dg}{ds} x_i^m$$

Adaline



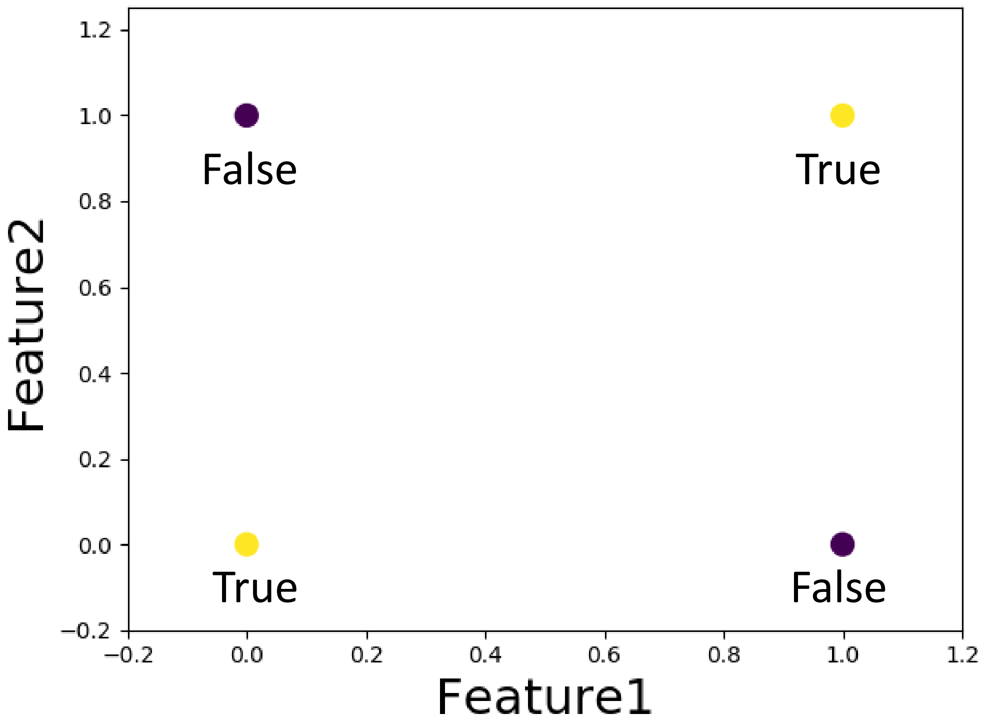
$$Training\ set: \left\{ x^m, y^m \right\} \quad m = 1, M \quad y^m = \pm 1$$

$$w_i^{t+1} = w_i^t + \Delta w_i^t$$

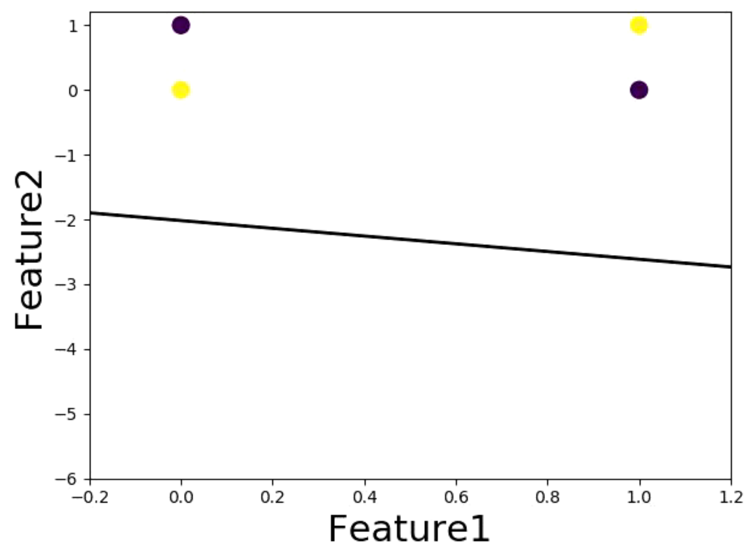
$$\Delta w_i^t = \alpha \frac{dg}{ds}\frac{ds}{dw_i}(y^m - y) = \alpha \frac{dg}{ds}(y^m - y)x_i^m$$

