Machine learning & deep learning basics

Lecture 1

Changho Suh

January 22, 2024

1. Logistics
2. Machine learning & optimization

Logistics

Instructor

Changho Suh
N1-912, 042-350-7429
chsuh@kaist.ac.kr
http://csuh.kaist.ac.kr

2 week course

Week 1: Lecture & Practice session

Week 2: Mini-projects Proposal

Week 1: Lecture & practice session

1.1: Machine learning & deep learning basics

1.2: Advanced techniques

1.3: Convolutional Neural Network (CNN)

1.4: Recurrent Neural Network (RNN)

1.5: Small data technique: Random Forests

Week 2: Mini-projects & proposal

Group A (권~박)

수강생	부서
권태운	전동화시스템시험3팀
김동현	모빌리티컨셉개발팀
김미진	차량제어성능개발팀
김수환	제네시스외장설계팀
김외태	전산재료과학연구팀
김종훈	MSV내구시험팀
김준영	MLV전동화연비시험팀
김진현	제네시스외장설계팀
박정수	자율주행전략기술개발팀
박한결	전동화시스템시험3팀

Group B (박~최)

박형호	제네시스샤시설계1팀
신용욱	연구개발품질확보팀
이은주	인포테인먼트기획팀
이창주	전동화시스템시험1팀
임경빈	전자전력제어개발팀
임재영	버추얼이노베이션리서치랩
조대길	자율주행시스템개발팀
조재설	상용제동설계팀
주장규	인포테인먼트기획팀
최정윤	차량에너지제어개발팀

Week 2: Mini-projects & proposal

2.1: Overview of two mini-projects
Mini-project #1

2.2: Mini-project #2
Proposal guideline & sample proposals

2.3: (Group A) Rehearsal & feedback

2.4: (Group B) Rehearsal & feedback

2.5: Proposal presentation

Week 1 schedule

1.1: Machine learning and deep learning basics

Lecture 1: 9:00 am ~ 10:00 am

Lecture 2: 10:10 am ~ 11:10 am

Lecture 3: 11:20 am ~ 12:30 pm

PS 1: 1:30 pm ~ 2:30 pm

PS 2: 2:40 pm ~ 3:40 pm

PS 3: 3:50 pm ~ 5:00 pm

Same format for 1.2 ~ 1.5

15 lectures & 15 PSs

Week 2: Day 1 schedule

2.1: Mini-project overview & mini-project #1

Lecture 16: 9:00 am ~ 10:00 am

Lecture 17: 10:10 am ~ 11:10 am

PS 16: 11:20 am ~ 12:30 pm

PS 17: 1:30 pm ~ 2:30 pm

PS 18: 2:40 pm ~ 3:40 pm

PS 19: 3:50 pm ~ 5:00 pm

Week 2: Day 2 schedule

2.2: Mini-project #2 & proposal guideline

PS 20: 9:00 am ~ 10:00 am

PS 21: 10:10 am ~ 11:10 am

PS 22: 11:20 am ~ 12:30 pm

Lecture 18: 1:30 pm ~ 2:30 pm

Lecture 19: 2:40 pm ~ 3:40 pm

Lecture 20: 3:50 pm ~ 5:00 pm

Week 2: Day 3 schedule

2.3: (Group A) Rehearsal & feedback

09:00 ~ 09:20	**권태운**
09:20 ~ 09:40	**김동현**
09:40 ~ 10:00	**김미진**
10:00 ~ 10:20	**김수환**
10:20 ~ 10:40	**김외태**
Break	
10:50 ~ 11:10	**김종훈**
11:10 ~ 11:30	**김준영**
11:30 ~ 11:50	**김진현**
11:50 ~ 12:10	**박정수**
12:10 ~ 12:30	**박한결**
Lunch	

13:30 ~ 16:00	**Proposal 수정**
16:00 ~ 17:00	**Q&A**

Week 2: Day 4 schedule

2.4: (Group B) Rehearsal & feedback

09:00 ~ 09:20	박형호
09:20 ~ 09:40	신용욱
09:40 ~ 10:00	이은주
10:00 ~ 10:20	이창주
10:20 ~ 10:40	임경빈
Break	
10:50 ~ 11:10	임재영
11:10 ~ 11:30	조대길
11:30 ~ 11:50	조재설
11:50 ~ 12:10	주장규
12:10 ~ 12:30	최정윤
Lunch	

