

⇒ (linear classification vs. linear regression
→ generalized linear model ↳ 확장된
↳ 연속적인 출력.

- ✓ • perceptron
- ✓ • logistic regression

Preclass 03: Linear Classification

[SCS4049] Machine Learning and Data Science

Seongsik Park (s.park@dgu.edu)

School of AI Convergence & Department of Artificial Intelligence, Dongguk University

Linear classification

MNIST dataset

- 70,000 images of 28×28 handwritten digits from 0 to 9
- “Hello, World!” of machine learning

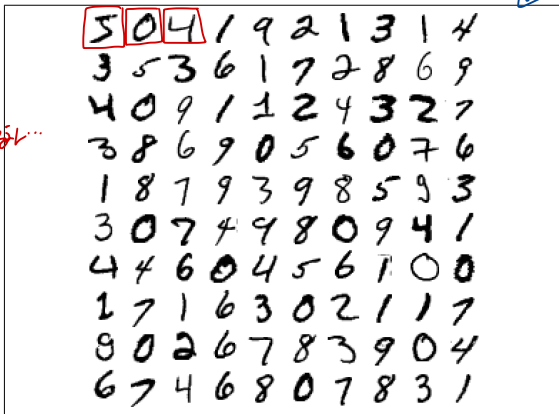
★ class

= 0~9, 숫자들

= 사람, 고양이, 자동차...

★ # class

→ class 인스.



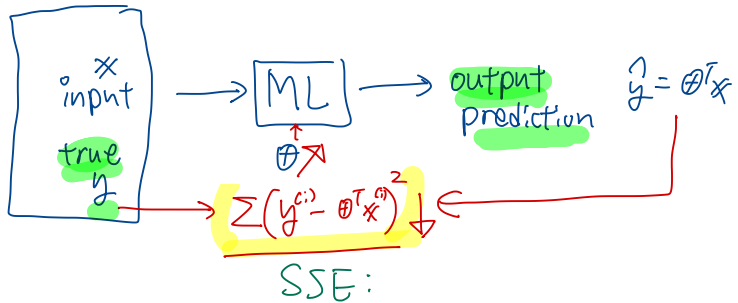
손글씨
image



classify

0~9
숫자

Figure 3-1. A few digits from the MNIST dataset



Classification problem

The goal in classification

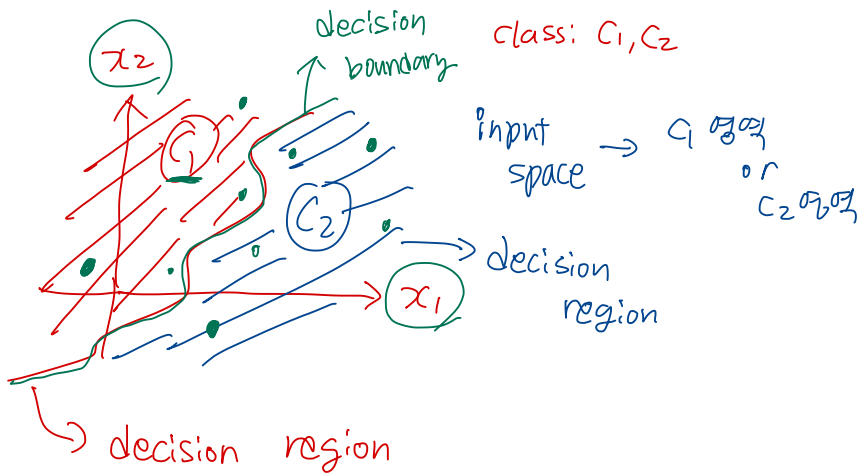
?input $\times \rightarrow ML \rightarrow 1, 2, \dots, K$ class
 $\sum_{i=1}^K \sigma_i = 1$ class prob
174.