$$\alpha > 0$$

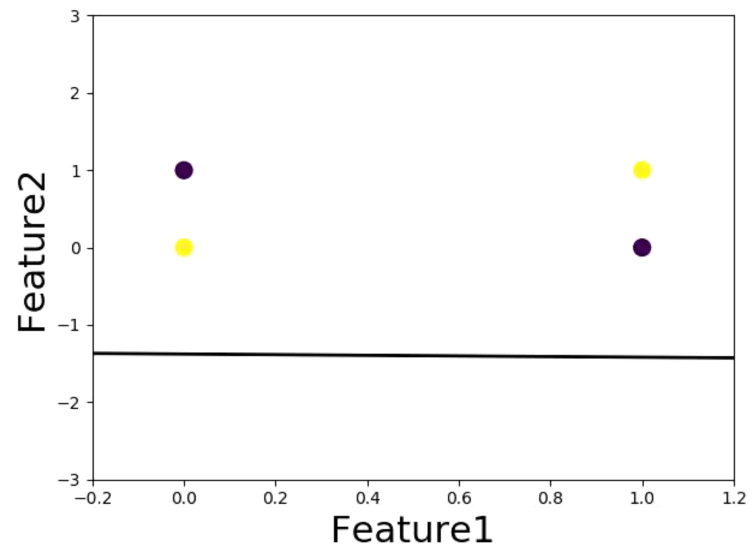
Perceptron versus Adaline for XNOR



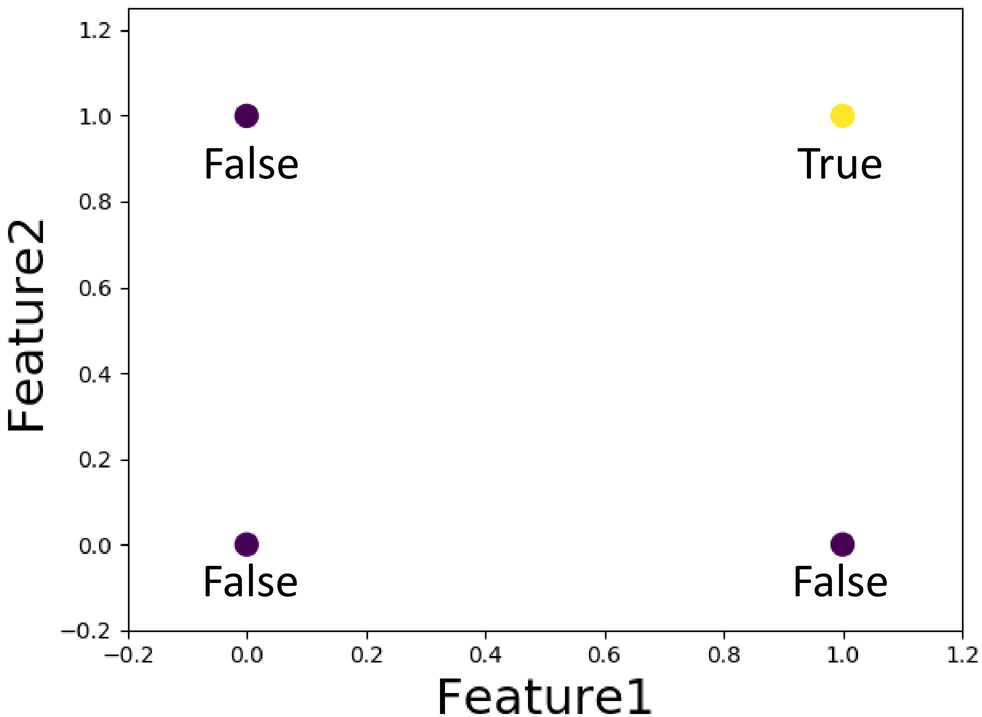
Perceptron



ADALINE

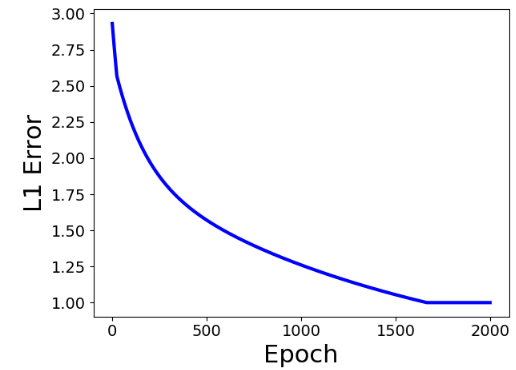


ADALINE AND gate : Linear activation function



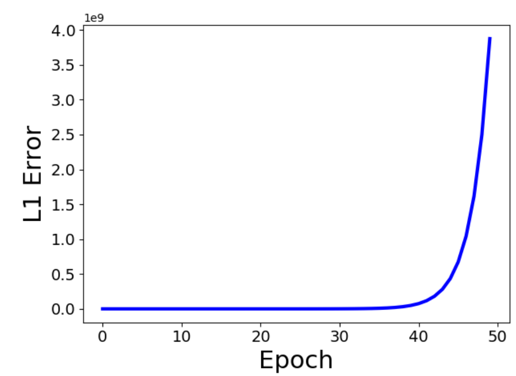
Learning rate = 0.001

Accuracy = 100%



Learning rate = 0.4

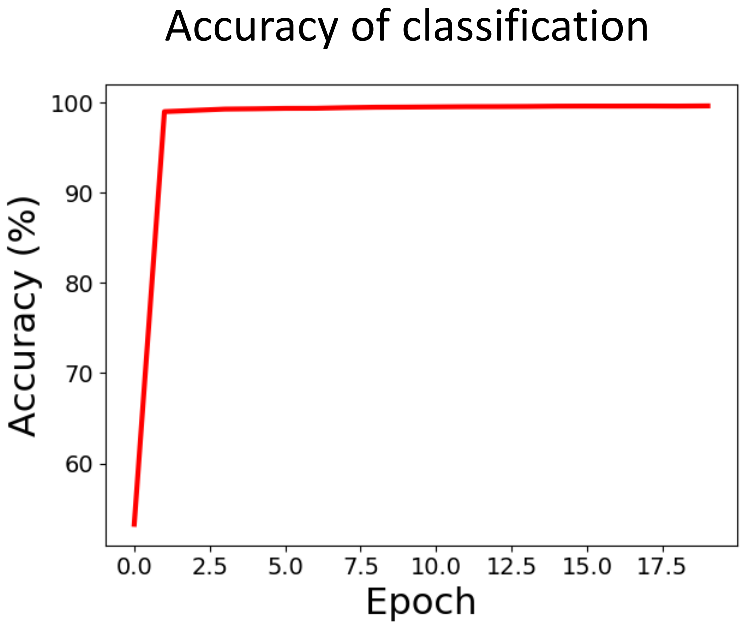
Accuracy = 25%

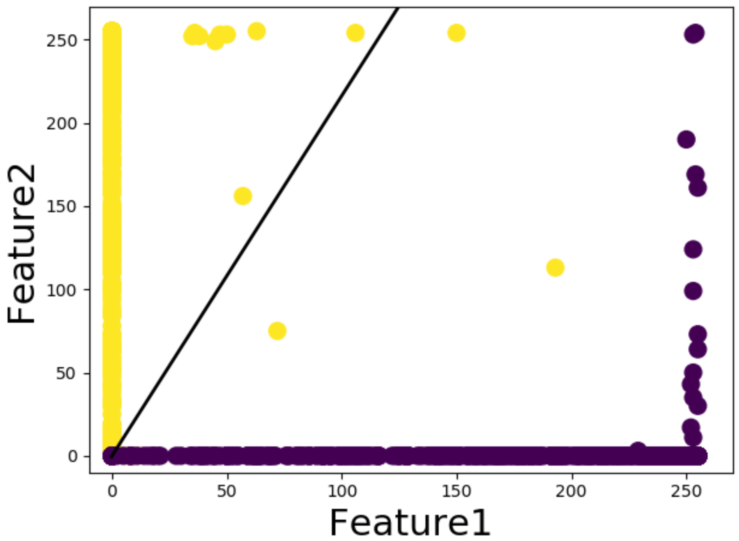


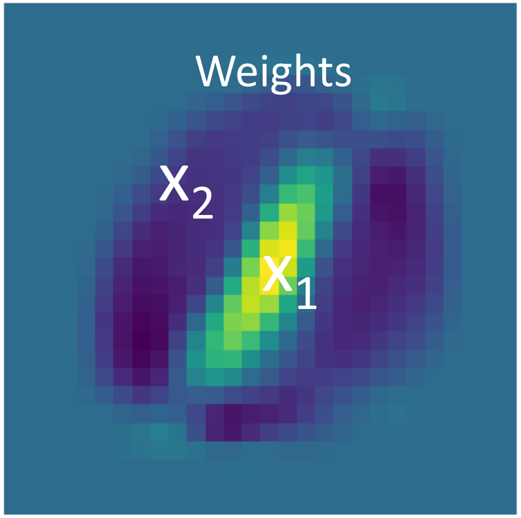
$$w_i^{t+1} = w_i^t + \Delta w_i^t$$

$$\Delta w_i^t = \alpha \frac{ds}{dw_i}(y^m - y) = \alpha (y^m - y) x_i^m$$

Binary classification of digits-ADALINE







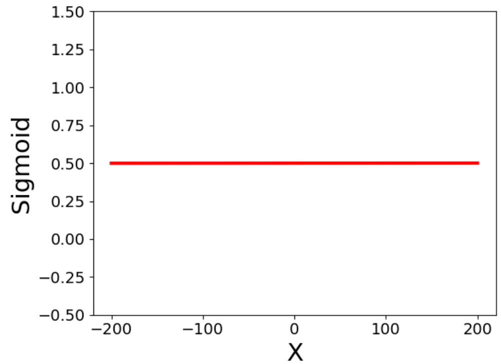
Training accuracy	99.66%
Test accuracy	99.95%