13:30 ~ 16:00	Proposal 수정
16:00 ~ 17:00	Q&A

Week 2: Day 5 schedule

2.5: Proposal presentation

09:00 ~ 09:05	opening
09:05 ~ 09:20	권태운
09:20 ~ 09:35	김동현
09:35 ~ 09:50	김미진
09:50 ~ 10:05	김수환
10:05 ~ 10:20	김외태
10:20 ~ 10:35	김종훈
Break	
10:45 ~ 11:00	김준영
11:00 ~ 11:15	김진현
11:15 ~ 11:30	박정수
11:30 ~ 11:45	박한결
11:45 ~ 12:00	박형호
Lunch	

13:30 ~ 13:45	신용욱
13:45 ~ 14:00	이은주
14:00 ~ 14:15	이창주
14:15 ~ 14:30	임경빈
14:30 ~ 14:45	임재영
Break	
14:45 ~ 15:00	조대길
15:10 ~ 15:25	조재설
15:25 ~ 15:40	주장규
15:40 ~ 15:55	최정윤
Closing	

Reference

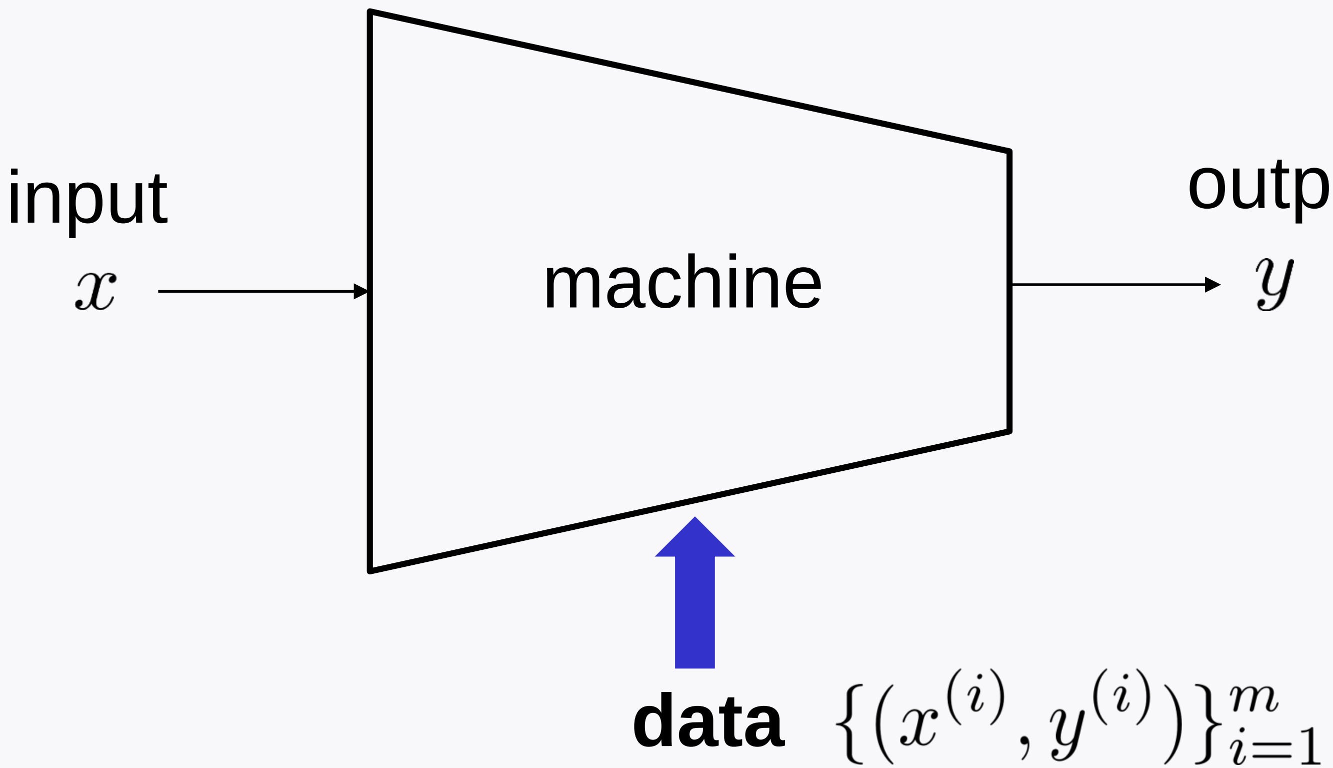
1. Lecture Slides (LS)