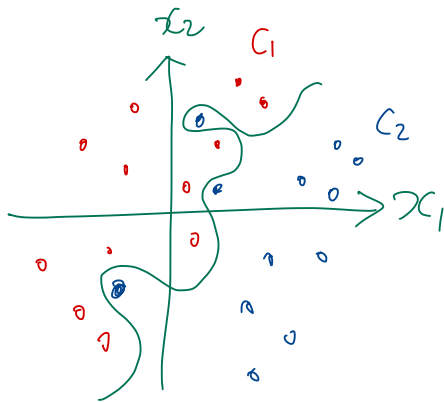
- Take an input vector $\mathbf{x}$
- Assign it to one-of-K discrete class C_k where $k = 1, 2, \dots, K$
- Assign each input to one and only one class (disjoint) $\Leftarrow$

The input space divided into decision region, whose boundaries are called decision boundaries or decision surfaces. ∇

- (D-1)-dimensional hyperplanes within the D-dimensional input space
- Linearly separable dataset that can be separated exactly by linear decision surface







linear surface $\Rightarrow$ 직선

모든 sample을
저기대론 구분해내는.

Classification: representation

Target output t → 어느 class 인가?

$$t \in \{-1, +1\}$$

perception • Two-class problem, $t \in \{0, 1\}$, e.g., $t = 1$ represents C_1

→ • $K > 2$ classes, $t \in \{0, 1\}^{(K)}$ and $\sum_k t_k = 1$ [one-hot coding
one-of-1C coding]

logistic regression → • t_k is probability of C_k and $\sum_k t_k = 1$

Two-class

$$t = \begin{cases} +1 \\ 0 \end{cases}$$

$\underline{C_1}$
 $\underline{C_2}$

$$t = \begin{cases} +1 \\ -1 \end{cases}$$

$\underline{C_1}$
 $\underline{C_2}$

multiclass $k=5$

$$t = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \end{bmatrix}^T \leftarrow$$

$$t = \begin{bmatrix} 0.1 & 0.8 & 0.05 & 0.05 & 0 \end{bmatrix}^T \leftarrow$$

Confusion matrix

Confusion matrix (contingency matrix)

☆

		Predicted	
		Negative	Positive
Ground truth	Negative	True Negative	False Positive (Type I Error)
	Positive	False Negative (Type II Error)	True Positive

←

metric

True positive rate (TPR): $TPR = \frac{TP}{TP + FN}$ (recall, sensitivity, hit rate) = $\frac{TP}{\text{전체 pos}}$

Positive predictive value (PPV): $PPV = \frac{TP}{TP + FP}$ (precision) = $\frac{\text{예측 pos}}{\text{맞은 경우의 수}}$

Accuracy (ACC): $ACC = \frac{TP + TN}{TP + FN + TN + FP}$ = $\frac{\text{전체 정답의 수}}{\text{전체 경우의 수}}$

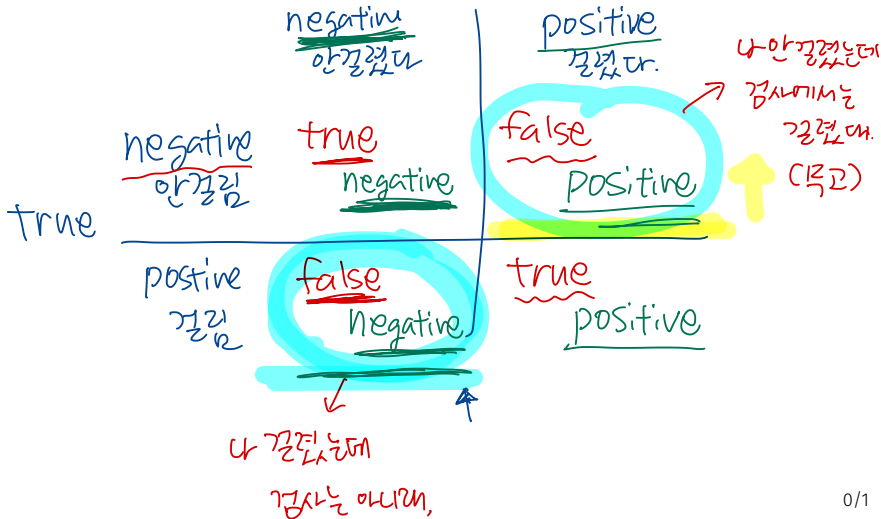
True negative rate (TNR): $TNR = \frac{TN}{TN + FP}$ (specificity)

False negative rate (FNR): $FNR = 1 - TPR$

False positive rate (FPR): $FPR = 1 - TNR$

COVID 19 - test.