12665 Training samples
2115 Test samples
Sigmoid activation function

Face classification accuracy on the training set (Adaline)

Sigmoid Slope = 0.00001

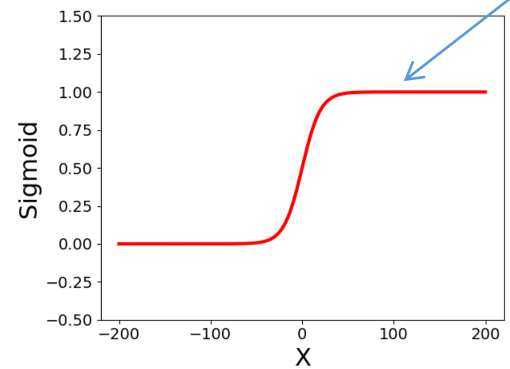
Accuracy = 100%



Sigmoid Slope = 0.1

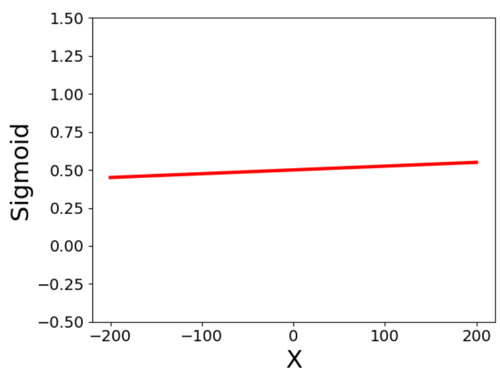
Accuracy = 50%

$$\frac{dg}{ds} \approx 0$$



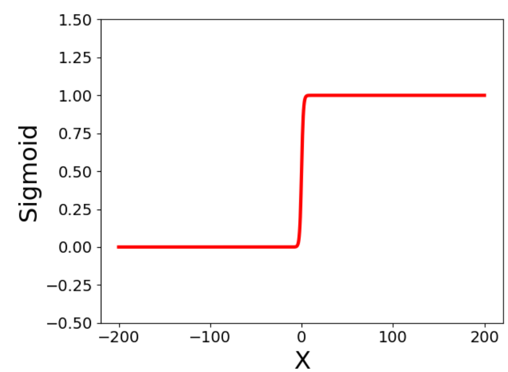
Sigmoid Slope = 0.001

Accuracy = 100%



Sigmoid Slope = 1

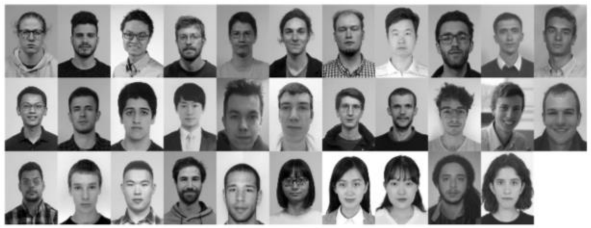
Accuracy = 50%



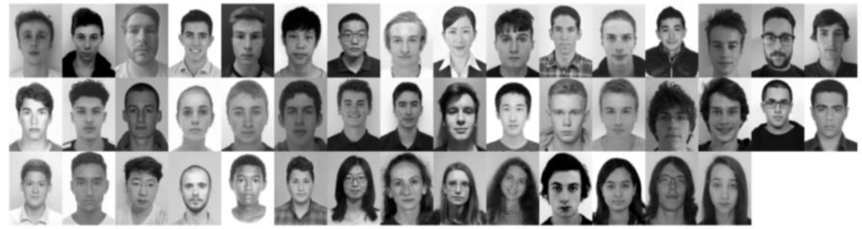
Face classification

Training:

Group 1
(2021)



Group 2
(2022)