2. Practice Session (PS):

   Slides & python code

Machine learning and optimization

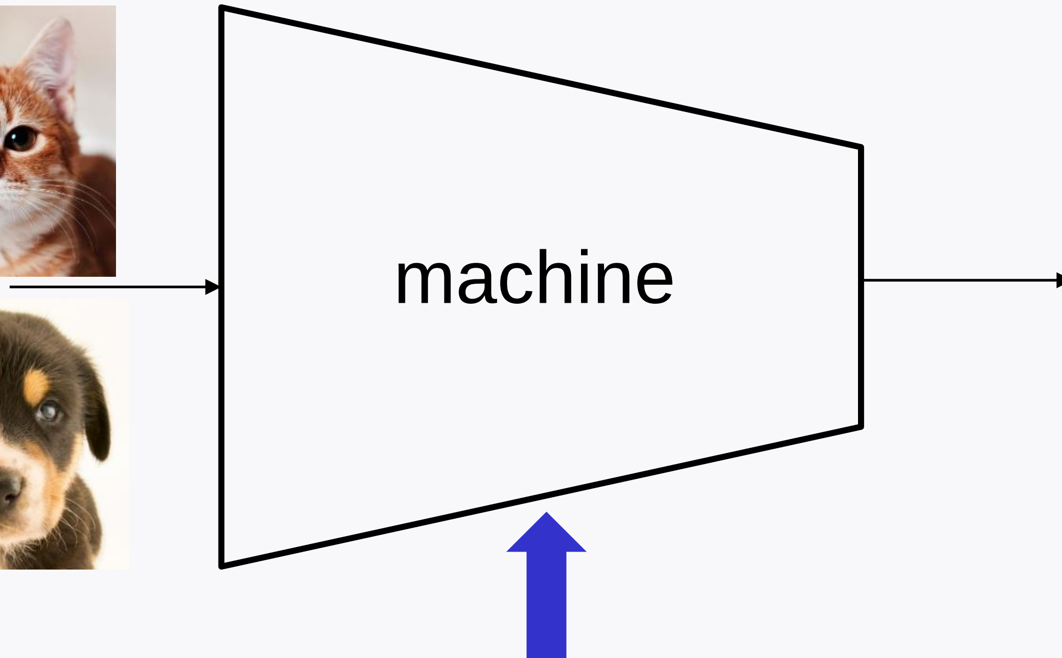
Machine learning



Cat-vs-dog classifier

$x^{(1)} =$ 

$x^{(2)} =$ 



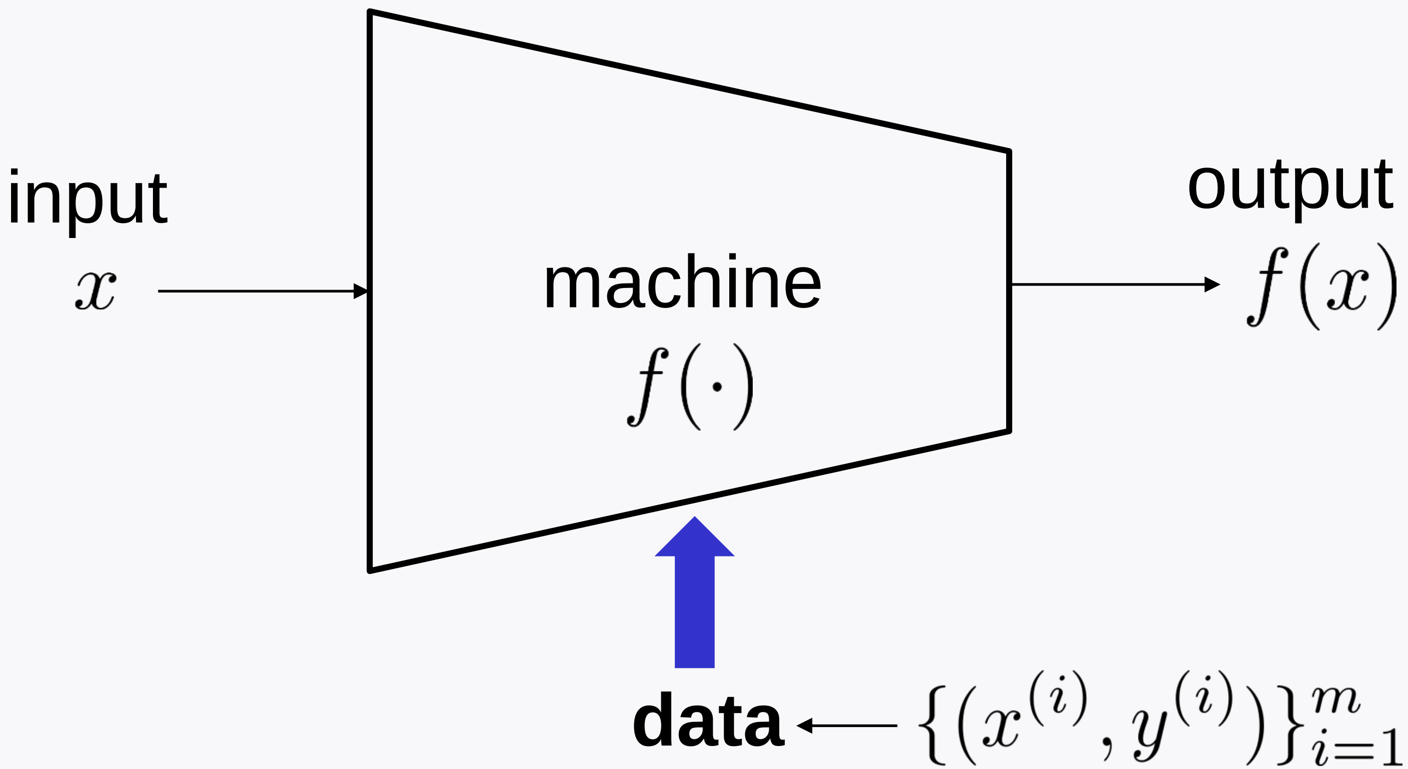
$y^{(1)} = 1$

$y^{(2)} = 0$

data $\{(x^{(i)}, y^{(i)})\}_{i=1}^{m}$

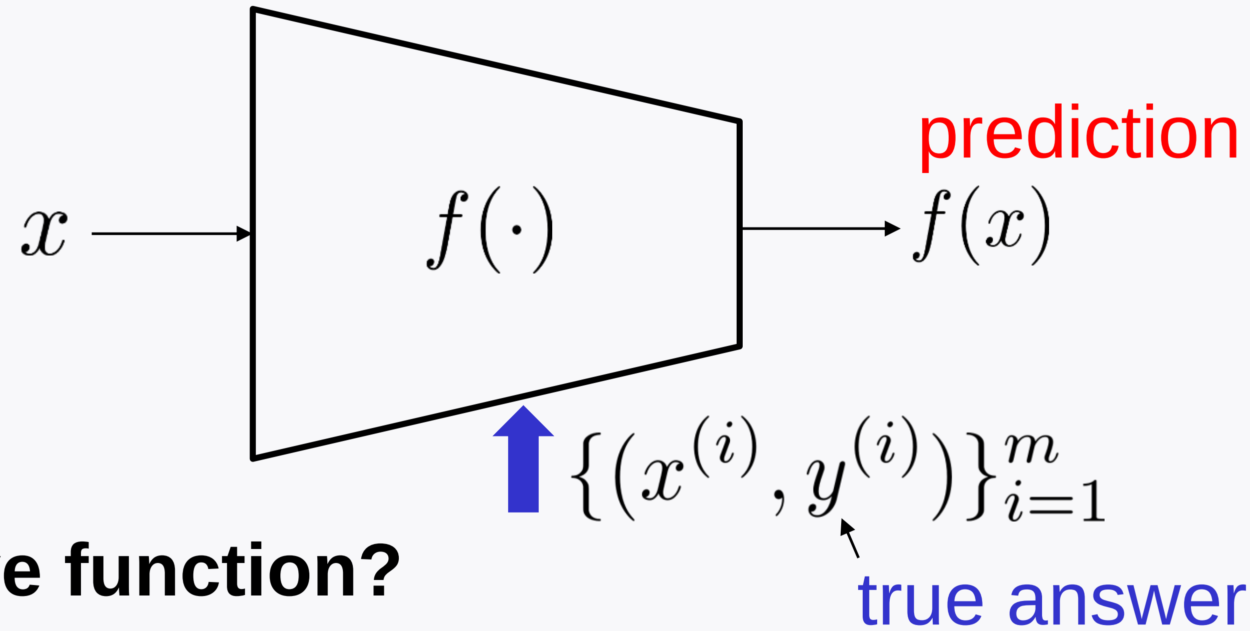
big data → large *m*

Goal of machine learning



Design $f(\cdot)$ using $\{(x^{(i)}, y^{(i)})\}_{i=1}^{m}$

Training via optimization



Objective function?

What we want: $f(x^{(i)}) \approx y^{(i)}$ for all i

prediction   true answer

How to **quantify closeness**?

One way is to employ a loss function: $\ell(y^{(i)}, f(x^{(i)}))$