검사
prediction



		prediction	
		N	P
truth	N	10,000	10
	P	<u>1</u>	<u>100</u>

$$\underline{\underline{\text{accuracy}}} = \frac{10,000 + 100}{10,000 + 10 + 1 + 100}$$

Trade off

$$\begin{aligned} \nearrow \text{recall} &= \frac{100}{1 + 100} = \frac{TP}{TP + P} \\ \searrow \text{precision} &= \frac{100}{10 + 100} = \frac{TP}{TP + N} \end{aligned}$$

Linear discriminant function

Two classes

Linear model

$$y(\mathbf{x}) = \theta^T \mathbf{x} + \theta_0 \begin{matrix} \geq 0 & C_1 \\ < 0 & C_2 \end{matrix} \text{ two class.}$$

$\mathbf{x}$ is assigned to class C_1 if $y(\mathbf{x}) \geq 0$
to class C_2 otherwise

Decision surface

- θ determines the orientation
- θ_0 determines the location
- $-\theta_0/\|\theta\|$ is the normal distance from the origin

Linear discriminant function

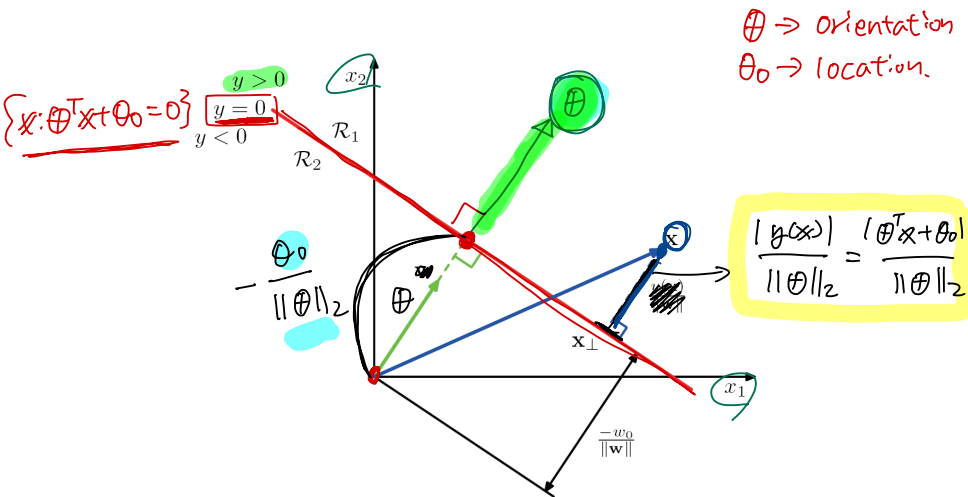


Figure 1: Decision surface of linear discriminant function

Normal equation for linear classification

Two classes

$$y(\mathbf{x}) = \theta^T \mathbf{x} + \theta_0 \rightarrow \text{예측값} \in \mathbb{R}$$

$\mathbf{x}$ is assigned to class $\mathcal{C}_1$ if $y(\mathbf{x}) \geq 0$
to class $\mathcal{C}_2$ otherwise

$t \in \{-1, +1\}$

Minimizing sum of squared errors $SSE \rightarrow$ normal equation.

$\rightarrow$ closed form solution for given a training dataset

$$\{(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2), \dots, (\mathbf{x}_m, t_m)\}$$

$$J(\theta) = \|\mathbf{X}\theta - \mathbf{t}\|_2^2$$

$$\text{where } \mathbf{t} \in \{0, 1\}^m \quad \mathbf{X} \in \mathbb{R}^{m \times (n+1)}$$

$$\text{then } \hat{\theta} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{t}$$

Is it working?