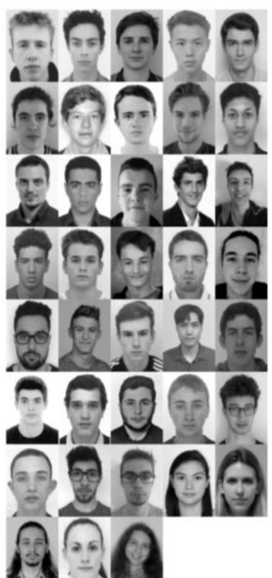


- ADALINE train accuracy : 80 %
- Perceptron train accuracy : 88 %

Perceptron

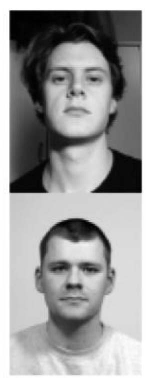
Group 1

Group 2



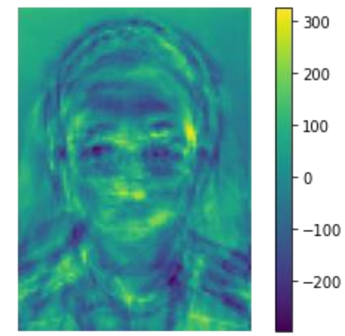
ADALINE

Group 1

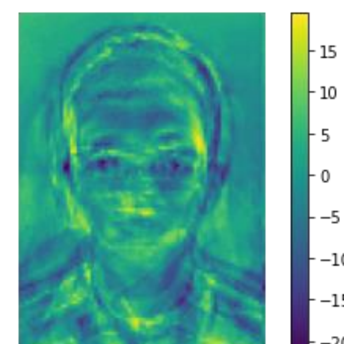


Group 2

ADALINE weights

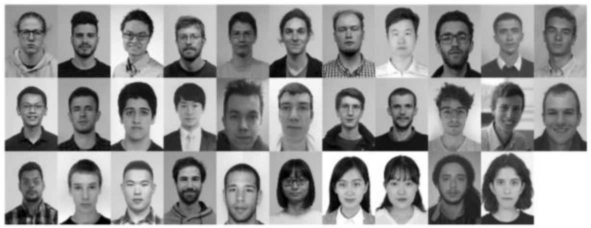


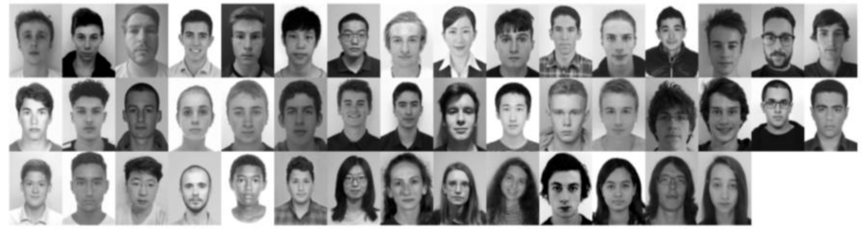
Perceptron



Face classification: Long hair/Short hair

Training: Images of Students in 2021 and 2022



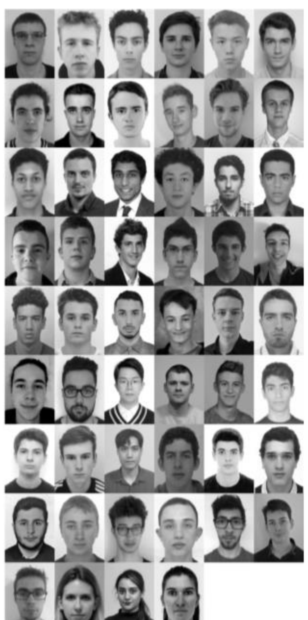


- ADALINE train accuracy : 100%
- Perceptron train accuracy : 100%

Perceptron (92% Accuracy)

Long Hair

Short Hair

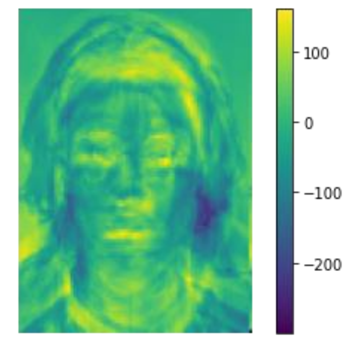


ADALINE (93% Accuracy)

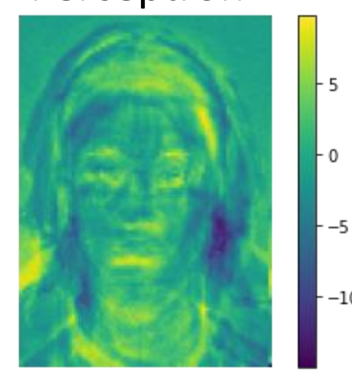
Long Hair

Short Hair

ADALINE weights



Perceptron



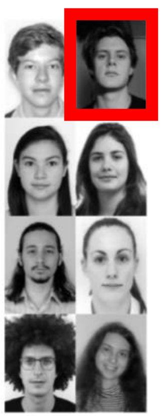
Face classification: Long hair/Short hair

Training: Images of Students in 2021 and 2022

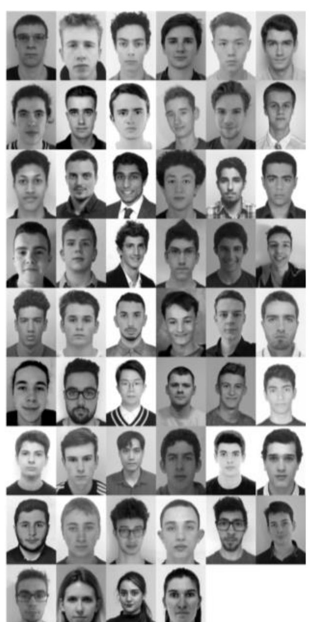
- ADALINE train accuracy : 100%
- Perceptron train accuracy : 100%

Perceptron (92% Accuracy)

Long Hair



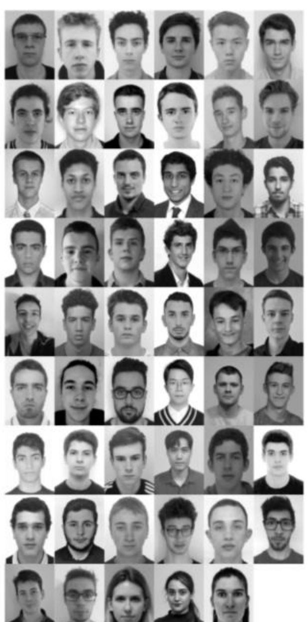
Short Hair



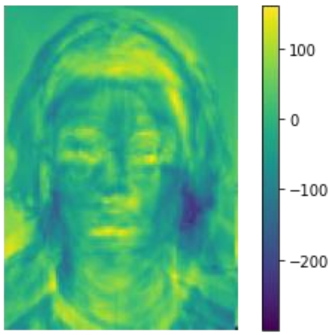
ADALINE (93% Accuracy)