Optimization variable?

$$\min_{f} \sum_{i=1}^{m} \ell(y^{(i)}, f(x^{(i)}))$$

Note: Function optimization!

Challenge: There are *so many* choices for function.

How to deal with function optimization?

$$\min_{f_w} \sum_{i=1}^{m} \ell(y^{(i)}, f_w(x^{(i)}))$$

A common way:

Specify a **function class** (e.g., linear, quadratic, …)

Represent $f(\cdot)$ with parameters w

Consider the parameters as optimization variable.

Parameterization

$$\min_{w} \sum_{i=1}^{m} \ell(y^{(i)}, f_{w}(x^{(i)}))$$

Depending on a choice of function class & loss function, there are **three** prominent problems:

1. Least Squares

2. Logistic regression

3. Deep learning

A choice for $f_w(\cdot)$

$$\min_{w} \sum_{i=1}^{m} \ell(y^{(i)}, f_w(x^{(i)}))$$

One architecture was suggested:

Perceptron

Consists of two operations:

1. Inner product: $w^T x$

2. Activation: $f_w(x) = \begin{cases} 1 & \text{if } w^T x > \text{th} \\ 0 & \text{otherwise} \end{cases}$

(inspired by neuron's behavior)

Frank Rosenblatt '57
(psychologist)

Least Squares

$$\min_{w} \sum_{i=1}^{m} \ell(y^{(i)}, w^T x^{(i)})$$

Employ: Perceptron w/o activation

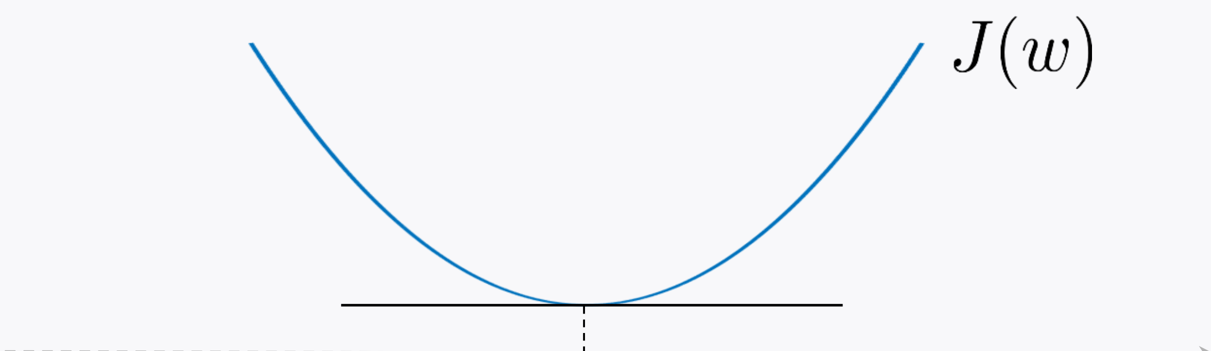
Least Squares

$$\min_{w} \sum_{i=1}^{m} \left\| y^{(i)} - w^T x^{(i)} \right\|^2 =: J(w)$$

Employ: Perceptron w/o activation

A squared error loss: $\ell(y, \hat{y}) = \|y - \hat{y}\|^2$

Convex optimization



w^* closed form solution!