Least squares for classification

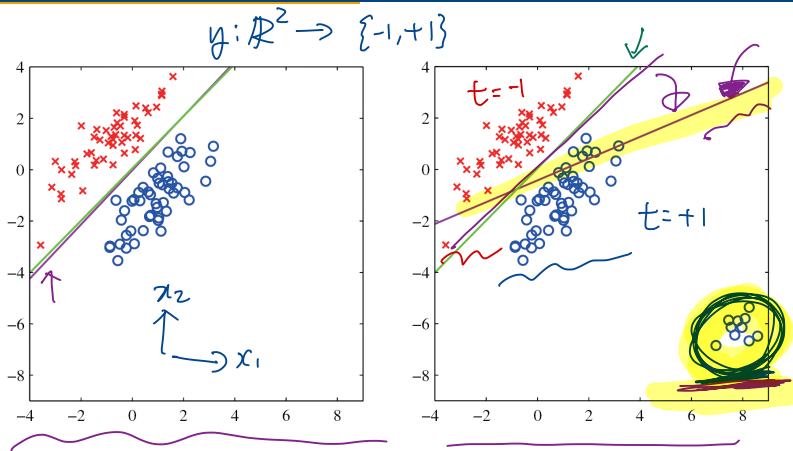
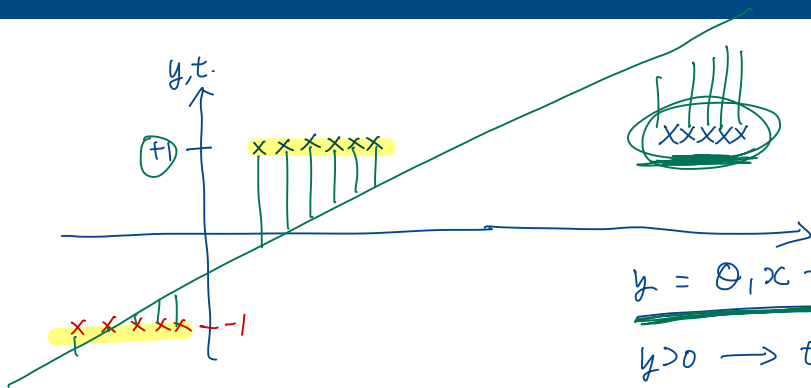


Figure 2: Two classes: least square and logistic regression

Normal.



$$y = \theta_1 x + \theta_0$$

$$y > 0 \rightarrow t = +1 \quad C_1$$

$$y < 0 \rightarrow t = -1 \quad C_2$$

$$J(\theta) = \text{SSE}(\theta).$$

truth

$t \leadsto$ labels, 분포, 범례(은지한)

classification



$$y = \theta^T x + \theta_0 \in \mathbb{R}$$

$$\sum_{\pm 1} \left(\underline{t} - \underset{\in \mathbb{R}}{(\theta^T x + \theta_0)} \right)^2$$

Generalized linear model

Generalized linear model

← perceptron
logistic regression

- $y(\mathbf{x}) = f(\theta^T \mathbf{x} + \theta_0)$ $\theta^T \mathbf{x} + \theta_0$
- Activation function $f: \mathbb{R} \rightarrow (0, 1)$
- Even if the function $f(\cdot)$ is nonlinear, the decision surfaces $\theta^T \mathbf{x} + \theta_0 = \text{const}$ are linear functions of $\mathbf{x}$

$$\begin{aligned} & \theta^T \mathbf{x} + \theta_0 \\ & f: \mathbb{R} \rightarrow \{0, 1\} \leftarrow \text{yes} \\ & \quad \underbrace{\hspace{1cm}} \\ & \quad \{ -1, +1 \} \leftarrow \text{yes} \\ & \quad \underline{[0, 1]} = \{ x : 0 \leq x \leq 1 \} \quad \text{yes.} \end{aligned}$$

① σ activation

② σ cost Φ

$\Rightarrow$ ③ gradient descent Φ
 $\rightarrow$ 어떻게 신경망?