Long Hair

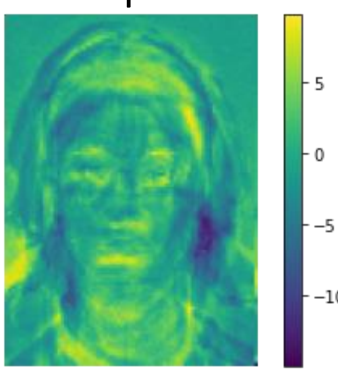
Short Hair



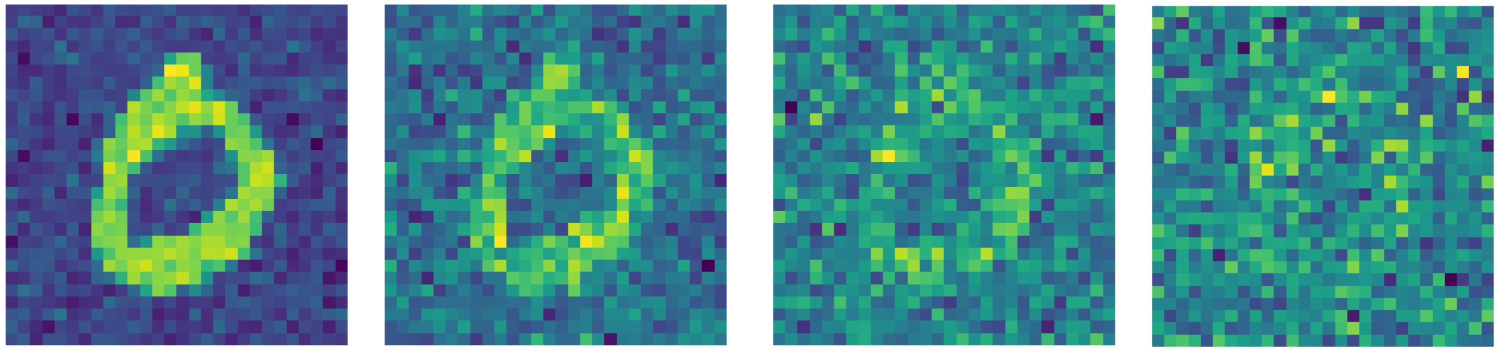
ADALINE weights



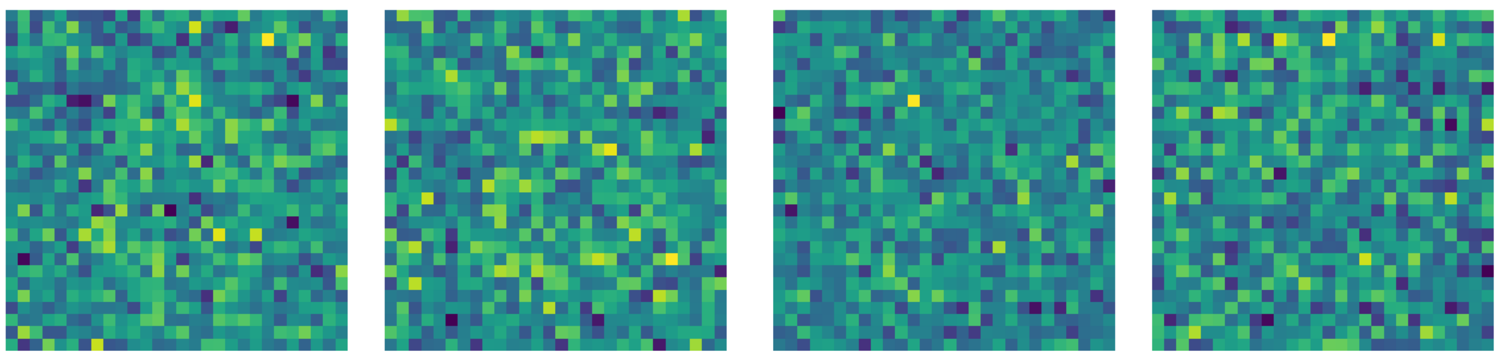
Perceptron



Digits: Gaussian noise



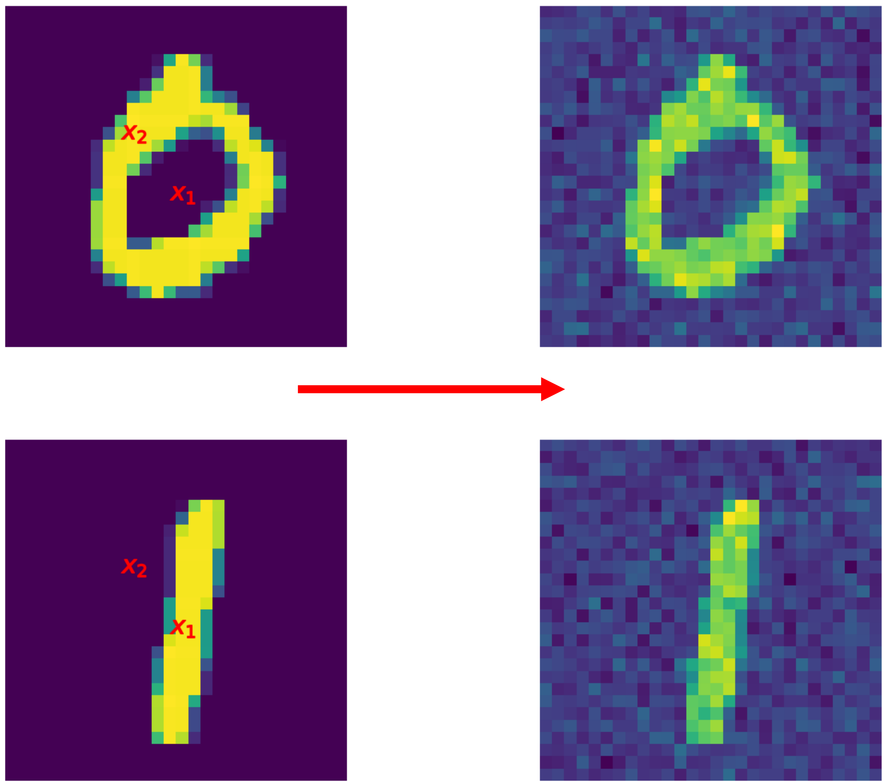
Perceptron	99.81%	99.76%	99.48%	98.30%
SNR	24.28	3.87	2.38	1.30



Perceptron	95.41%	90.07%	84.44%	76.88%
SNR	0.85	0.61	0.36	0.26

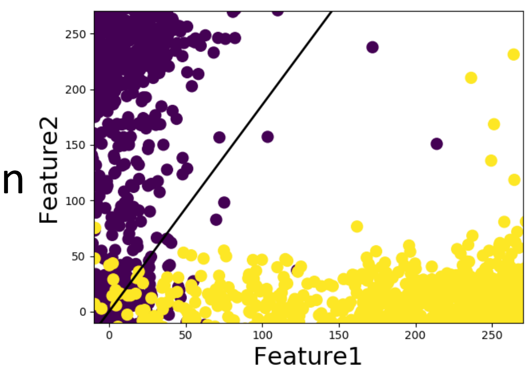
12665 Training samples
2115 Test samples

Digits: Gaussian noise

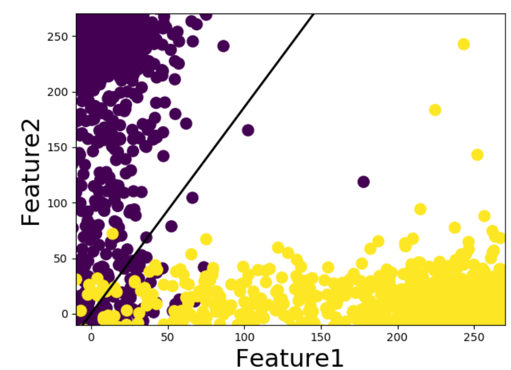


Perceptron - Test	99.78%
ADALINE - Test	99.81%

Perceptron



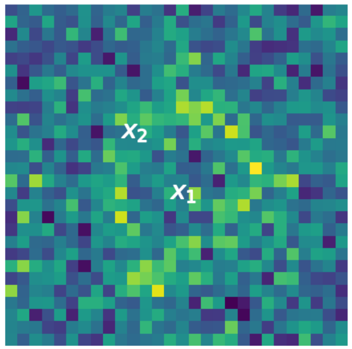
ADALINE

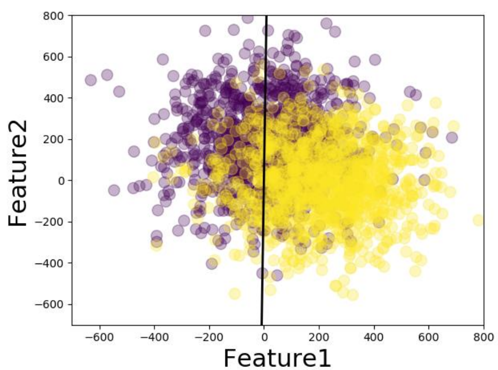


12665 Training samples
2115 Test samples
Sigmoid activation function is used for ADALINE

Digits: Gaussian noise (perceptron)