Least Squares

$$\min_{w} \sum_{i=1}^{m} \left\| y^{(i)} - w^T x^{(i)} \right\|^2 =: J(w)$$

Employ: Perceptron w/o activation

A squared error loss: $\ell(y, \hat{y}) = \|y - \hat{y}\|^2$

Convex optimization

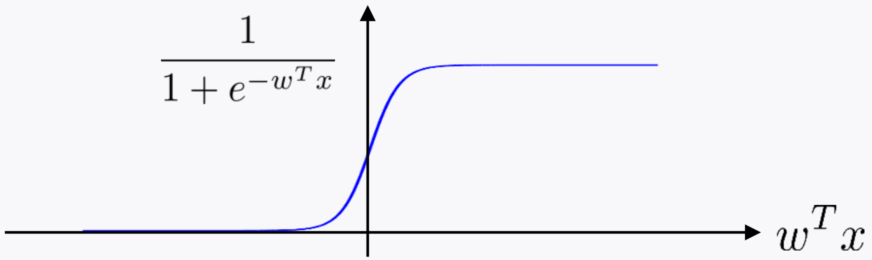
Has the closed form solution.

But performance is not that great.

Logistic regression

$$\min_{w} \sum_{i=1}^{m} \ell\left(y^{(i)}, \frac{1}{1+e^{-w^T x^{(i)}}}\right)$$

Employ: Perceptron w/ logistic function



Cross Entropy (CE) loss:

$$\ell(y, \hat{y}) = -y \log \hat{y} - (1-y) \log(1-\hat{y})$$

Outperforms least squares.

Logistic regression

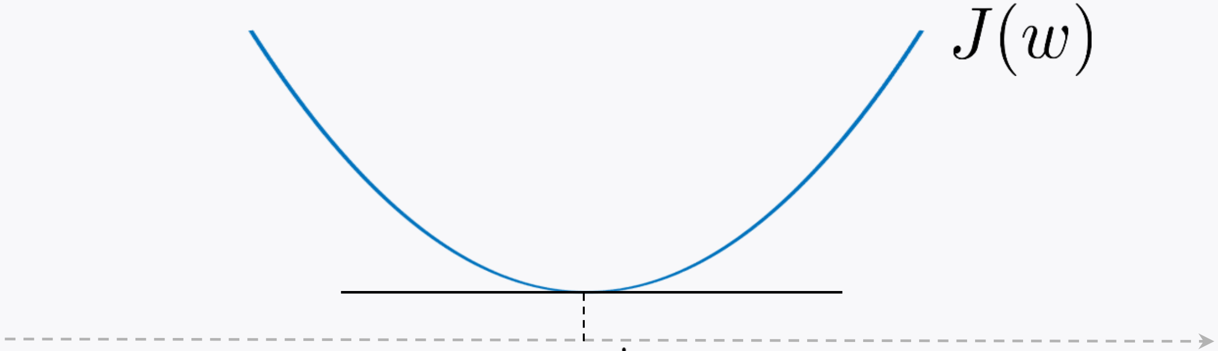
$$\min_{w} \boxed{\sum_{i=1}^{m} -y^{(i)} \log \frac{1}{1+e^{-w^T x^{(i)}}} - (1-y^{(i)}) \log \left(1 - \frac{1}{1+e^{-w^T x^{(i)}}}\right)}$$

$$=: J(w)$$

Employ: Perceptron w/ logistic function

Cross Entropy (CE) loss

Convex optimization



But no closed form solution

How to train logical regression?

No closed form solution.

Good news: There exist algorithms that allow us to find the solution numerically.

One prominent algorithm:

Gradient descent!

Look ahead

Will study: Gradient descent.

Then move onto deep learning.