Perceptron



The perceptron algorithm

Generalized linear model

$$y(\mathbf{x}) = f(\boldsymbol{\theta}^T \mathbf{x})$$

Nonlinear activation function given by a step function Sign

$$f(a) = \begin{cases} +1, & a \geq 0 \\ -1, & a < 0 \end{cases} \begin{matrix} C_1 \\ C_2 \end{matrix}$$


The diagram illustrates the step function $f(a)$ used in the perceptron algorithm. The horizontal axis represents the linear combination $\theta^T \mathbf{x}$, and the vertical axis represents the output $f(\theta^T \mathbf{x})$. The function is defined as $+1$ for $a \geq 0$ (labeled C_1) and -1 for $a < 0$ (labeled C_2). The output values $+1$ and -1 are marked on the vertical axis.

Target values $t = +1$ for class C_1 and $t = -1$ for class C_2 .

The perceptron algorithm

Perceptron criterion that we want to minimize

$$\min J(\theta) = - \sum_{\mathbf{x}_n \in \mathcal{M}} \theta^T \mathbf{x}_n t_n \geq 0$$

Misclassified pattern $\mathcal{M} = \{\mathbf{x}_n : \theta^T \mathbf{x}_n t_n < 0\}$

The stochastic gradient descent algorithm

$$\theta^{(\tau+1)} = \theta^{(\tau)} - \eta \nabla J(\theta) = \theta^{(\tau)} + \eta \mathbf{x}_n t_n$$

$$J(\theta) = - \sum_{\mathbf{x}_n \in \mathcal{M}} \theta^T \mathbf{x}_n t_n$$

$$\nabla_{\theta} J(\theta) \approx - \left(\sum_{\mathbf{x}_n \in \mathcal{M}} \mathbf{x}_n t_n \right)$$

$$J(\theta) = 0$$

↓

전부 다

지켜주

여러, 판별

→ 그림

$$\mathbf{x}_n \in \mathcal{M}$$

공식 2. 2. 2.

미러 θ	중단, target t
x_1	$t_1 = +1$
x_2	$t_2 = -1$
x_3	$t_3 = +1$
x_4	$t_4 = -1$
x_5	$t_5 = +1$

$\theta^T x$	$f(\theta^T x) = \text{여러}$
+5	$\rightarrow +1$
-2	$\rightarrow -1$
-3	$\rightarrow -1$
+4	$\rightarrow +1$
+2	$\rightarrow +1$

○
○
X
X
○

제대로 classify $\rightarrow t, \theta^T x$ 부호가 동일.

잘못 classify $\rightarrow t, \theta^T x$ 부호가 반대.

$$\{x_n : t_n \theta^T x_n < 0\} = \{x_3, x_4\}$$

입력 $\theta^T x$	주어진 target $\theta^T x, \text{target}$
x_1	$t_1 = +1$
x_2	$t_2 = -1$
x_3	$t_3 = +1$
x_4	$t_4 = -1$
x_5	$t_5 = +1$

$\theta^T x$	$f(\theta^T x) = \text{예측}$
+5	$\rightarrow +1$
-2	$\rightarrow -1$
+4	$\rightarrow +1$
+1	$\rightarrow +1$
+2	$\rightarrow +1$

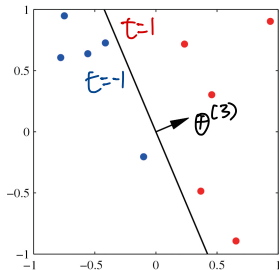
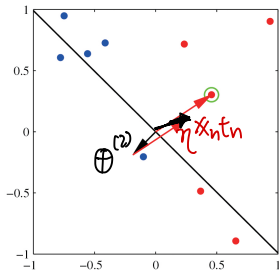
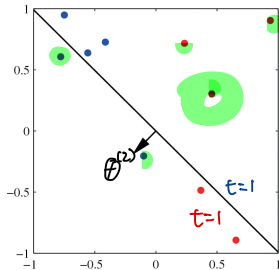
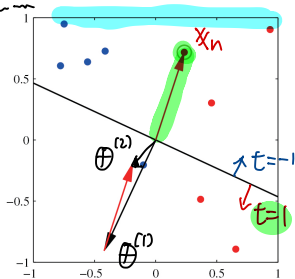
○
○
○
X
○