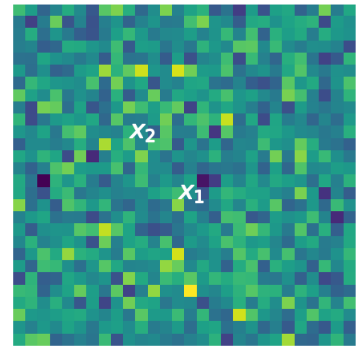
SNR 1.3

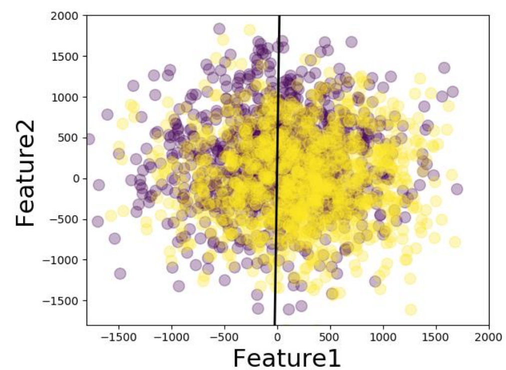




Test accuracy : 98.35

SNR 0.26



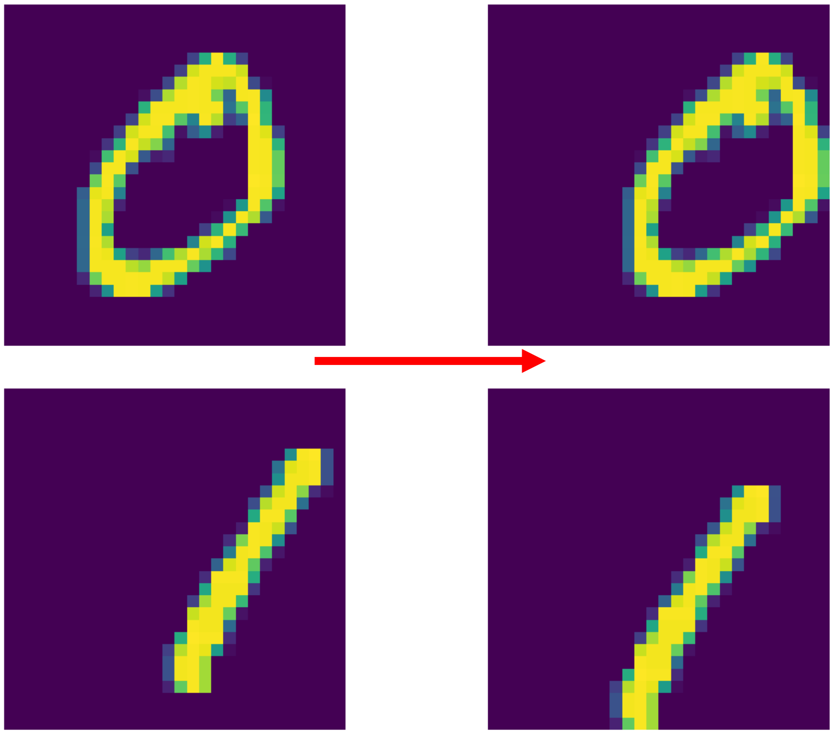


Test accuracy : 79.39

12665 Training samples
2115 Test samples

Digits: Shift

Images in test set are moved randomly in different locations



Perceptron - Test	61.70%
ADALINE - Test	59.95%

Including shifted images in the training

Perceptron - Test	85.72%
ADALINE - Test	88.56%

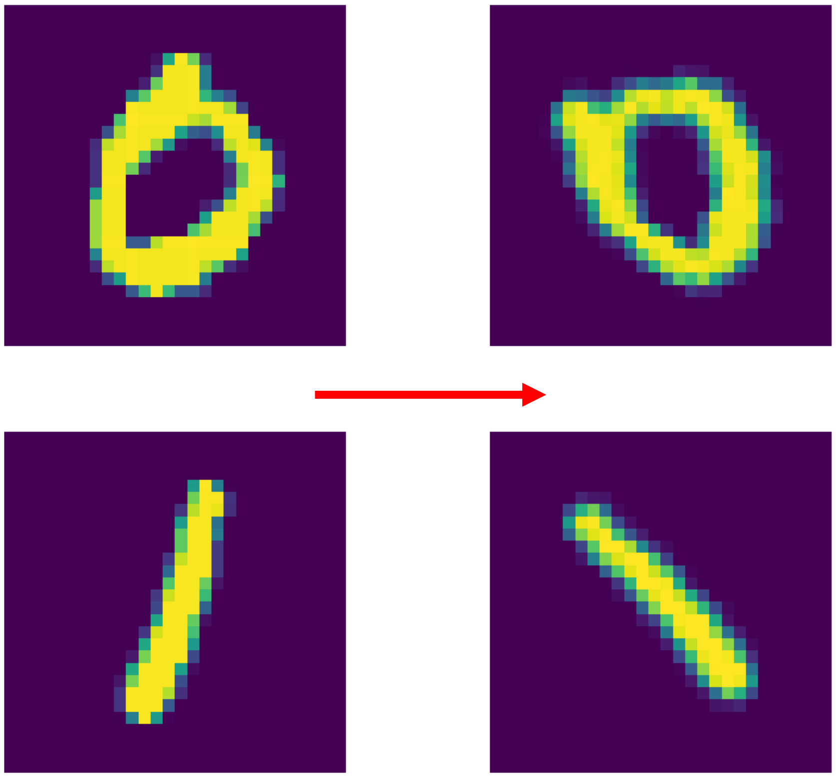
It needs more iteration when the shifted images are included in the training.

12665 Training samples
2115 Test samples
Sigmoid activation function is used for ADALINE

Digits: Rotation (in the test set or the training)

Images in test set are rotated randomly between 60° to 90°.