$$M = \{x_n: \theta^T x_n t_n < 0\} = \{x_4\}$$

$$J(\theta) = 1$$

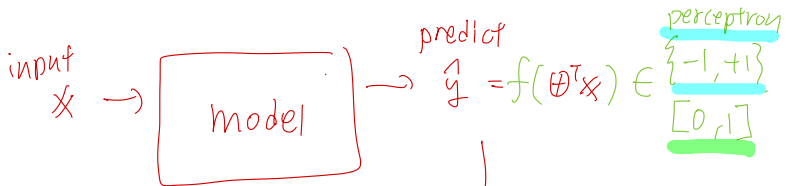
The perceptron algorithm

$$\underline{\theta} \leftarrow \underline{\theta} + \eta \underline{x}_n t_n$$



idea → 수식으로 풀이법.

물리각의미. ← 수식



output $t \rightarrow$

$$J(\theta) = - \sum_{x_n \in \underline{M}} \theta^T x_n t_n \quad \leftarrow \{x_n : \theta^T x_n t_n < 0\}$$

$$\nabla J(\theta) = - t_n x_n$$

logistic regression ↙

- activation function $\in [0, 1] = \{x; 0 \leq x \leq 1\}$
- probability
- cost: likelihood, $-\log$ likelihood ↘
- G.D.

Logistic regression

linear regression	vs.	logistic regression
perceptron	vs.	logistic regression

Logistic regression

$\Rightarrow$ classification

We begin our treatment of generalized linear models by considering the problem of two-class classification. Under rather general assumptions, the posterior probability of class C_1 can be written as a logistic sigmoid acting on a linear function of the feature vector $\mathbf{x}$ so that

$$C_1, C_2 \quad p(C_1 | \mathbf{x}) = y(\mathbf{x}) = \sigma(\theta^T \mathbf{x}) \quad (1)$$

with $p(C_2 | \mathbf{x}) = 1 - p(C_1 | \mathbf{x})$. Here, $\sigma(\cdot)$ is the logistic sigmoid function defined by

$$\sigma(a) = \frac{1}{1 + \exp(-a)} \quad (2)$$

In the terminology of statistics, this model is known as logistic regression, although it should be emphasized that this is a model for classification rather than regression.

	t_n
x_1	+1
x_2	<u>+1</u>
x_3	0
x_4	0
x_5	+1

$$\sigma(\theta^T x_n) = y_n$$

<u>0.9</u>	C_1
<u>0.4</u>	<u>C_2</u>
<u>0.2</u>	<u>C_2</u>
<u>0.6</u>	C_1
<u>0.8</u>	C_1

0	C_1
0	C_1
1	C_2
1	C_2
0	C_1

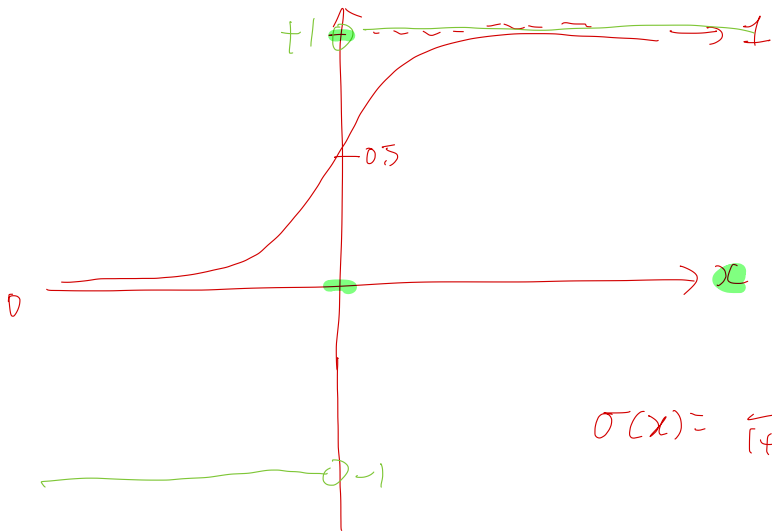
$C_1 \rightarrow +1$

$C_2 \rightarrow 0$

likelihood = 정답을 맞추는 것 ↑ 좋음.