Perceptron - Test	51.63%
ADALINE - Test	51.54%



Including rotated images in the training

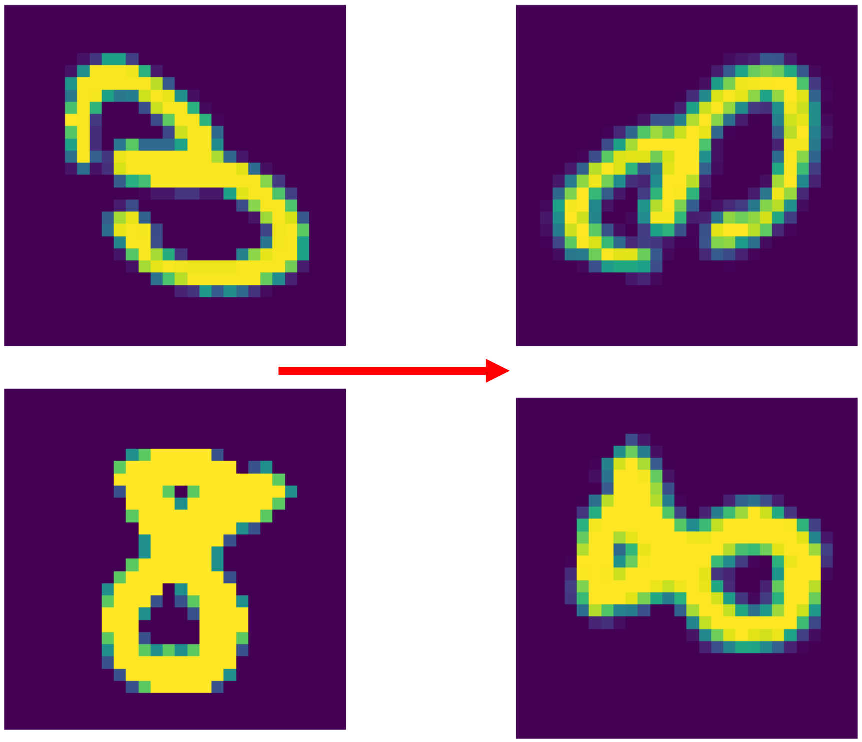
Perceptron - Test	99.67%
ADALINE - Test	99.86%

It needs more iteration when the rotated images are included in the training.

12665 Training samples
2115 Test samples
Sigmoid activation function is used for ADALINE

Digits: Rotation

Images in test set are rotated randomly between 60° to 90°.



Test with rotated images

Perceptron - Test	46.88%
ADALINE - Test	44.41%

Including rotated images in the training

Perceptron - Test	51.06%
ADALINE - Test	48.94%

11198 Training samples
1984 Test samples
Sigmoid activation function is used for ADALINE

Direct inversion

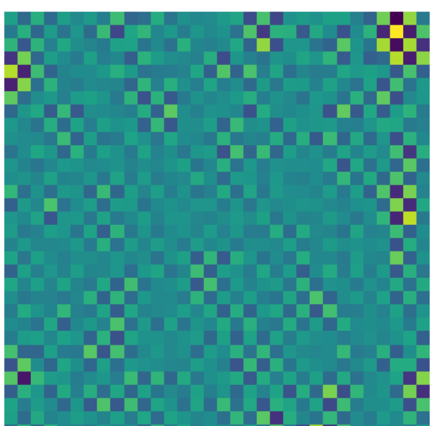
Weight calculation by matrix inversion

$$\underline{\underline{X}}\,\underline{w} = \underline{t} \Rightarrow \underline{w} = \underline{\underline{X}}^{-1}\underline{t}$$

$\underline{\underline{X}}$ is 1024 by 1024

Images used for weight calculation	100%
Test 2000 new images	52%

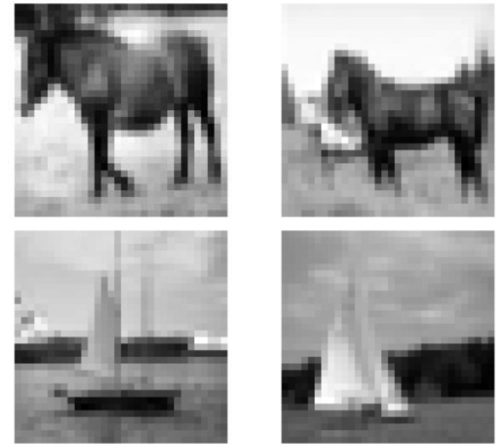
Weights ($\underline{w}$)



Database
1024 training images
2000 test images

Class 1

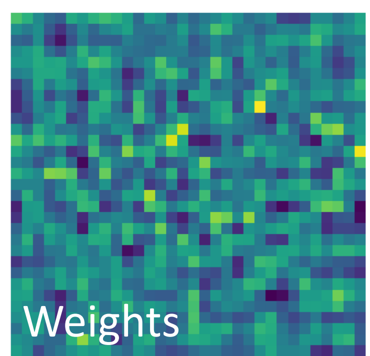
Class 2



Perceptron

Training accuracy: 51.00%
Test accuracy: 50.50%

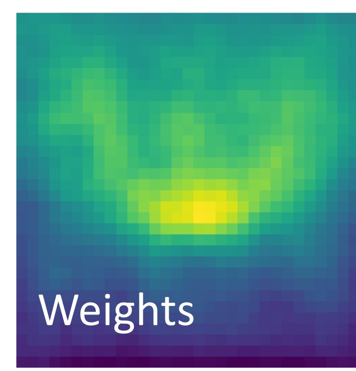
1000 iterations



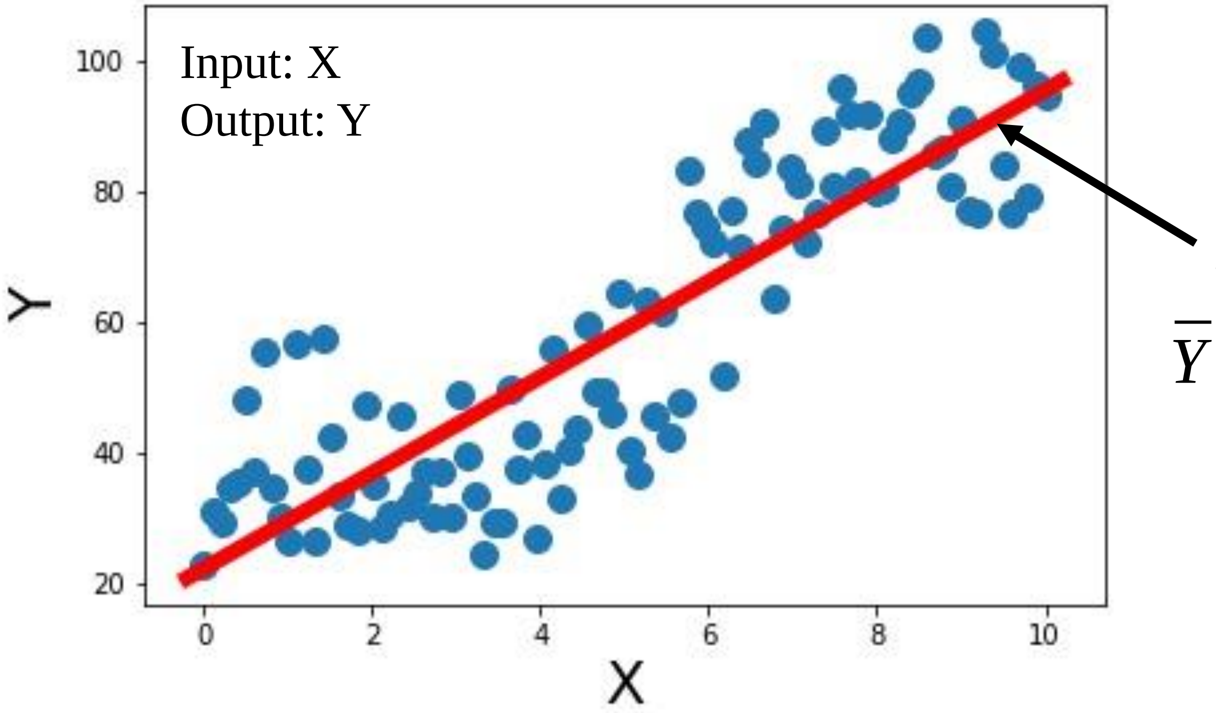
ADALINE

Training accuracy: 73.44%
Test accuracy: 71.50%

Learning rate = 0.0001
2000 iterations



Regression

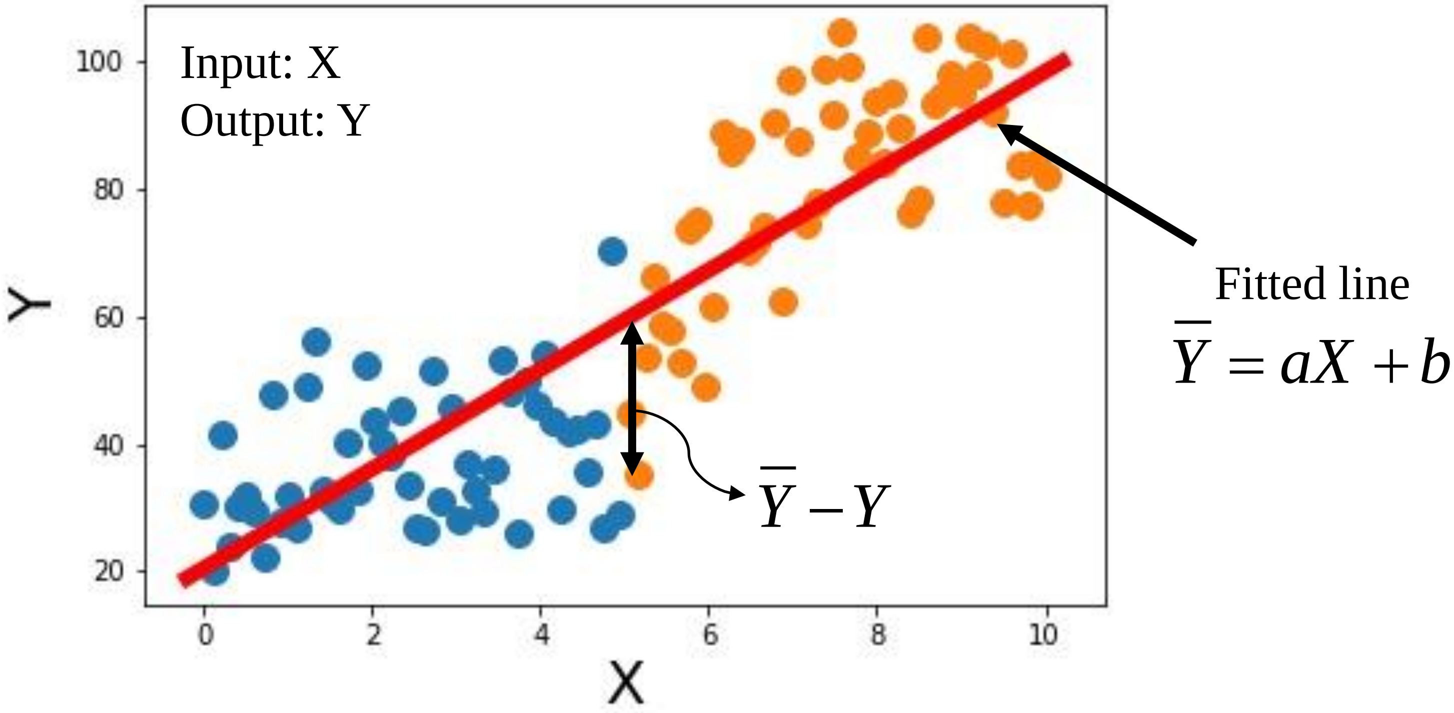


Fitted line

$$\overline{Y} = aX + b$$

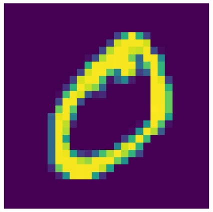
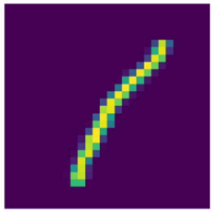
What is the difference between linear regression and ADALINE?

Regression

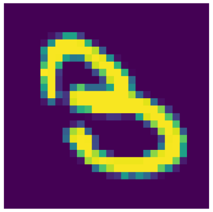
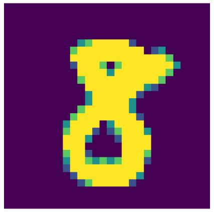


Input: X
Output: Y
100
80
60
40
20
0
2
4
6
8
10
Y
X
$\overline{Y} - Y$
Fitted line
$\overline{Y} = aX + b$

Regression vs. ADALINE: digit classification

ADALINE - Test	99.91%
Regression - Test	99.29%

ADALINE - Test	96.57%
Regression - Test	96.02%

ADALINE:

- Sigmoid activation function
- Sigmoid slope = 0.0001
- Learning rate = 0.0001
- Epoch = 200

A threshold function is used after the regression in order to classify the outputs.

Regression

Linear regression finds the best linear fit relationship between the input variables (x) and the single output (y).

$$y^{(m)} = \sum_{i}^{N} \beta_i x_i^{(m)} = \vec{\beta}.\overrightarrow{x^{(m)}}$$

The model parameters (β) can be calculated using least-squares estimation:

$$\vec{\beta} = \min\left(\sum_{m}^{M} (\vec{\beta}.\overrightarrow{x^{(m)}} - y_i)^2 \right)$$

We can put input and out variables in matrices X and Y.

$$\vec{\beta} = \min\left(\left(X\vec{\beta} - Y \right)^2 \right)$$

The optimum model parameter (β) lies at gradient zero:

$$\frac{\partial \left[\left(X\vec{\beta} - Y \right)^2 \right]}{\partial \vec{\beta}} = 0 \rightarrow \vec{\beta} = (X^T X)^{-1} X^T Y$$

The index *m* is used for the samples. *M* is the total number of the samples. N is the dimension of the input.