$$0.9 \times 0.4 \times 0.8 \times 0.4 \times 0.8$$

$$0.9 \times 0.8 \times 0.9 \times 0.8 \times 0.9$$



$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Likelihood probability

For a data set $\{\mathbf{x}_n, t_n\}$, where $t_n \in \{0, 1\}$ with $n = 1, \dots, N$, the likelihood function can be written

$$\mathcal{L}(\theta) = p(\mathbf{t} | \theta) = \prod_{n=1}^N y_n^{t_n} \{1 - y_n\}^{1-t_n} \quad (3)$$

where $\mathbf{t} = (t_1, \dots, t_N)^T$ and $y_n = p(C_1 | \phi_n)$.

likelihood: $\exists$ θ $\frac{1}{N} \sum_{n=1}^N \log y_n$ $\frac{1}{N} \sum_{n=1}^N \log (1 - y_n)$ $\uparrow$ 

- log likelihood: neg. log likelihood, $\downarrow \frac{1}{N} \sum_{n=1}^N \log y_n$

$$L(\theta) = \prod$$

$$t_n = +1$$

$$t_n = 0$$

$$\frac{y_n^{t_n}}{1 - y_n^{1-t_n}} \Leftrightarrow$$

$$y_n = \sigma(\theta^T x_n)$$

$$1 - y_n = 1 - \sigma(\theta^T x_n)$$

Cross entropy and gradient

As usual, we can define an error function by taking the negative logarithm of the likelihood, which gives the

✧ cross-entropy error function in the form

$$J(\boldsymbol{\theta}) = -\log p(\mathbf{t} | \boldsymbol{\theta}) = -\sum_{n=1}^N \{t_n \log y_n + (1 - t_n) \log(1 - y_n)\} \quad (4)$$

where $y_n = \sigma(a_n)$ and $a_n = \boldsymbol{\theta}^T \boldsymbol{\phi}_n$. Taking the gradient of the error function with respect to $\boldsymbol{\theta}$, we obtain

$$\nabla J(\boldsymbol{\theta}) = \sum_{n=1}^N (y_n - t_n) \mathbf{x}_n. \quad (5)$$

Perceptron

$$y(x) = \text{Sign}(\theta^T x) \\ \in \{-1, +1\}$$

Model:

Cost: $M = \{x_n; \theta^T x_n t_n < 0\}$

$$J(\theta) = - \sum_n \theta^T x_n t_n \geq 0$$

GD: $\theta \leftarrow \theta + \eta x_n t_n$

Logistic

$$y(x) = \sigma(\theta^T x) \\ \in [0, 1] \\ = P(C_1 | x)$$

$$\mathcal{L}(\theta) = \prod y_n^{t_n} (1 - y_n)^{1 - t_n}$$

$$J(\theta) = -\log \mathcal{L}(\theta)$$

$$= -\sum t_n \log y_n + (1 - t_n) \log (1 - y_n)$$

$$\theta \leftarrow \theta - \eta (y_n - t_n) x_n$$

linear regression.

$$y = \theta^T x$$

$$J(\theta) = \|X\theta - y\|$$

$$\nabla J(\theta)$$

$$\theta \leftarrow \theta - 2\eta (\hat{y}_n - y_n) x_n$$

- linear regression + SSE $\Rightarrow$ G.D.
- logistic regression + $-\log \mathcal{L}(\theta)$ $\Rightarrow$ G.D.

Gradient descent

The contribution to the gradient from data point n is given by the error $y_n - t_n$ between the target value and the prediction of the model, times the basis function vector $\mathbf{x}_n$. This takes precisely the same form as the gradient of the sum-of-squares error function for the linear regression model.

Reference and further reading

- “Chap 4” of C. Bishop, Pattern Recognition and Machine Learning
- “Chap 3” of A. Geron, Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow