모두의연구소

딥러닝 기초

모두의연구소
박은수 Research Director

3일 동안의 진행계획

모두의연구소

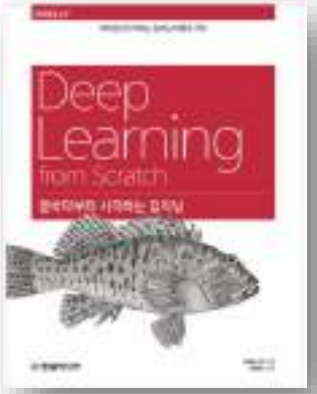
- 1일
  - Numpy tutorial
  - Loss Function and Optimization
- 2일
  - Introduction to Neural Networks
  - Training Neural Networks - 1
  - Training Neural Networks - 2
- 3 일
  - Convolutional Neural Networks
  - Recurrent Neural Networks, LSTM

강의를 만들때 참고한것

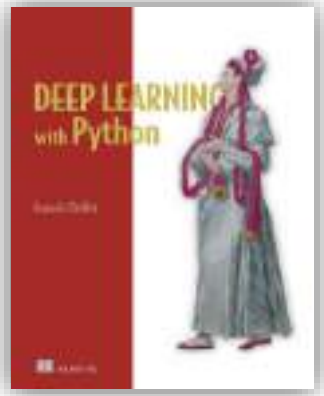
모두의연구소

- CS231n : Convolutional Neural Networks for Visual Recognition
  - http://cs231n.stanford.edu/syllabus.html

- Deep Learning from the Scratch (밑바닥 부터 시작하는 딥러닝)



- Deep Learning with Python



당신은 뭐하는 사람인가요

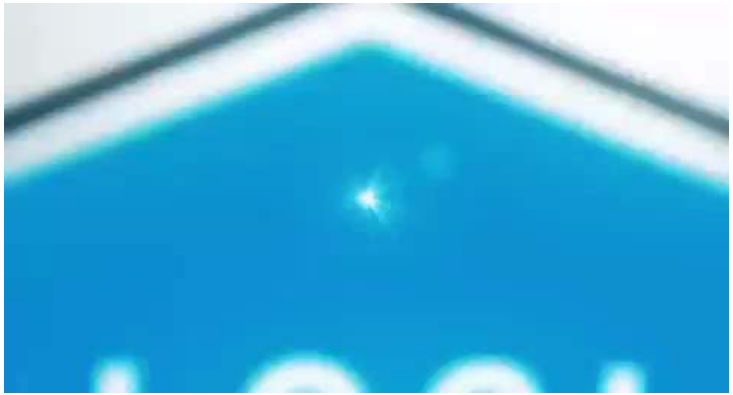
모두의연구소

Lecturer (Inha Univ.)
Digital Image Processing (Programming)
Digital Signal Processing (Programming)

~ 2012.11

Leader (Inha Autonomous Vehicle Team)

- Coordinating Computer vision and mechanical engineering Lab.
- Introducing ROS to simplify the sensor synchronization issue



- Results
  - Rank 7th , 1 patent, 3 conference papers
  - Interview Video

당신은 뭐하는 사람인가요

모두의연구소

2012.11
~
2014.01

Team reader (Inha Autonomous Vehicle Team)

CEO (VisionIn)

Setting up Office

Enhancing video Quality

Display Power Checker

[Intelligent Photo Album : NooGoo]

Motivation

- Contribution to Society
- Making better World using our technology

당신은 뭐하는 사람인가요

모두의연구소

CEO (VisionIn)

2014.03~2014.07 **Lecturer (Inha Technical College) : Software application**

…… 방황 ? ㅋㅋ …..

복학

2016.03 ~2016.06 **Lecturer (Inha Univ.) : CNN for Visual Recognition**

2016.08 **Lecturer (Inha Univ.) : Understanding CNN (one week course)**

2017.01 **Lecturer (Inha Univ.) :**

- **Python Basics**
- **Understanding CNN (one week course)**

2017.06 **Graduate**

저와 연구실 동료들이 했던 연구

모두의연구소

Graduate

2017.07 **ML Jeju Camp**

모두연에
나오시는 분들

2017.08 **모두연 합류**

2017.11 **모두의연구소에서 프로젝트 및 딥러닝 컬리지 운영**

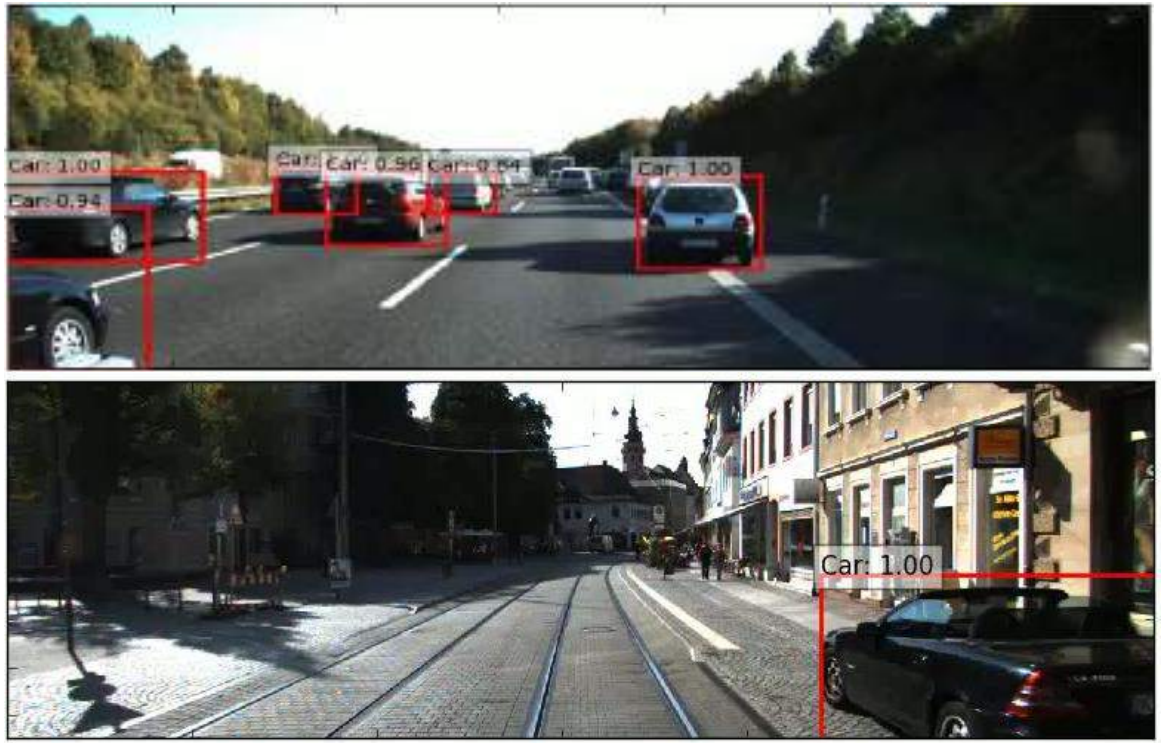
⋮

2018.09 **여러분과의 만남**

저와 연구실 동료들이 했던 연구

모두의연구소

차량에서의
물체인식



2016년

저와 연구실 동료들이 했던 연구

모두의연구소

위조지문 검출

진짜 지문은
무엇일까요?

2016년

저와 연구실 동료들이 했던 연구

모두의연구소

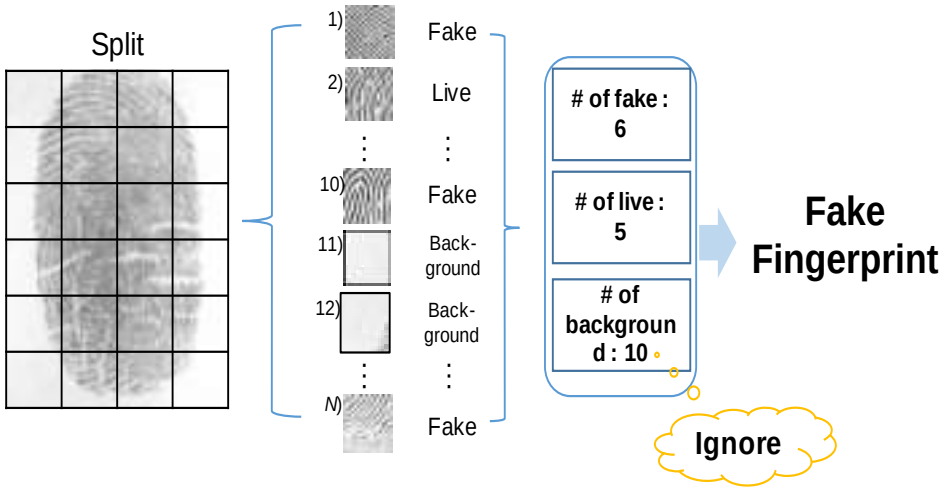
위조지문 검출

진짜 지문은
무엇일까요?

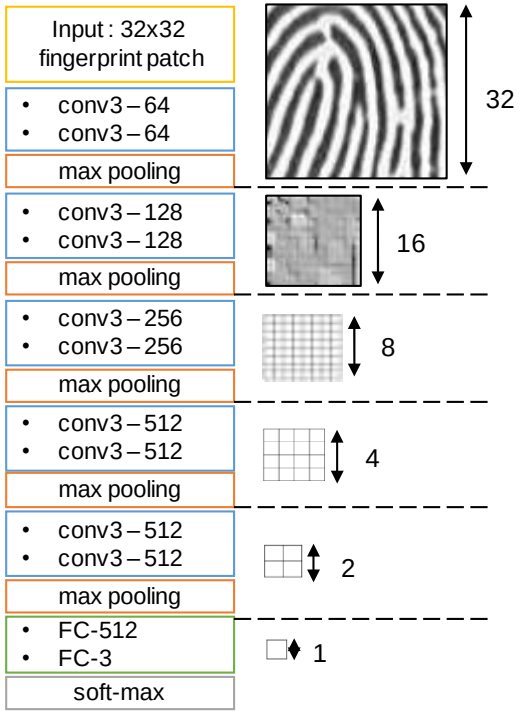
2016년

저와 연구실 동료들이 했던 연구

모두의연구소



2016년



모두연 분들과 함께 해본

LivDet 2017

모두의연구소

모두연 분들과도 해봤어요~

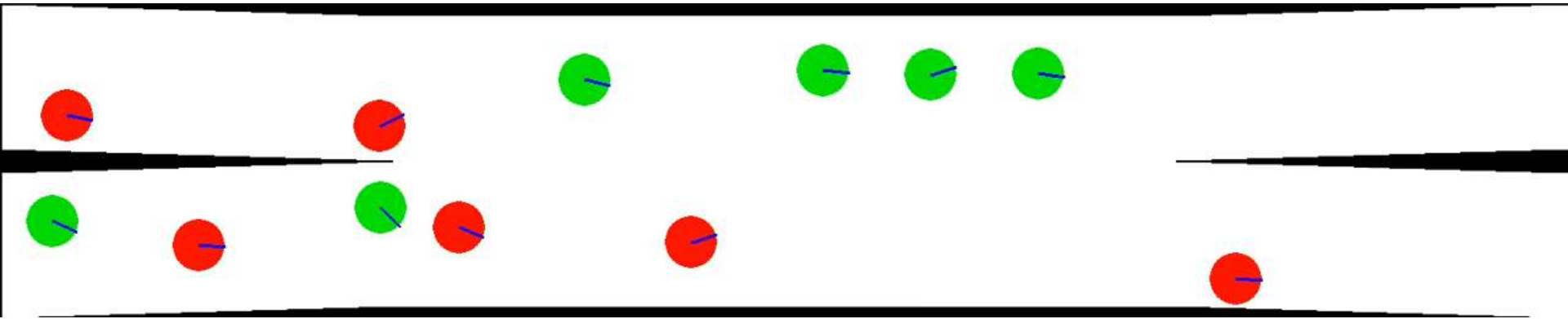
모두연 분들과 함께 해본

모두의연구소

Deep Deterministic Policy Gradient

부들부들 ~~~

2017



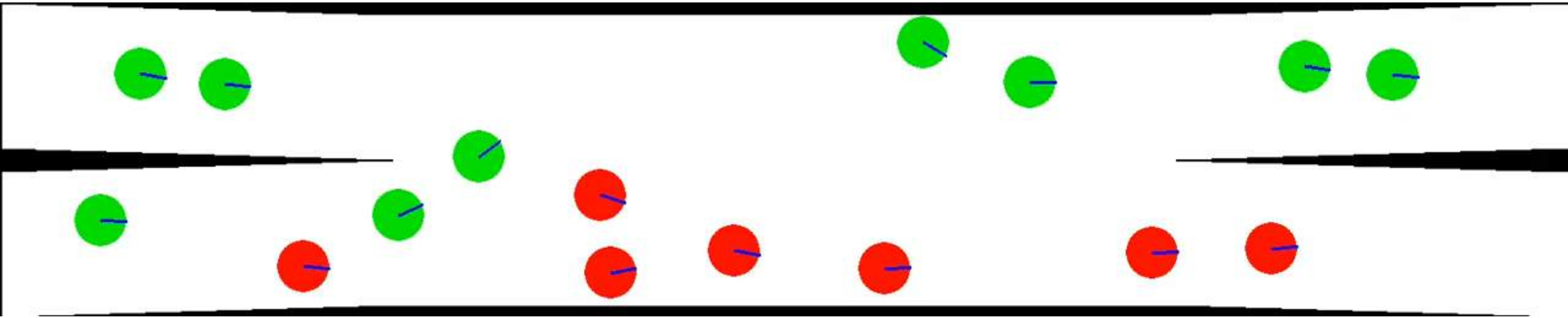
모두연 분들과 함께 해본

모두의연구소

Deep Deterministic Policy Gradient

슈우욱 ~~~

2017



모두의연구소

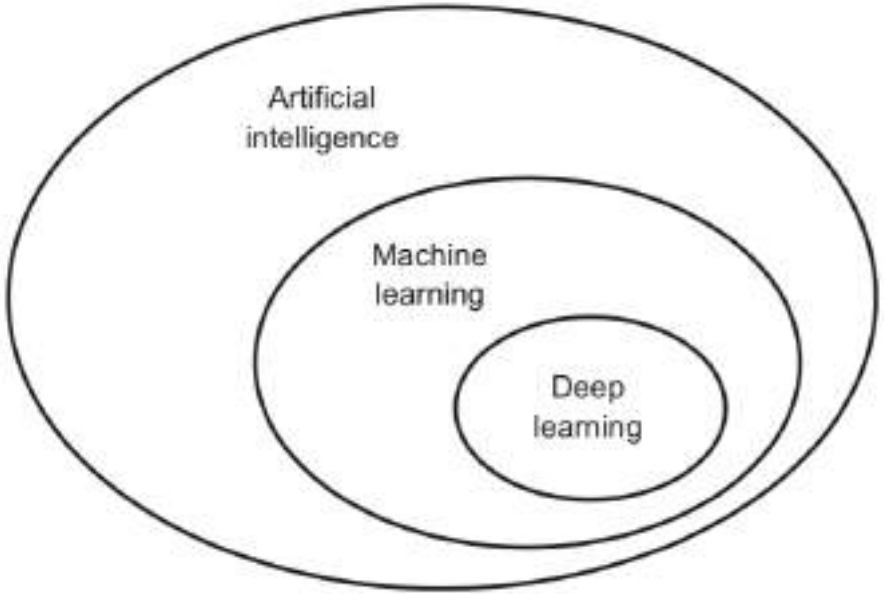
딥러닝 피플이 되신 걸 환영합니다

모두의연구소

딥러닝의 30가지 적용사례

https://brunch.co.kr/@itschloe1/23

Artificial Intelligence, Machine Learning, Deep Learning



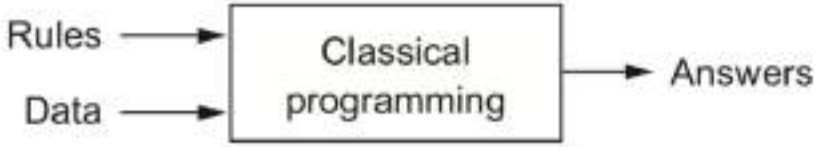
- Artificial Intelligence
  - 인간이 수행할 수 있는 지능적 업무를 자동화하려는 노력에서 기인
  - 가장 넓은 개념
- Symbolic AI : 룰을 기반으로 작동되는 AI (1950s ~ 1980s)
  - 1980년대 expert system(전문가 시스템)
  - 인간 성능의 AI가 가능할 것이라 믿음

From: Deep learning with Python, Francois Chollet

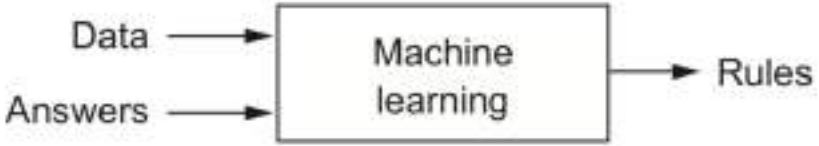
Artificial Intelligence, Machine Learning, Deep Learning

모두의연구소

- Machine Learning



Symbolic AI



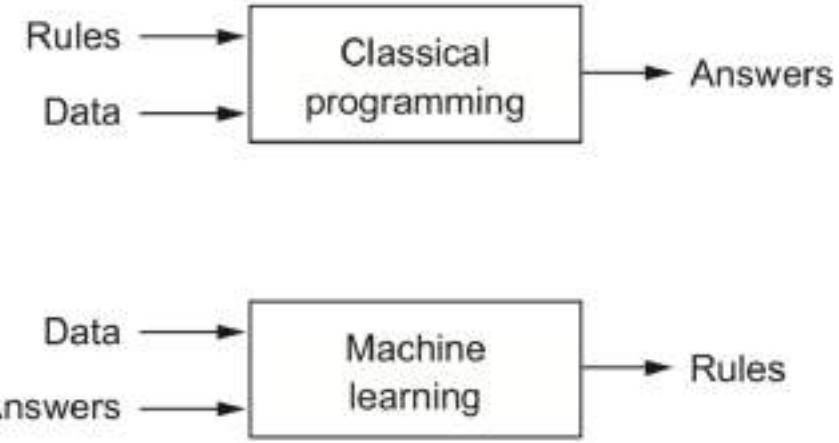
Machine Learning

From: Deep learning with Python, Francois Chollet

Artificial Intelligence, Machine Learning, Deep Learning

모두의연구소

- Machine Learning



- 1990년대에 전성기
- Machine Learning은 룰을 이용하여 프로그래밍 된다기 보다는 학습된다

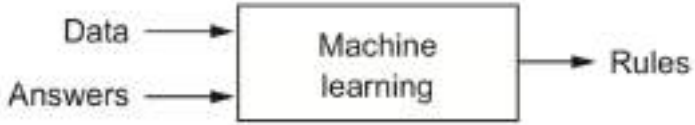
예시) 많은 수의 사진(Data)과 그에 해당하는 태그(Answers)들이 있다면 머신러닝 시스템은 사진과 그에 해당하는 태그를 연결하는 통계적 룰을 학습 할 수 있다

From: Deep learning with Python, Francois Chollet

Artificial Intelligence, Machine Learning, Deep Learning

모두의연구소

- Machine Learning



Statistics vs. Machine Learning

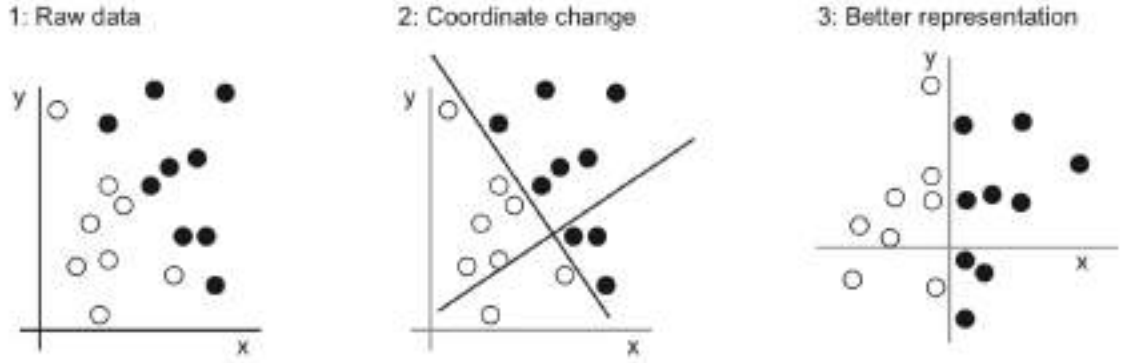
- 통계학과 달리 머신러닝은 베이지안으로 분석하기 어려운 수백개의 이미지와 픽셀과 같은 복잡하고 큰 데이터를 주로 다루게 된다. 결과적으로 머신러닝, 특히 딥 러닝은 비교적 통계학에서 다루는 방법보다 덜 수학적이고 공학적인 성격이 강하다. 따라서 이론적이기 보다는 실험적인 부분이 많이 존재한다

From: Deep learning with Python, Francois Chollet

Artificial Intelligence, Machine Learning, Deep Learning

모두의연구소

- Learning **representations** from data
  - 데이터와 정답을 관계짓는 룰을 잘 학습하기 위해서 데이터는 변형된다
  - 즉 정답과 잘 관계짓기 위한 입력되는 데이터의 representation을 학습하여야 한다
    - Ex) 좌표변환, 이동, 비선형 연산 (x>0) 등

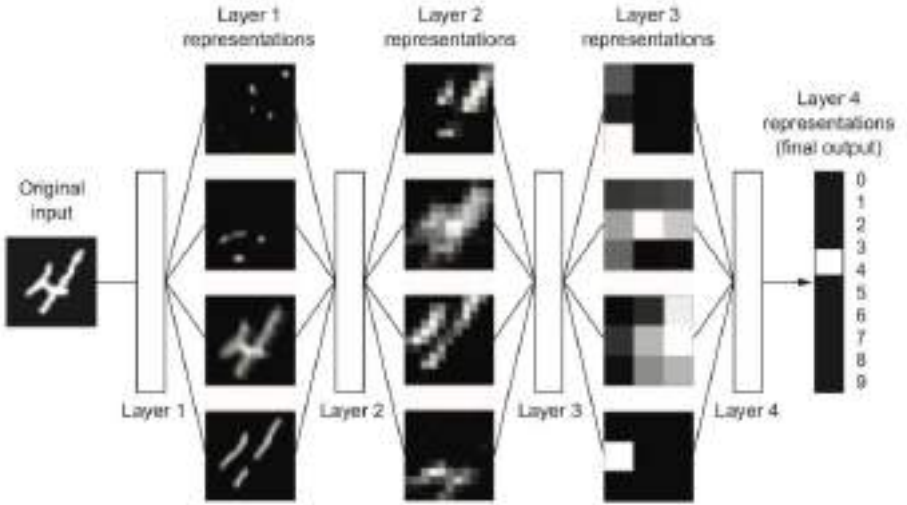


From: Deep learning with Python, Francois Chollet

Artificial Intelligence, Machine Learning, Deep Learning

모두의연구소

- 딥러닝은 뉴럴네트워크를 이용하는 계층적 representation learning 이다

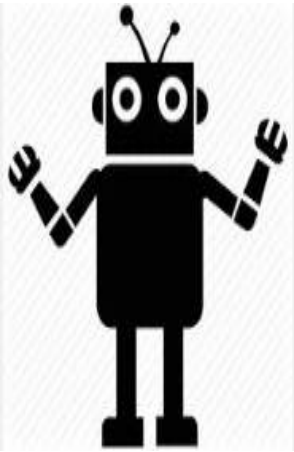


From: Deep learning with Python, Francois Chollet

물체인식에서의 딥러닝

모두의연구소

인간 vs. 기계



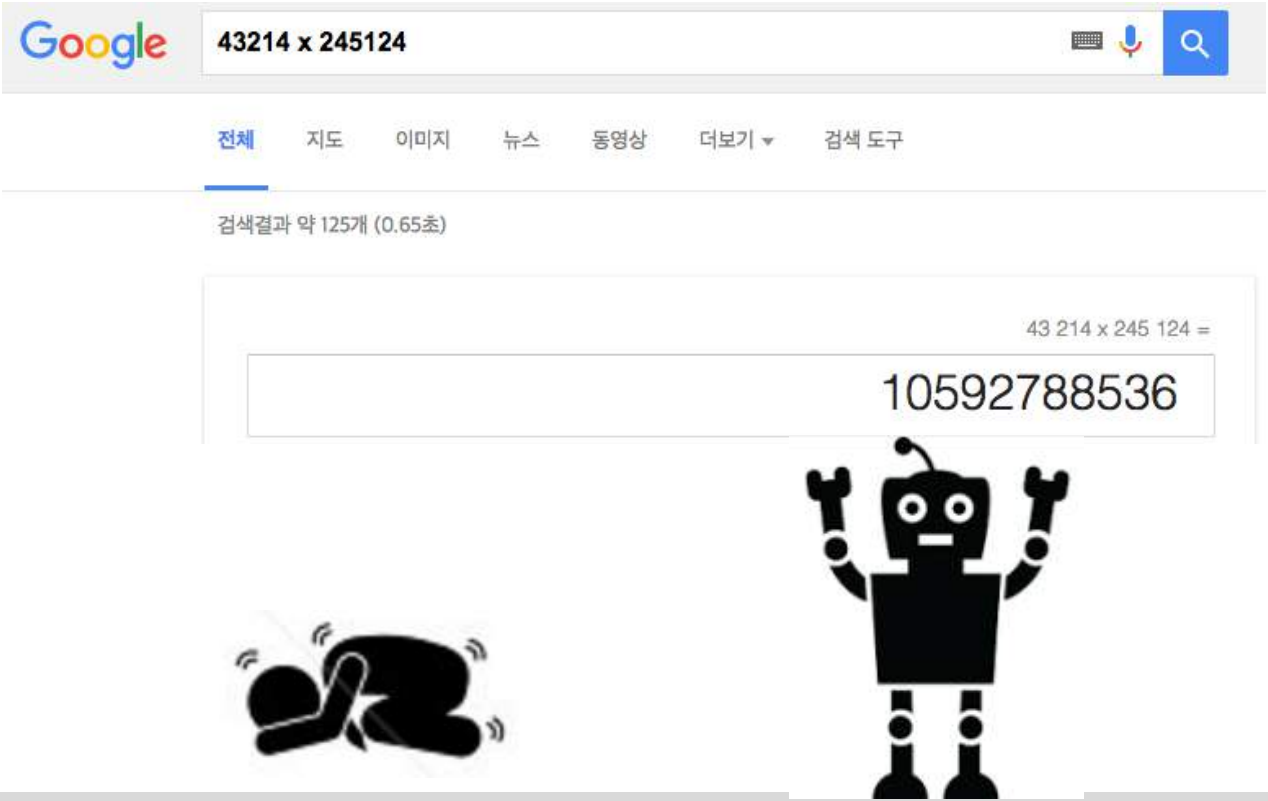
물체인식에서의 딥러닝

모두의연구소

43214 x 245124 = ?

물체인식에서의 딥러닝

모두의연구소



물체인식에서의 딥러닝

모두의연구소

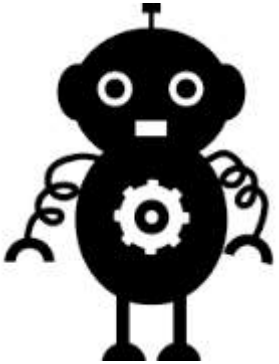
고양이랑 강아지 구분하기

물체인식에서의 딥러닝

모두의연구소

왼쪽 고양이
오른쪽 강아지 ~ !!

아...



물체인식에서의 딥러닝

모두의연구소

컴퓨터 비젼에서의 물체인식

{고양이, 강아지, 버스, 자동차, ..., 집}

→ 고양이

* From : Lecture slide 2 of "CS231n : Convolutional Neural Networks for Visual Recognition"

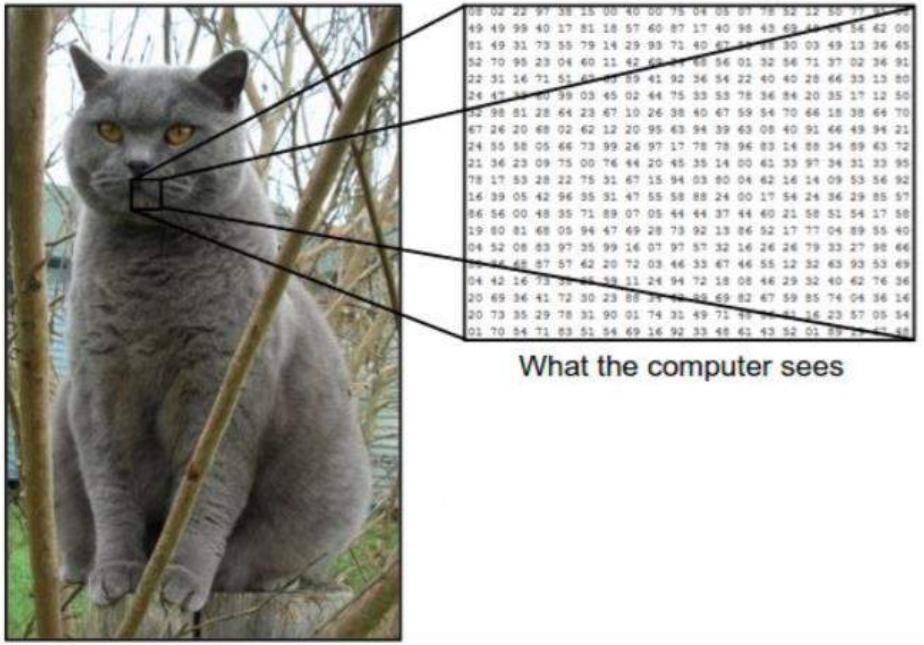
물체인식에서의 딥러닝

모두의연구소

컴퓨터 비젼에서의 물체인식

영상은 [0, 255] 범위의 값을 갖는 RGB 3채널로 구성되어 있다

옆의 고양이는 300x100x3 의 배열 (3은 RGB (Red, Green, Blue) 채널)



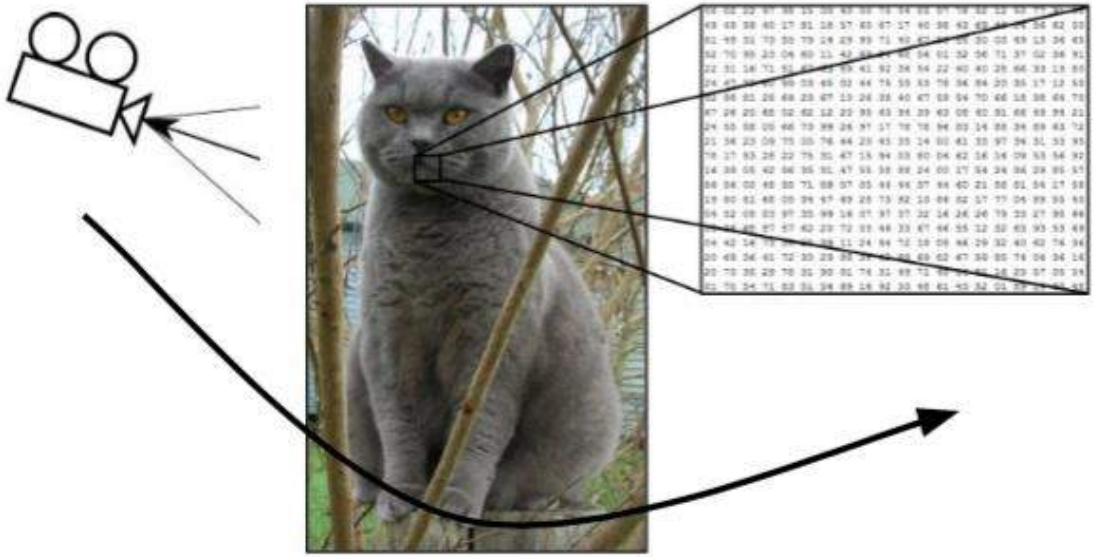
* From : Lecture slide 2 of "CS231n : Convolutional Neural Networks for Visual Recognition"

물체인식에서의 딥러닝

모두의연구소

컴퓨터 비젼에서의 물체인식의 어려움

카메라 위치에 따른
변화



* From : Lecture slide 2 of "CS231n : Convolutional Neural Networks for Visual Recognition"

물체인식에서의 딥러닝

모두의연구소

컴퓨터 비젼에서의 물체 인식의 어려움

조명에 따른 변화

* From : Lecture slide 2 of "CS231n : Convolutional Neural Networks for Visual Recognition"

물체인식에서의 딥러닝

모두의연구소

컴퓨터 비젼에서의 물체인식의 어려움

객체 형태의 변화

* From : Lecture slide 2 of "CS231n : Convolutional Neural Networks for Visual Recognition"

물체인식에서의 딥러닝

모두의연구소

컴퓨터 비젼에서의 물체인식의 어려움

가려짐

* From : Lecture slide 2 of "CS231n : Convolutional Neural Networks for Visual Recognition"

물체인식에서의 딥러닝

모두의연구소

컴퓨터 비젼에서의 물체인식의 어려움

배경과의 유사성

* From : Lecture slide 2 of "CS231n : Convolutional Neural Networks for Visual Recognition"

물체인식에서의 딥러닝

모두의연구소

컴퓨터 비젼에서의 물체인식의 어려움

같으면서 다른 형태의 물체

* From : Lecture slide 2 of "CS231n : Convolutional Neural Networks for Visual Recognition"

물체인식에서의 딥러닝

모두의연구소

컴퓨터 비젼에서의 물체인식의 어려움

```
def predict(image):
    # ????
    return class_label
```

고양이와 개를 구분하는 알고리즘을 만들어 보세요

* From : Lecture slide 2 of "CS231n : Convolutional Neural Networks for Visual Recognition"

물체인식에서의 딥러닝

모두의연구소

컴퓨터 비젼에서의 물체인식의 어려움

죄송합니다

어려운걸 알고있습니다

If else ~~ if else if else if else, ... ??

switch case 1 : case 2 case : ..... ??

Data-driven 방법

모두의연구소

1. 이미지와 그에 해당하는 레이블(label)을 모음
2. 머신러닝 방법론으로 분류기를 학습
3. 학습된 분류기를 테스트 이미지에 평가

우리는 Data Driven 방법을
취할 겁니다

```
def train(train_images, train_labels):
  # build a model for images -> labels...
  return model

def predict(model, test_images):
  # predict test_labels using the model...
  return test_labels
```

* From : Lecture slide 2 of "CS231n : Convolutional Neural Networks for Visual Recognition"

모두의연구소

뉴럴네트워크 구현 시 기본적으로 알아야 할 Python package

Numpy

미련한 짓

모두의연구소

- 우린 Backpropagation을 하나하나 다 구현해 볼 생각이에요
  - 이걸 왜 구현해요? Tensorflow 가 해주는데 어렵게 구현해 봐야 하나요?
  - 답 : Yes you should understand backprop

모두의연구소

Numpy에 익숙해 집시다

가장 기본이 됩니다

```
In [1]: import numpy as np
```

Numpy Tutorial :
http://cs231n.stanford.edu/notebooks/python_numpy_tutorial.ipynb

Broad casting Rule

모두의연구소

- Rank 가 다르면 부족한 rank를 보유한 행렬 앞에 1을 추가하여 shape의 길이가 같게 만듦
- Compatible의 의미 :
    - 두 행렬의 차원이 같을 때 혹은
    - 두 행렬 중 다른 차원의 값이 1을 포함 할 때
- Compatible 하면 broad casting이 가능함
- Compatible 중 1을 가지고 있는 더 작은 차원은 더 큰 행렬의 차원으로 자기자신을 복제 함
- 출력은 :
    - 두 입력의 shape의 요소별 max와 같음

Broad casting Rule

모두의연구소

```
x = np.array([[1, 2, 3],
              [3, 5, 7]])
print(x.shape)  # Prints (2, 3)

col_means = x.mean(axis=0)
print(col_means)           # Prints [2. 3.5 5.]
print(col_means.shape)     # Prints (3,)
                           # Has a smaller rank than x!
```

```
mean_shifted = x - col_means
print('\n', mean_shifted)
print(mean_shifted.shape)  # Prints (2, 3)
```

(2,3)와 (3,)의 뺄샘

- Rank 가 다르면 부족한 rank를 보유한 행렬 앞에 1을 추가하여 shape의 길이가 같게 만듦
- Compatible :
  - 두 행렬의 차원이 같을 때 혹은
  - 두 행렬 중 다른 차원의 값이 1을 포함 할 때
- Compatible 하면 broad casting이 가능함
- Compatible 중 1을 가지고 있는 더 작은 차원은 더 큰 행렬의 차원으로 자기자신을 복제 함
- 출력은 :
  - 두 입력의 shape의 요소별 max와 같음

1) 더 작은 rank를 가진 col_means를 (1,3)으로 바꿈 , rank추가

2) 더 작은 행렬 (1,3)은 (2,3)으로 0차원을 따라 broad casting

3) 연산

Broad casting Rule

모두의연구소

```
x = np.array([[1, 2, 3],
              [3, 5, 7]])
print(x.shape)  # Prints (2, 3)

row_means = x.mean(axis=1)
print(row_means)          # Prints [2. 5.]
print(row_means.shape)    # Prints (2,)

# Results in the following error: ValueError:
mean_shifted = x - row_means
```

연산 실패 (2,3)와 (2,)의 뺄샘

- Rank 가 다르면 부족한 rank를 보유한 행렬 앞에 1을 추가하여 shape의 길이가 같게 만듦
- Compatible :
  - 두 행렬의 차원이 같을 때 혹은
  - 두 행렬 중 다른 차원의 값이 1을 포함 할 때
- Compatible 하면 broad casting이 가능함
- Compatible 중 1을 가지고 있는 더 작은 차원은 더 큰 행렬의 차원으로 자기자신을 복제 함
- 출력은 :
  - 두 입력의 shape의 요소별 max와 같음

1) 더 작은 rank를 가진 row_means를 (1,2)으로 바꿈 , rank추가
2) (2,3)과 (1,2)의 경우 마지막 차원의 3과 2가 맞지 않음
3) 연산 실패

Broad casting Rule

모두의연구소

```
x = np.array([[1, 2, 3],
              [3, 5, 7]])
print(x.shape)  # Prints (2, 3)

row_means = x.mean(axis=1).reshape((-1, 1))
print(row_means)         # Prints [[2.], [5.]]
print(row_means.shape)   # Prints (2, 1)

mean_shifted = x - row_means
print(mean_shifted)
print(mean_shifted.shape)  # Prints (2, 3)
```

- Rank 가 다르면 부족한 rank를 보유한 행렬 앞에 1을 추가하여 shape의 길이가 같게 만듦
- Compatible :
  - 두 행렬의 차원이 같을 때 혹은
  - 두 행렬 중 다른 차원의 값이 1을 포함 할 때
- Compatible 하면 broad casting이 가능함
- Compatible 중 1을 가지고 있는 더 작은 차원은 더 큰 행렬의 차원으로 자기자신을 복제 함
- 출력은 :
  - 두 입력의 shape의 요소별 max와 같음

해결

1) row_means를 (1,2)가 아닌 (2,1)로 reshape

2) (2,3)과 (2,1)의 경우 (2,1)이 더 큰 형렬 형태로 복사됨

3) 연산 성공

Views vs. Copies

모두의연구소

- View : 데이터가 공유됨. 일반적인 Copy와 다름
- View는 array slicing 그리고 view를 통한 데이터 타입 전환 시 만들어진다
  - arr.view(dtype) -> View
  - arr.astype(dtype) -> Copy

Views vs. Copies

모두의연구소

- View : 데이터가 공유됨. 일반적인 Copy와 다름

```
x = np.arange(5)
print('Original:\n', x)  # Prints [0 1 2 3 4]

# Modifying the view will modify the array
view = x[1:3]
view[1] = -1
print('Array After Modified View:\n', x)  # Prints [0 1 -1 3 4]
```

```
Original:
 [0 1 2 3 4]
Array After Modified View:
 [ 0  1 -1  3  4]
```

```
x = np.arange(5)
view = x[1:3]
view[1] = -1

# Modifying the array will modify the view
print('View Before Array Modification:\n', view)  # Prints [1 -1]
x[2] = 10
print('Array After Modifications:\n', x)          # Prints [0 1 10 3 4
print('View After Array Modification:\n', view)   # Prints [1 10]
```

Views vs. Copies

- View : 데이터가 공유됨. 일반적인 Copy와 다름

```
x = np.arange(5)
print('Original:\n', x)  # Prints [0 1 2 3 4]

# Modifying the result of the selection due to fancy indexing
# will not modify the original array.
copy = x[[1, 2]]
copy[1] = -1
print('Copy:\n', copy) # Prints [1 -1]
print('Array After Modified Copy:\n', x)  # Prints [0 1 2 3 4]
```

```
Original:
 [0 1 2 3 4]
Copy:
 [ 1 -1]
Array After Modified Copy:
 [0 1 2 3 4]
```

```
# Another example involving fancy indexing
x = np.arange(5)
print('Original:\n', x)  # Prints [0 1 2 3 4]

copy = x[x >= 2]
print('Copy:\n', copy) # Prints [2 3 4]
x[3] = 10
print('Modified Array:\n', x)  # Prints [0 1 2 10 4]
print('Copy After Modified Array:\n', copy)  # Prints [2 3 4]
```

```
Original:
 [0 1 2 3 4]
Copy:
 [2 3 4]
Modified Array:
 [ 0  1  2 10  4]
Copy After Modified Array:
 [2 3 4]
```

Summary

모두의연구소

- Numpy는 빠른 연산을 편리하게 해주는 라이브러리 입니다
- Vectorization 방법은 일반적인 방법에 비해 수십 배 빠르게 동작합니다
- Numpy 가 수 많은 종류의 함수들을 제공합니다
- Broadcasting으로 다른 크기의 행렬간 연산이 가능합니다
- Views 와 Copy를 주의하여야 합니다

모두의연구소

Linear Classification

모두의연구소
박은수 Research Director

돌아보기

모두의연구소

- 고양이 분류 룰을 만들기 어려움 ..

우리는 Data Driven 방법을 취할 겁니다

```
def train(train_images, train_labels):
  # build a model for images -> labels...
  return model

def predict(model, test_images):
  # predict test_labels using the model...
  return test_labels
```

* From : Lecture slide 2 of "CS231n : Convolutional Neural Networks for Visual Recognition"

Parametric Approach

모두의연구소

- 분류기의 구성
  - Score function
  - Loss function
  - Optimization

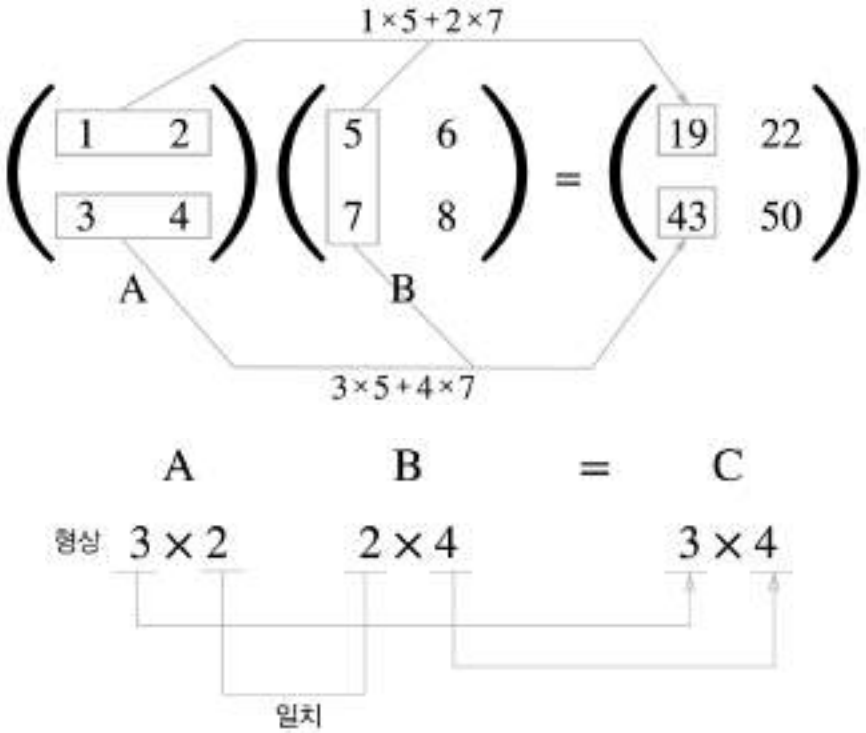
모두의연구소

들어가기 전에 ....

다차원 배열의 계산

모두의연구소

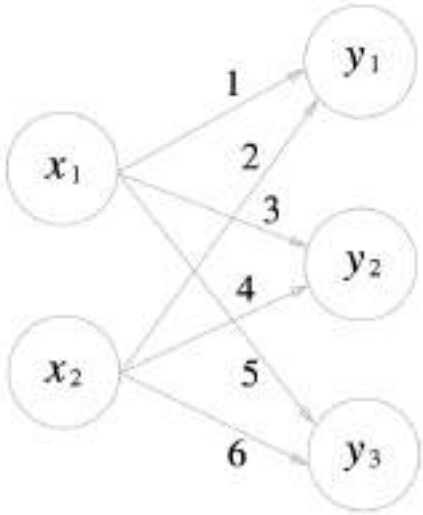
- 다차원 배



다차원 배열의 계산

모두의연구소

- **다차원 배열**



$$\begin{pmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{pmatrix}$$

$$\underset{2}{X} \quad \underset{2\times 3}{W} \quad = \quad \underset{3}{Y}$$

일치

From : Deep Learning from the Scratch

Matrix 연산 리뷰 : Matrix-vector

모두의연구소

Example

$$\begin{bmatrix} 1 & 3 \\ 4 & 0 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 5 \end{bmatrix} = \begin{bmatrix} 16 \\ 4 \\ 7 \end{bmatrix}$$

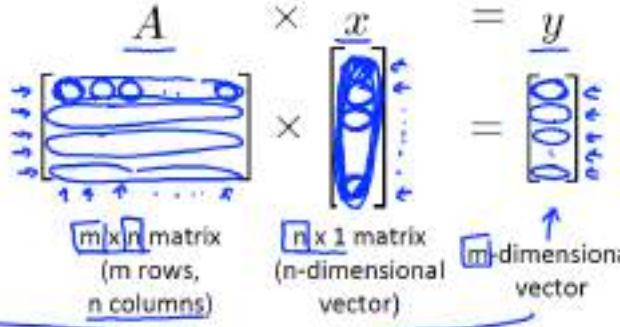
3×2 matrix × 2×1 matrix = 3×1 matrix

$1 \times 1 + 3 \times 5 = 16$

$4 \times 1 + 0 \times 5 = 4$

$2 \times 1 + 1 \times 5 = 7$

Details:



To get y_i, multiply A's i^{th} row with elements of vector x, and add them up.

Matrix 연산 리뷰 : Matrix-vector

모두의연구소

Example

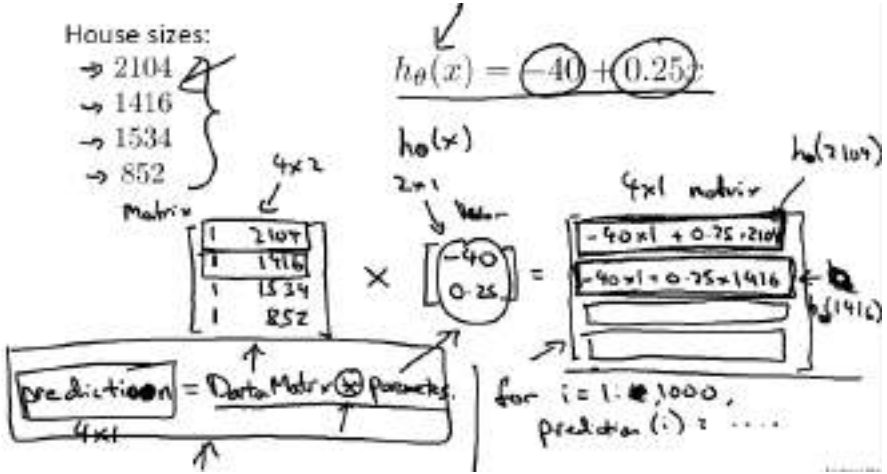
$$\begin{bmatrix} 1 & 2 & 1 & 5 \\ 0 & 3 & 0 & 4 \\ -1 & -2 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \\ 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 14 \\ 12 \\ -3 \end{bmatrix} = \begin{bmatrix} 14 \\ 13 \\ -7 \end{bmatrix}$$

3 × 4 · 4 × 1 · 3 × 1

$1 \times 1 + 2 \times 3 + 1 \times 2 + 5 \times 1 = 14$

$0 \times 1 + 3 \times 3 + 0 \times 2 + 4 \times 1 = 13$

$-1 \times 1 + (-2) \times 3 + 0 \times 2 + 0 \times 1 = -7$



Matrix 연산 리뷰 : Matrix-matrix

모두의연구소

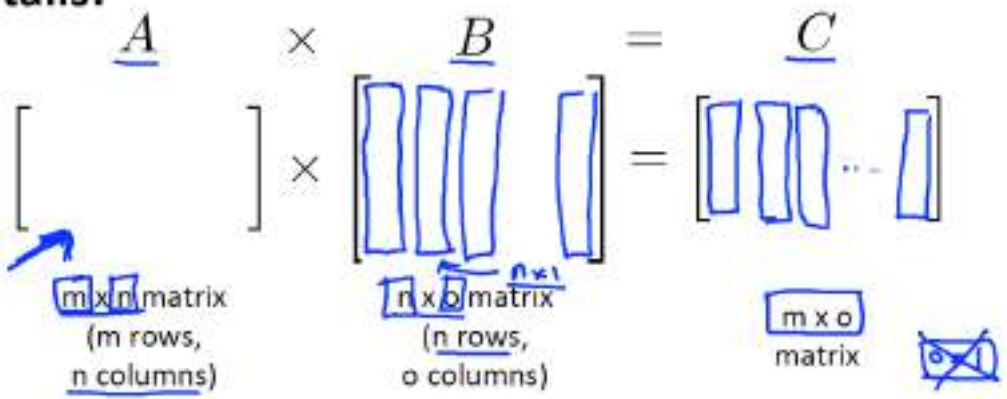
Example

$$\begin{bmatrix}1 & 3 & 2\\ 4 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 3\\ 0 & 1\\ 5 & 2\end{bmatrix} = \begin{bmatrix}11 & 10\\ 9 & 14\end{bmatrix}$$

$$\begin{bmatrix}1 & 3 & 2\\ 4 & 0 & 1\end{bmatrix} \times \begin{bmatrix}1\\ 0\\ 5\end{bmatrix} = \begin{bmatrix}11\\ 9\end{bmatrix}$$

$$\begin{bmatrix}1 & 3 & 2\\ 4 & 0 & 1\end{bmatrix} \times \begin{bmatrix}3\\ 1\\ 2\end{bmatrix} = \begin{bmatrix}10\\ 14\end{bmatrix}$$

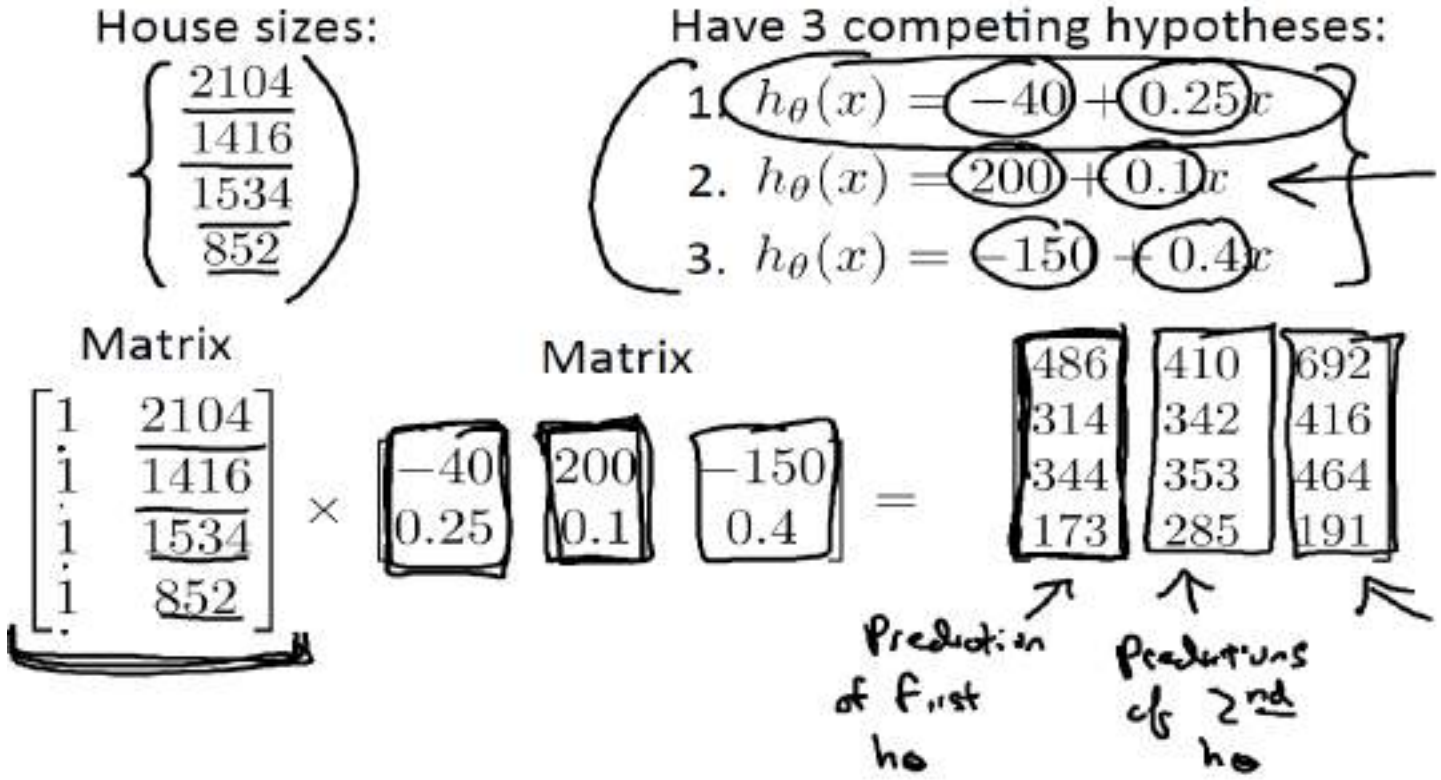
Details:



The i^{th} column of the matrix C is obtained by multiplying A with the i^{th} column of B. (for i = 1,2,...,o)

From : Andrew Ng, "Machine Learning : Linear Algebra review", Lecture note; Coursera

Matrix 연산 리뷰 : Matrix-matrix

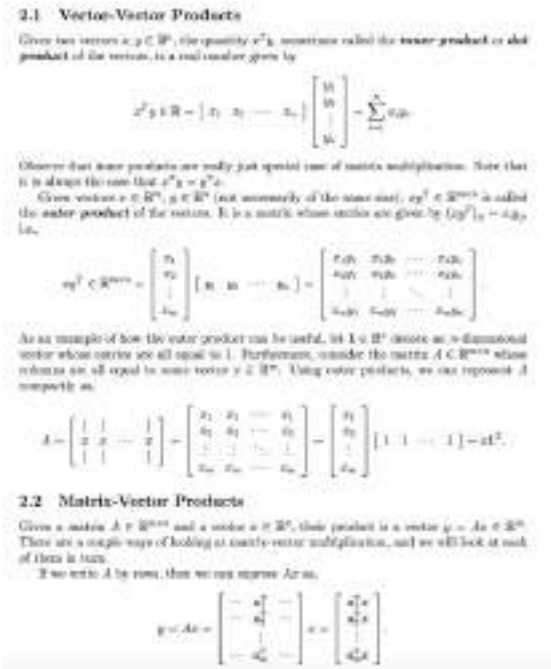


선형대수 리뷰

모두의연구소

http://cs229.stanford.edu/section/cs229-linalg.pdf

좀 더 깊은 리뷰를 보시려면 참고하세요



2.1 Vector-Vector Products

2.2 Matrix-Vector Products

다시 돌아와서 …

모두의연구소

픽셀값들을 조합하여 단순히 각 레이블에 대한 점수를 부여하면 어떻까요? Score function?

Parametric approach

모두의연구소

- Score function

f( , W)

- 강아지 점수
- 고양이 점수

f( , W)

- 강아지 점수
- 고양이 점수

⋮ ⋮ ⋮

점수가 높은것으로 분류

Parametric approach

모두의연구소

- Score function : Simple Linear Classifier

$$f(x, W) = Wx \quad 3{,}072 \times 1$$

32x32x3 영상 =
3,072 픽셀

1	155
2	200
...	...
3071	110
3072	78

x

$f(x, W)$

- 강아지 점수
- 고양이 점수

Parametric approach

모두의연구소

- Score function : Simple Linear Classifier

$$\underset{2\times1}{f(x, W)} = \underset{3,072\times1}{Wx}$$

32x32x3 영상 =
3,072 픽셀

1	155
2	200
...	...
3071	110
3072	78

x

f(x, W)

- 강아지 점수
- 고양이 점수

0.3
0.7

Parametric approach

- Score function : Simple Linear Classifier

$$\underset{2\times1}{f(x, W)} = \underset{2\times3{,}072}{W}\underset{3{,}072\times1}{x}$$

32x32x3 영상 =
3,072 픽셀

1	155
2	200
...	...
3071	110
3072	78

x

f(x, W)

0.2	0.1	...	0.3	0.7
0.7	0.8	...	0.9	0.4

- 강아지 점수
- 고양이 점수

0.3
0.7

Parametric approach

모두의연구소

- Score function : Simple Linear Classifier

3,072x1

2x1 $f(x, W) = Wx + b$ 2x1

2x3,072

32x32x3 영상 =
3,072 픽셀

1	155
2	200
...	...
3071	110
3072	78

x

$f(x, W)$

- 강아지 점수
- 고양이 점수

0.3
0.7

0.2	0.1	...	0.3	0.7
0.7	0.8	...	0.9	0.4

Parametric approach

모두의연구소

- Score function : Simple Linear Classifier

$$f(x, W) = Wx + b$$

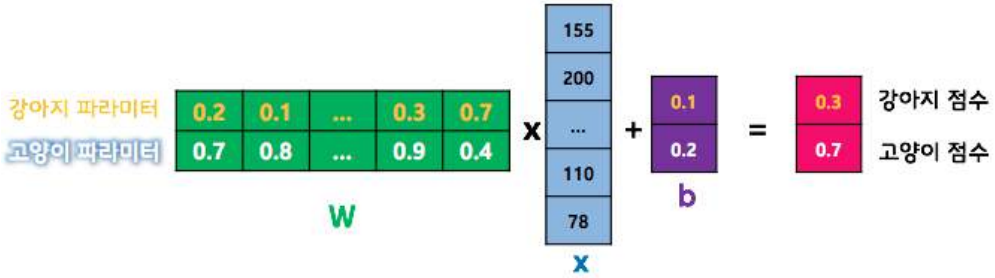
(2x1 = 2x3,072 · 3,072x1 + 2x1)

32x32x3 영상 = 3,072 픽셀

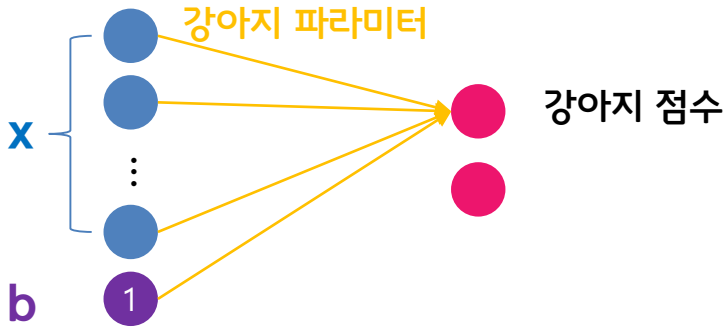
강아지 파라미터	0.2	0.1	...	0.3	0.7	x	155	+	0.1	=	0.3	강아지 점수
고양이 파라미터	0.7	0.8	...	0.9	0.4		200		0.2		0.7	고양이 점수
	W						...		b			
							110					
							78					
							x					

Parametric approach

- Score function : Simple Linear Classifier

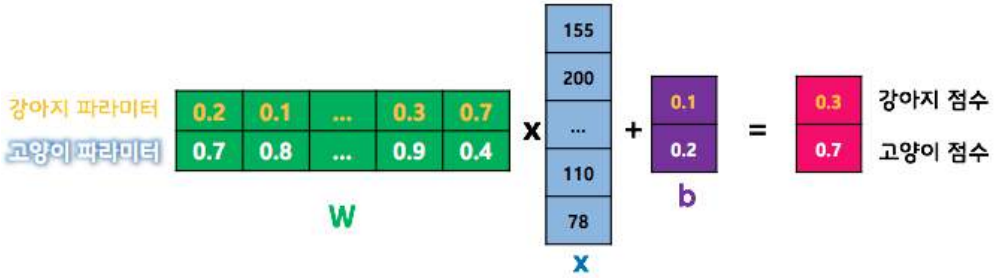


선형분류기의
뉴럴 네트워크 표현

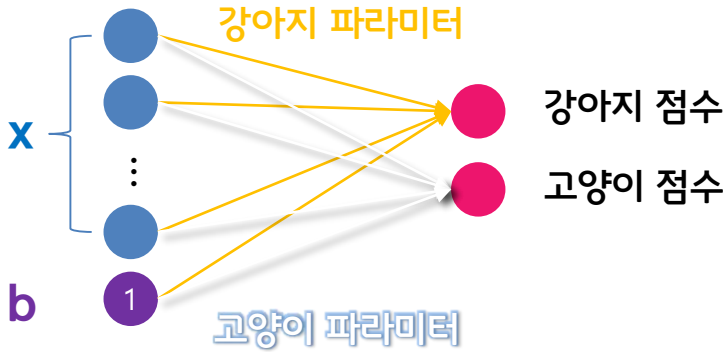


Parametric approach

- Score function : Simple Linear Classifier

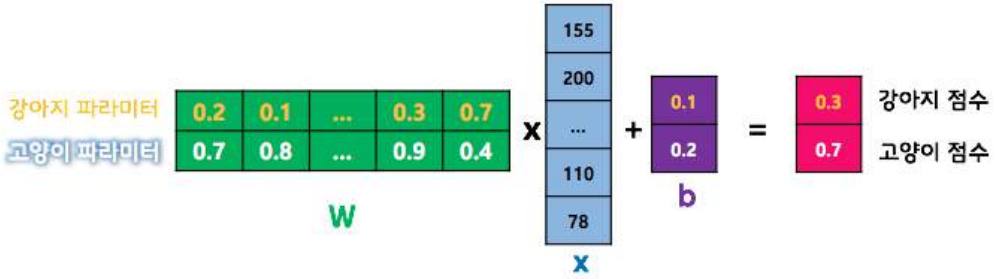


선형분류기의
뉴럴 네트워크 표현

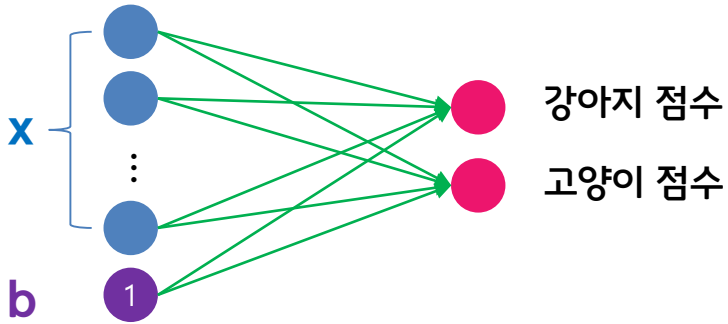


Parametric approach

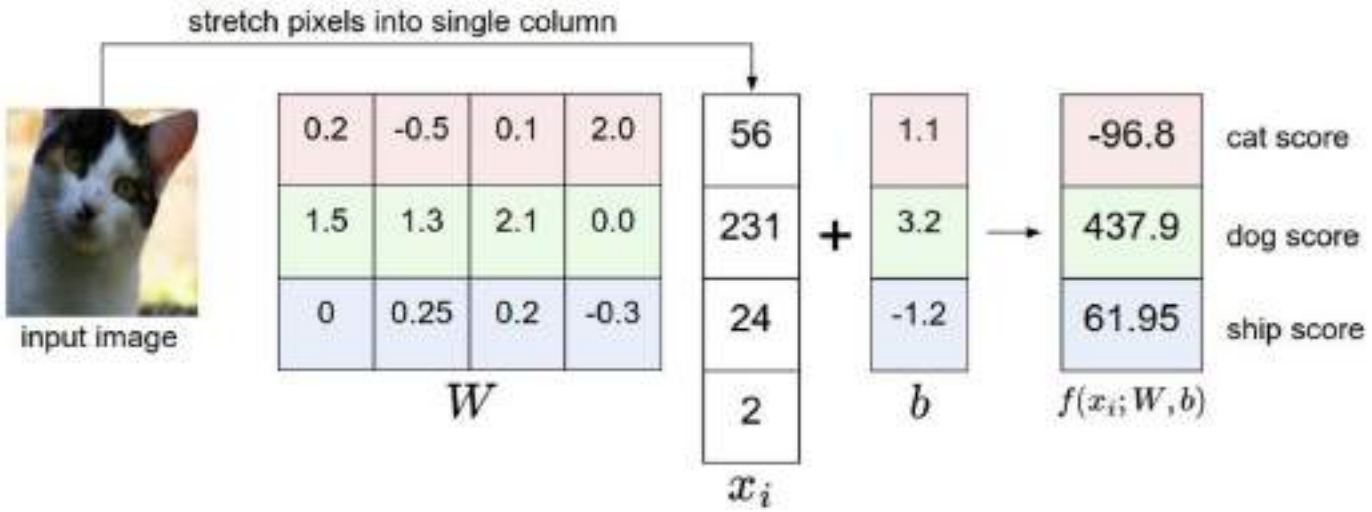
- Score function : Simple Linear Classifier



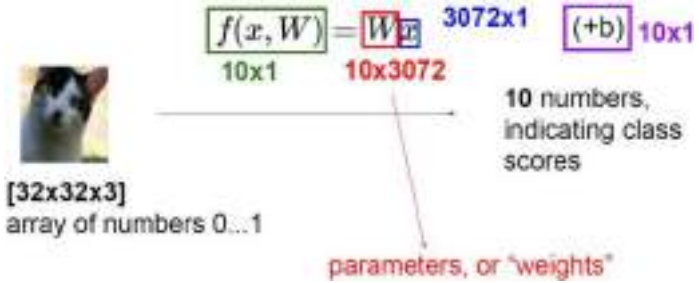
선형분류기의
뉴럴 네트워크 표현



Example with an image with 4 pixels, and 3 classes (cat/dog/ship)



영상이 32x32x3 이라면

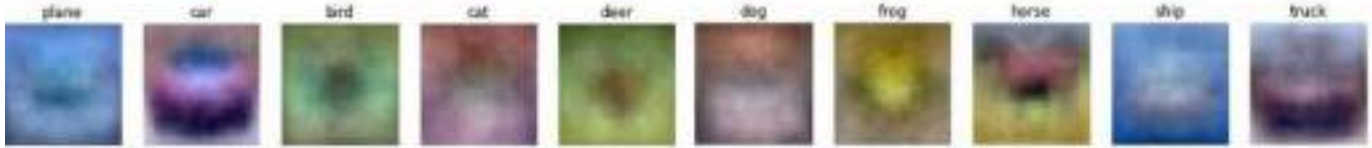


W를 열어보면 ...

Interpreting a Linear Classifier

$$f(x_i, W, b) = Wx_i + b$$

Example trained weights of a linear classifier trained on CIFAR-10:



W를 어떤 템플릿으로 볼 수도 있겠군요...

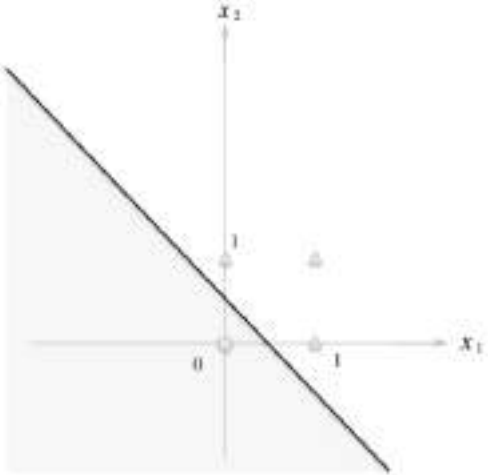
Decision Boundary의 해석

모두의연구소

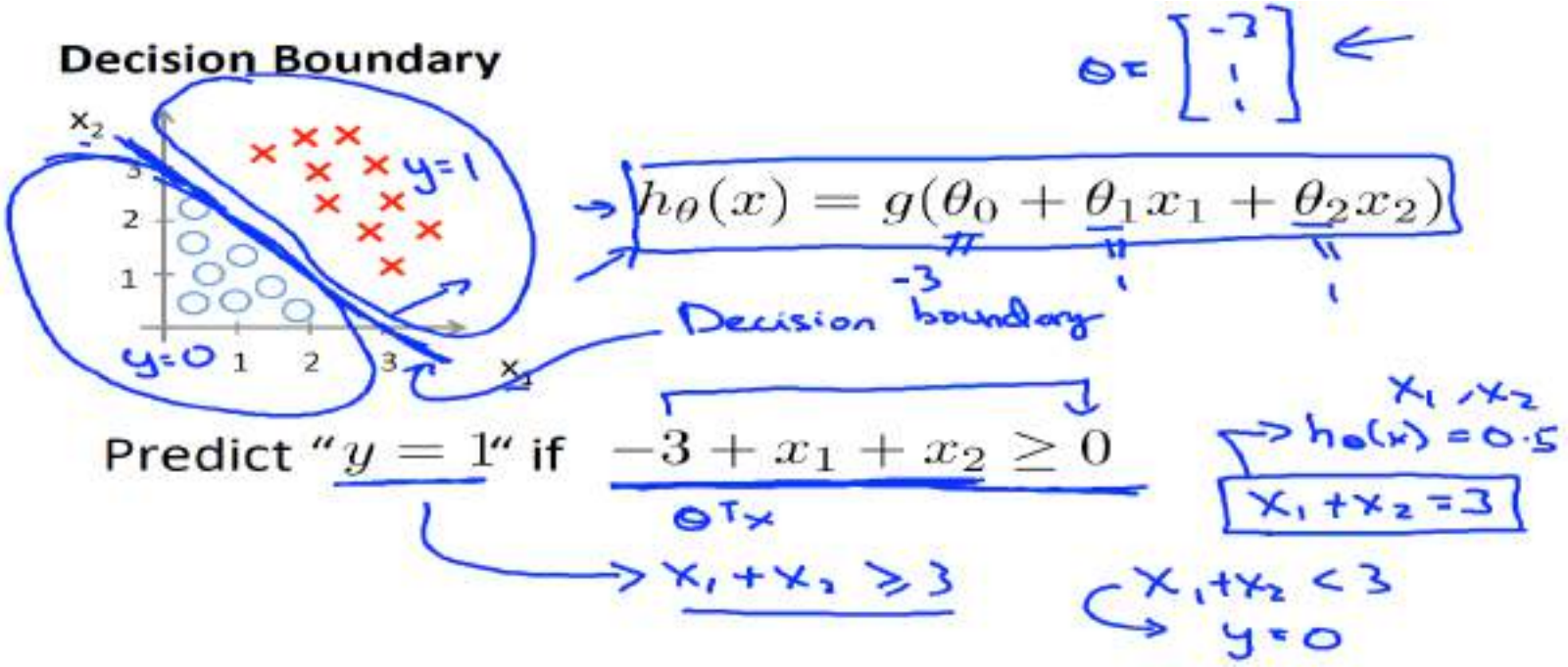
OR 게이트 :

x_1	x_2	y
0	0	0
1	0	1
0	1	1
1	1	1

$$y = \begin{cases} 0 & (-0.5 + x_1 + x_2 \leq 0) \\ 1 & (-0.5 + x_1 + x_2 > 0) \end{cases}$$



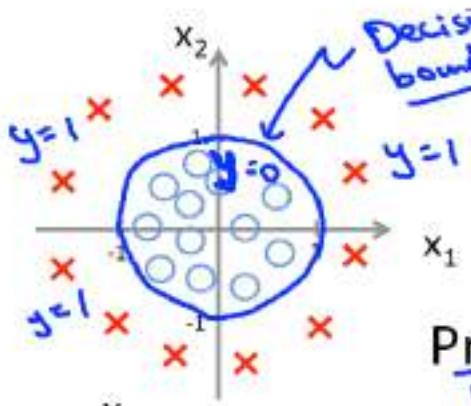
Decision Boundary의 해석



Decision Boundary의 해석

모두의연구소

Non-linear decision boundaries



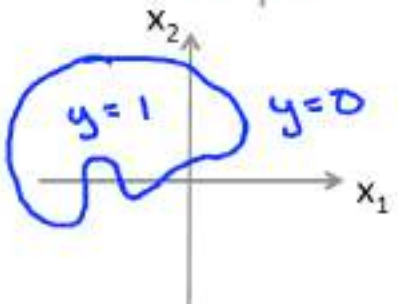
$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$$

$\theta = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$

Predict "$y = 1$" if $-1 + x_1^2 + x_2^2 \geq 0$

$x_1^2 + x_2^2 = 1$

$x_1^2 + x_2^2 \geq 1$

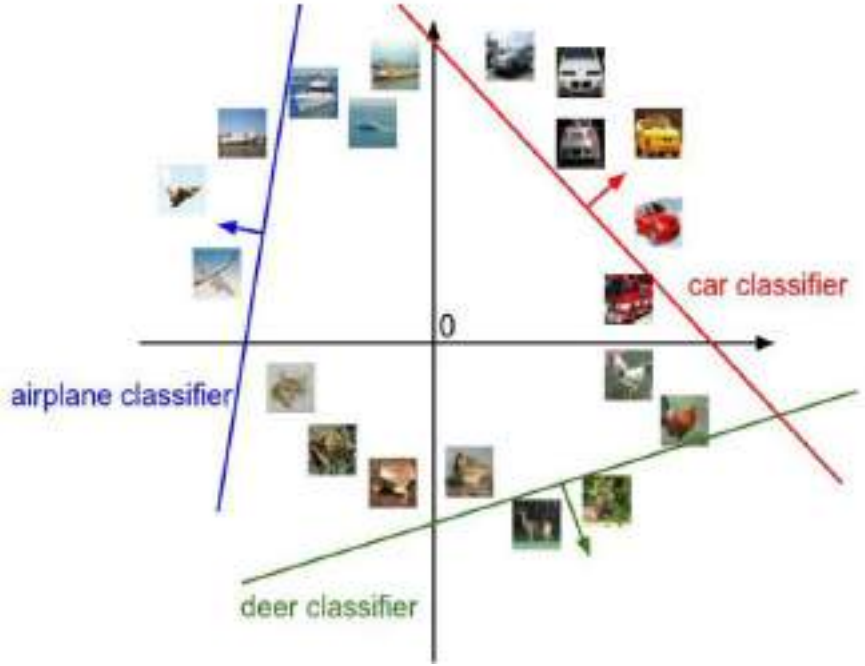


$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_1^2 x_2 + \theta_5 x_1^2 x_2^2 + \theta_6 x_1^3 x_2 + \dots)$$

같은 방식으로 해석될 수 있습니다

모두의연구소

Interpreting a Linear Classifier



$$f(x_i, W, b) = Wx_i + b$$

[32x32x3]
array of numbers 0...1
(3072 numbers total)

지금까지 : Score function (Linear Classifier) 을 알아봤습니다

모두의연구소



$$f(x_i, W, b) = Wx_i + b$$

Example class scores for 3 images, with a random W:

airplane	-3.45	-0.51	3.42
automobile	-8.87	**6.04**	4.64
bird	0.09	5.31	2.65
cat	**2.9**	-4.22	5.1
deer	4.48	-4.19	2.64
dog	8.02	3.58	5.55
frog	3.78	4.49	**-4.34**
horse	1.06	-4.37	-1.5
ship	-0.36	-2.09	-4.79
truck	-0.72	-2.93	6.14

Score Function 구현해보기

모두의연구소

- 1_day/prac0_score_function.py

$$\mathbf{A}^{(1)} = \mathbf{X}\mathbf{W}^{(1)} + \mathbf{B}^{(1)}$$

```python
import numpy as np

X = np.array([1.0, 0.5])
W = np.array([[0.1, 0.3, 0.5], [0.2, 0.4, 0.6]])
B = np.array([0.1, 0.2, 0.3])

print(X.shape)
print(W.shape)
print(B.shape)

# matrix multiplication : np.dot
# element wise multiplication : just *
# matrix add : just +

# your code
A = # make A
```

정답 : [0.3, 0.7, 1.1]

$$\mathbf{A}^{(1)} = (a_1^{(1)}\ a_2^{(1)}\ a_3^{(1)})$$

$$\mathbf{X} = (x_1\ \ x_2)$$

$$\mathbf{B}^{(1)} = (b_1^{(1)}\ b_2^{(1)}\ b_3^{(1)})$$

$$\mathbf{W}^1 = \begin{pmatrix} w_{11}^{(1)} & w_{21}^{(1)} & w_{31}^{(1)} \\ w_{12}^{(1)} & w_{22}^{(1)} & w_{32}^{(1)} \end{pmatrix}$$

Coming up:

- Loss function
- Optimization

$$f(x, W) = Wx$$

- Loss Function : 현재 내가 구성한 W가 좋은지 안좋은지 측정할 수 있게 해줍니다
- Optimization : random W로 시작해서 Loss를 최소로 갖게 W를 변경해 줍니다

모두의연구소

Loss Function

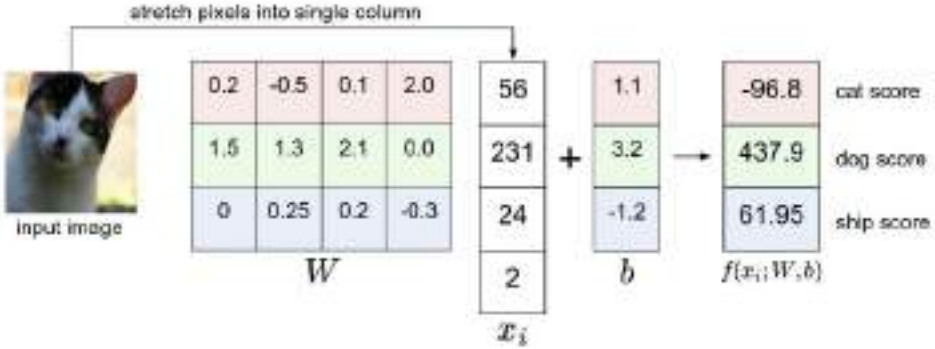
모두의연구소
박은수 Research Director

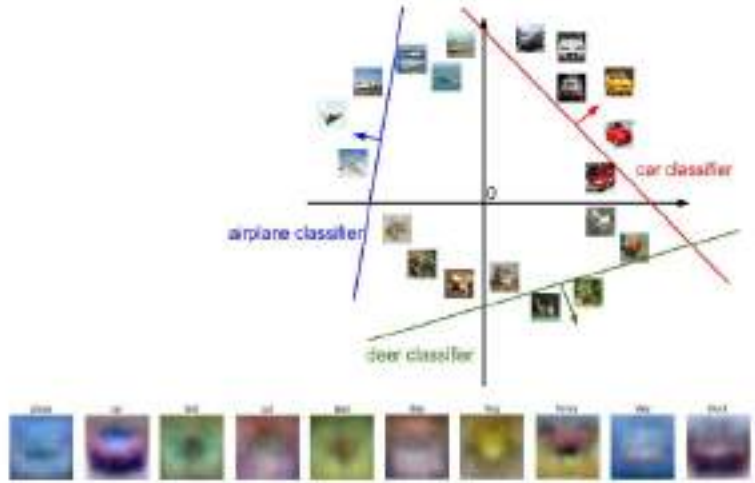
돌아보기 ...

[32x32x3]
array of numbers 0...1
(3072 numbers total)

image parameters
f(x,W)

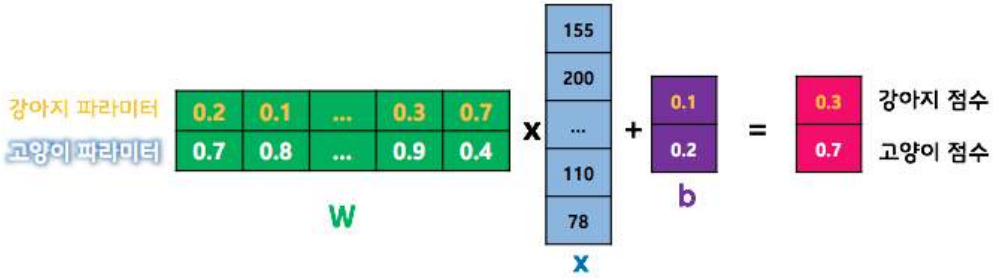
10 numbers, indicating
class scores



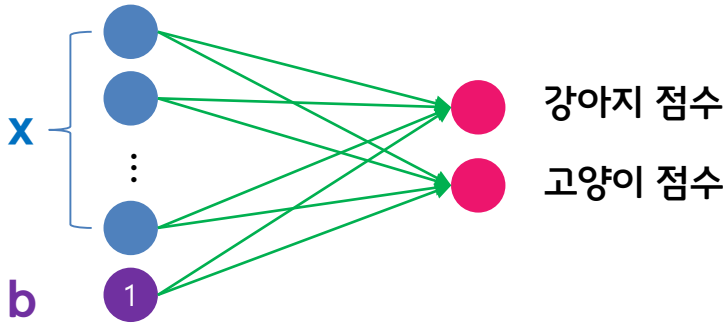


돌아보기 ...

- Score function



선형분류기의
뉴럴 네트워크 표현



- 분류기의 구성
  - Score function
  - Loss function
  - Optimization

Loss Function

- 측정된 Score가 얼마나 안좋은지를 나타내는 하는 함수
- Optimization은 이 측정된 Loss function의 값을 줄이는 방법

이제 해야 할 것은

모두의연구소

Score

airplane	-3.45	-0.51	3.42
automobile	-8.87	**6.04**	4.64
bird	0.09	5.31	2.65
cat	**2.9**	-4.22	5.1
deer	4.48	-4.19	2.64
dog	8.02	3.58	5.55
frog	3.78	4.49	**-4.34**
horse	1.06	-4.37	-1.5
ship	-0.36	-2.09	-4.79
truck	-0.72	-2.93	6.14

1. 계산된 Score가 얼마나 안좋은지 측정할 수 있는 함수를 만들어야 합니다

   ➡ Loss function

2. Loss function을 최소화 할수 있는 방법을 만들어야 합니다

   (Optimization)

이제 해야 할 것은

Score

airplane	-3.45	-0.51	3.42
automobile	-8.87	**6.04**	4.64
bird	0.09	5.31	2.65
cat	**2.9**	-4.22	5.1
deer	4.48	-4.19	2.64
dog	8.02	3.58	5.55
frog	3.78	4.49	**-4.34**
horse	1.06	-4.37	-1.5
ship	-0.36	-2.09	-4.79
truck	-0.72	-2.93	6.14

1. 계산된 Score가 얼마나 후진지 알려줄 함수를 만들어야 합니다
   ➡ Loss function

2. Loss function을 최소화 할수 있는 방법을 만들어야 합니다 (Optimization)

모두의연구소

첫번째로 살펴 볼 Loss function은 SVM Loss 입니다

$$L_i = \sum_{j \neq y_i} \max(0, f(x_i, W)_j - f(x_i, W)_{y_i} + \Delta)$$

SVM (Hinge) Loss는 어떤 의미를 갖는가

$$L_i = \sum_{j \neq y_i} \max(0, f(x_i, W)_j - f(x_i, W)_{y_i} + \Delta)$$

i 번째 이미지의 Loss

정답이 아닌 것들만 합한다

정답이 아닌 score와 정답 score와의 차이

- 최소한 Loss가 0이 아닌 값을 가지려면 정답이 아닌 녀석들이 저 붉은 delta위치에는 오거나 정답 Score보다 크면 되겠네요

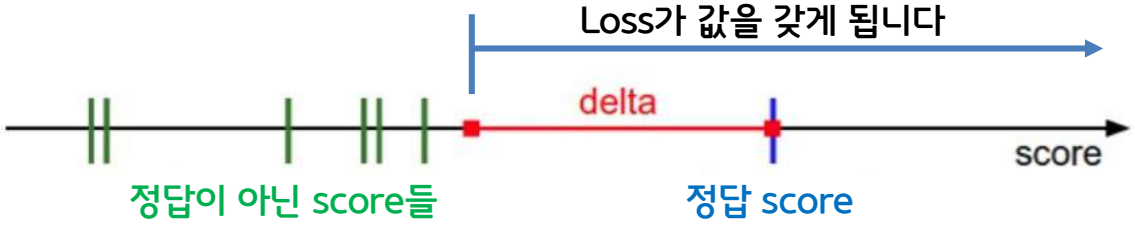
SVM (Hinge) Loss는 어떤 의미를 갖는가

$$L_i = \sum_{j \neq y_i} \max(0, f(x_i, W)_j - f(x_i, W)_{y_i} + \Delta)$$

i 번째 이미지의 Loss

정답이 아닌 것들만 합한다

정답이 아닌 score와 정답 score와의 차이



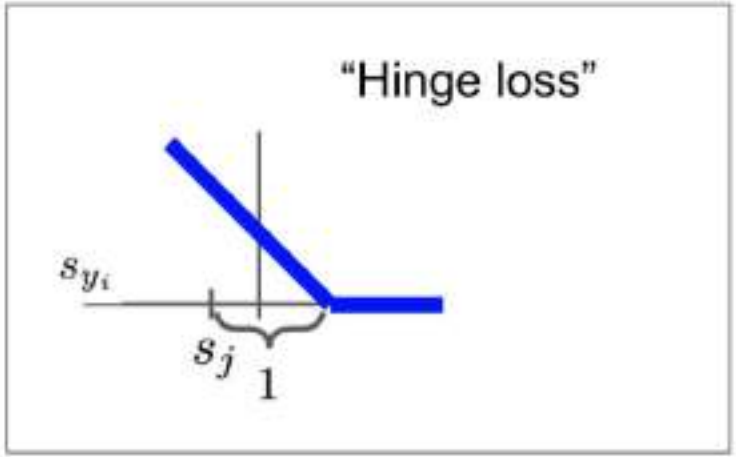
- 최소한 Loss가 0이 아닌 값을 가지려면 정답이 아닌 녀석들이 저 붉은 delta위치에는 오거나 정답 Score보다 크면 되겠네요

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**

Multiclass SVM loss:



$$L_i = \sum_{j \neq y_i} \begin{cases} 0 & \text{if } s_{y_i} \geq s_j + 1 \\ s_j - s_{y_i} + 1 & \text{otherwise} \end{cases}$$

$$= \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**

A **loss function** tells how good our current classifier is

Given a dataset of examples

$$\{(x_i, y_i)\}_{i=1}^{N}$$

Where x_i is image and
y_i is (integer) label

Loss over the dataset is a sum of loss over examples:

$$L = \frac{1}{N} \sum_i L_i(f(x_i, W), y_i)$$

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**

Multiclass SVM loss:

Given an example (x_i, y_i)
where x_i is the image and
where y_i is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \begin{cases} 0 & \text{if } s_{y_i} \geq s_j + 1 \\ s_j - s_{y_i} + 1 & \text{otherwise} \end{cases}$$

$$= \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**

Multiclass SVM loss:

Given an example (x_i, y_i)
where x_i is the image and
where y_i is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9		

Multiclass SVM loss:

Given an example (x_i, y_i)
where x_i is the image and
where y_i is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

= max(0, 5.1 - 3.2 + 1)
+max(0, -1.7 - 3.2 + 1)
= max(0, 2.9) + max(0, -3.9)
= 2.9 + 0
= 2.9

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	

Multiclass SVM loss:

Given an example (x_i, y_i)
where x_i is the image and
where y_i is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

= max(0, 1.3 - 4.9 + 1)
+max(0, 2.0 - 4.9 + 1)
= max(0, -2.6) + max(0, -1.9)
= 0 + 0
= 0

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	12.9

Multiclass SVM loss:

Given an example (x_i, y_i)
where x_i is the image and
where y_i is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

= max(0, 2.2 - (-3.1) + 1)
+max(0, 2.5 - (-3.1) + 1)
= max(0, 6.3) + max(0, 6.6)
= 6.3 + 6.6
= 12.9

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	12.9

Multiclass SVM loss:

Given an example (x_i, y_i)
where x_i is the image and
where y_i is the (integer) label,

and using the shorthand for the scores vector: $s = f(x_i, W)$

the SVM loss has the form:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Loss over full dataset is average:

$$L = \frac{1}{N} \sum_{i=1}^{N} L_i$$

L = (2.9 + 0 + 12.9)/3
= **5.27**



$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	12.9

Q1 : 자동차의 스코어가 조금 변했을 때 각 L_i 는 어떻게 될까?

모두의연구소



cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	12.9

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Q1 : 자동차의 스코어가 조금 변했을 때 각 L_i 는 어떻게 될까?

Q2 : loss function L 이 가질 수 있는 min 값과 max값은 ?

모두의연구소



cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	12.9

≈ 0

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Q1 : 자동차의 스코어가 조금 변했을 때 각 L_i 는 어떻게 될까?

Q2 : loss function L_i가 가질 수 있는 min 값과 max값은 ?

Q3 : W의 초기값이 매우 작아서 s 값들이 전부 0과 거의 같은 값을 갖게 될 때 L_i 값은 무엇과 유사할까?

모두의연구소



cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	12.9

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Q1 : 자동차의 스코어가 조금 변했을 때 각 L_i 는 어떻게 될까?

Q2 : loss function L_i가 가질 수 있는 min 값과 max값은 ?

Q3 : W의 초기값이 매우 작아서 s 값들이 전부 0과 거의 같은 값을 갖게 될 때 L_i 값은 무엇과 유사할까?

Q4 : $\sum_{j \neq y_i}$ 가 아니라 $\sum_j$ 일 경우 (즉 정답 레이블을 s_j에 포함) L_i 는?

모두의연구소



cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	12.9

$$\max(0, s_j - s_{y_i})^2$$

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

모두의연구소

Q1 : 자동차의 스코어가 조금 변했을 때 각 L_i 는 어떻게 될까?

Q2 : loss function L_i가 가질 수 있는 min 값과 max값은 ?

Q3 : W의 초기값이 매우 작아서 s 값들이 전부 0과 거의 같은 값을 갖게 될 때 L_i 값은 무엇과 유사할까?

Q4 : $\sum_{j \neq y_i}$가 아니라 $\sum_j$ 일 경우 L_i 는?

Q5 : L_i 계산 시 합이 아니라 평균으로 하면?

Q6 : L_i 계산 시 제곱해서 더하면 ?

$$f(x, W) = Wx$$

$$L = \frac{1}{N} \sum_{i=1}^{N} \sum_{j \neq y_i} \max(0, f(x_i; W)_j - f(x_i; W)_{y_i} + 1)$$

E.g. Suppose that we found a W such that L = 0. Is this W unique?

$$f(x, W) = Wx$$

$$L = \frac{1}{N}\sum_{i=1}^{N}\sum_{j \neq y_i} \max(0, f(x_i; W)_j - f(x_i; W)_{y_i} + 1)$$

E.g. Suppose that we found a W such that L = 0. Is this W unique?

No! 2W is also has L = 0!

Suppose: 3 training examples, 3 classes.
With some W the scores $f(x, W) = Wx$ are:



cat	**3.2**	1.3	2.2
car	5.1	**4.9**	2.5
frog	-1.7	2.0	**-3.1**
Losses:	2.9	0	

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1)$$

Before:

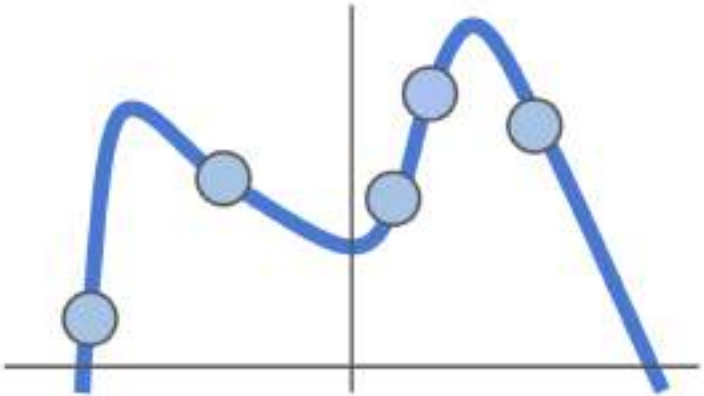
= max(0, 1.3 - 4.9 + 1)
+max(0, 2.0 - 4.9 + 1)
= max(0, -2.6) + max(0, -1.9)
= 0 + 0
= 0

With W twice as large:

= max(0, 2.6 - 9.8 + 1)
+max(0, 4.0 - 9.8 + 1)
= max(0, -6.2) + max(0, -4.8)
= 0 + 0
= 0

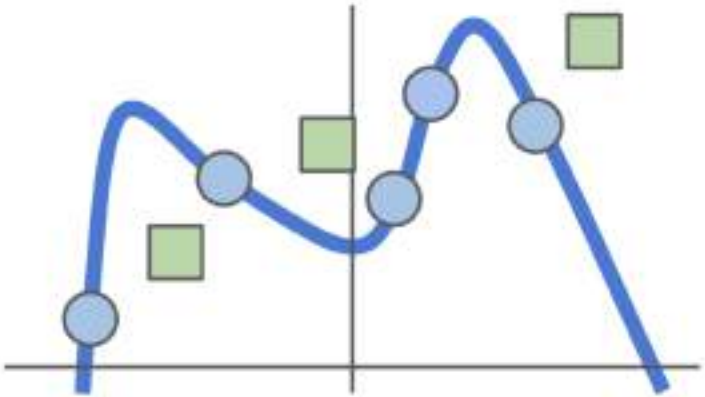
$$L(W) = \underbrace{\frac{1}{N}\sum_{i=1}^{N} L_i(f(x_i, W), y_i)}$$

Data loss: Model predictions should match training data



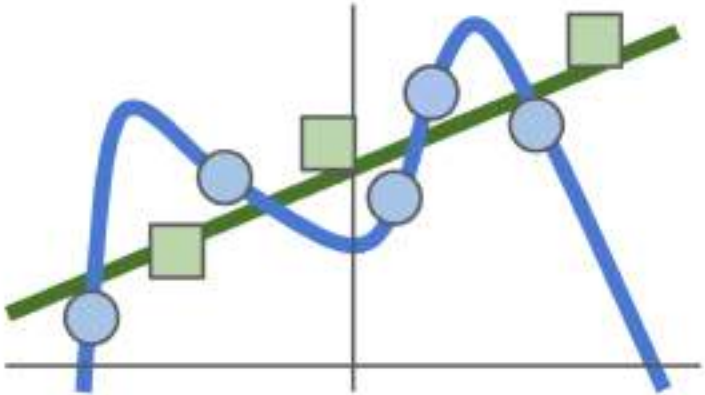
$$L(W) = \underbrace{\frac{1}{N}\sum_{i=1}^{N} L_i(f(x_i, W), y_i)}$$

Data loss: Model predictions should match training data



$$L(W) = \underbrace{\frac{1}{N}\sum_{i=1}^{N} L_i(f(x_i, W), y_i)}_{\text{Data loss}} + \underbrace{\lambda R(W)}_{\text{Regularization}}$$

Data loss: Model predictions should match training data



Regularization: Model should be "simple", so it works on test data

Occam's Razor:
"Among competing hypotheses, the simplest is the best"
William of Ockham, 1285 - 1347

Regularization

λ = regularization strength (hyperparameter)

$$L = \frac{1}{N}\sum_{i=1}^{N}\sum_{j\neq y_i}\max(0, f(x_i; W)_j - f(x_i; W)_{y_i} + 1) + \boxed{\lambda R(W)}$$

In common use:

L2 regularization $R(W) = \sum_k \sum_l W_{k,l}^2$

L1 regularization $R(W) = \sum_k \sum_l |W_{k,l}|$

Elastic net (L1 + L2) $R(W) = \sum_k \sum_l \beta W_{k,l}^2 + |W_{k,l}|$

Max norm regularization (might see later)

Dropout (will see later)

Fancier: Batch normalization, stochastic depth

L2 Regularization (Weight Decay)

$$x = [1, 1, 1, 1]$$

$$R(W) = \sum_k \sum_l W_{k,l}^2$$

$$w_1 = [1, 0, 0, 0]$$

$$w_2 = [0.25, 0.25, 0.25, 0.25]$$

(If you are a Bayesian: L2 regularization also corresponds MAP inference using a Gaussian prior on W)

$$w_1^T x = w_2^T x = 1$$

Softmax Loss

- Cross entropy loss -

$$P(Y = k|X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$$ where $$s = f(x_i; W)$$

Want to maximize the log likelihood, or (for a loss function) to minimize the negative log likelihood of the correct class:

$$L_i = -\log P(Y = y_i|X = x_i)$$

강아지 점수 2.3

고양이 점수 -1.2

---

in summary: $$L_i = -\log(\frac{e^{sy_i}}{\sum_j e^{s_j}})$$

Softmax Loss

- Cross entropy loss -

$$L_i = -\log\left(\frac{e^{sy_i}}{\sum_j e^{s_j}}\right)$$

강아지 점수	2.3	exp →	9.97
고양이 점수	-1.2		0.3

전부다 양수가 됨

모두의연구소

Softmax Loss

- Cross entropy loss -

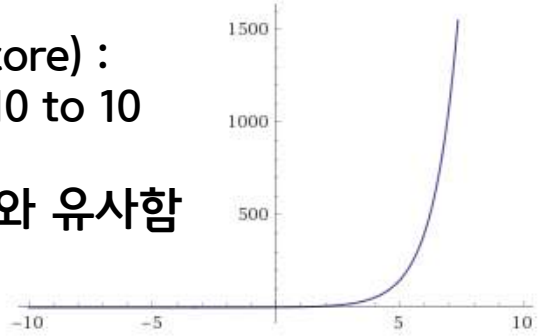
$$L_i = -\log\left(\frac{e^{sy_i}}{\sum_j e^{s_j}}\right)$$

강아지 점수	2.3	exp	9.97
고양이 점수	-1.2		0.3

전부다 양수가 됨

exp(score) :
from -10 to 10

Max함수와 유사함



Softmax Loss

- Cross entropy loss -

$$L_i = -\log\left(\frac{e^{sy_i}}{\sum_j e^{s_j}}\right)$$

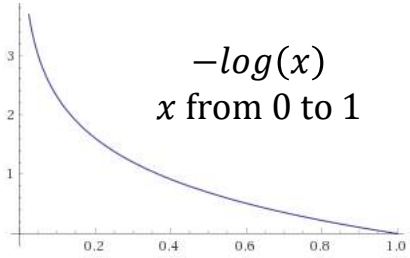
강아지 점수	2.3	exp →	9.97	normalize →	0.97
고양이 점수	-1.2		0.3		0.03
			전부다 양수가 됨		확률

Softmax Loss

- Cross entropy loss -

모두의연구소

$$L_i = -\log\left(\frac{e^{sy_i}}{\sum_j e^{s_j}}\right)$$



강아지 점수	2.3	exp →	9.97	normalize →	0.97
고양이 점수	-1.2		0.3		**0.03**
			전부다 양수가 됨		**확률**

$L_i = -log(0.03) = 3.5$

Softmax 구현

모두의연구소

- 소프트 맥스 함수 구현 시 유의점
  - 오버 플로우(overflow) 문제 : e^{10}은 20,000이 넘고, e^{100}은 0이 40개가 넘는 큰 수이며, e^{1000}은 무한대로 계산됨
  - 이런 큰 값끼리 나눗셈을 하면 결과 수치가 '불안정' 해짐
  - 개선해 봅시다

$$y_k = \frac{\exp(a_k)}{\sum_{i=1}^{n}\exp(a_i)}$$

$$y_k = \frac{\exp(a_k)}{\sum_{i=1}^{n}\exp(a_i)} = \frac{C\exp(a_k)}{C\sum_{i=1}^{n}\exp(a_i)}$$

$$= \frac{\exp(a_k + \log C)}{\sum_{i=1}^{n}\exp(a_i + \log C)}$$

$$= \frac{\exp(a_k + C')}{\sum_{i=1}^{n}\exp(a_i + C')}$$

- 어떤 수를 더해도 그 결과값이 변하지 않음
- 오버플로우를 막기 위해 최대값을 빼면 좋을 것임

Softmax 구현

모두의연구소

- 소프트 맥스 함수 구현 시 유의점

오버 플로우

```
In [1]: import numpy as np

In [2]: a = np.array([1010, 1000, 990])

In [3]: np.exp(a)/np.sum(np.exp(a))
/Users/eunsoo/anaconda/bin/ipython:1: RuntimeWarning: overflow encountered in exp
  #!/Users/eunsoo/anaconda/bin/python
/Users/eunsoo/anaconda/bin/ipython:1: RuntimeWarning: invalid value encountered in divide
  #!/Users/eunsoo/anaconda/bin/python
Out[3]: array([ nan,  nan,  nan])
```

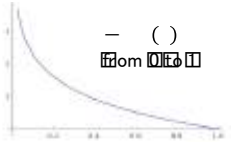
오버 플로우 개선

```
In [4]: c = np.max(a)

In [5]: a-c
Out[5]: array([  0, -10, -20])

In [6]: np.exp(a-c)/np.sum(np.exp(a-c))
Out[6]: array([  9.99954600e-01,   4.53978686e-05,   2.06106005e-09])
```

Softmax 특징

$$L_i = -\log\left(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}}\right)$$

강아지 점수	2.3	exp	9.97	normalize	0.97
고양이 점수	-1.2		0.3		0.03
		전부다 양수가 됨			확률



$_i$ = — (0.03) = 3.5

Softmax

- 출력은 0과 1사이의 실수
- 총합은 1 (확률로 해석가능)
- 입력과 출력의 대소 관계에는 변함이 없음
  - y=exp(x)가 단조 증가함수이기 때문
- 결과적으로 분류(추론) 할 때는 출력층의 소프트 맥스를 생략해도 되며, 일반적으로 생략 함

```
def softmax(x):
    x = x - np.max(x) # 오버플로 대책
    return np.exp(x) / np.sum(np.exp(x))
```

```
In [9]: a = np.array([0.3, 2.9, 4.0])

In [10]: y = softmax(a)

In [11]: print(y)
[ 0.01821127  0.24519181  0.73659691]

In [12]: np.sum(y)
Out[12]: 1.0
```

* 2번 클래스가 확률이 가장 크니 답은 2번째 클래스다

* 단조 증가함수란 정의역 원소 a, b가 $a \leq b$ 일때 $f(a) \leq f(b)$가 성립하는 함수

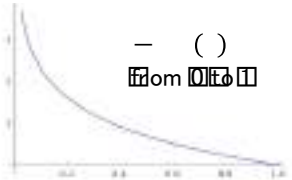
실습 : Softmax Loss

- Cross entropy loss -

모두의연구소

- 1day/prac1_softmax_loss.py

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$



강아지 점수	2.3	exp	9.97	normalize	0.97
고양이 점수	-1.2		0.3		0.03

전부다 양수가 됨

확률

$_i = -$ (0.03) = 3.5

구현해야 할 부분

Softmax

Softmax loss

실습 : Softmax Loss

- Cross entropy loss -

모두의연구소

- 1_day/prac1_softmax_loss.py

$$E = -\sum_{k} t_k \log y_k$$

log : 밑이 e인 자연로그
y_k : 신경망의 출력
t_k : 정답 레이블

- 원-핫 인코딩의 경우 정답에 해당하는 인덱스의 원소만 1이고 나머지는 0이기 때문에 실제적으로 정답에 해당하는 추정치의 자연로그 연산이 됩니다
- 즉, 교차엔트로피 오차는 정답일 때의 출력이 전체 값을 결정하게 됨

실습 : Softmax Loss

- Cross entropy loss -

모두의연구소

- 1_day/prac1_softmax_loss.py

$$E = -\sum_k t_k \log y_k$$

정답이
인덱스 2, 추정도 2

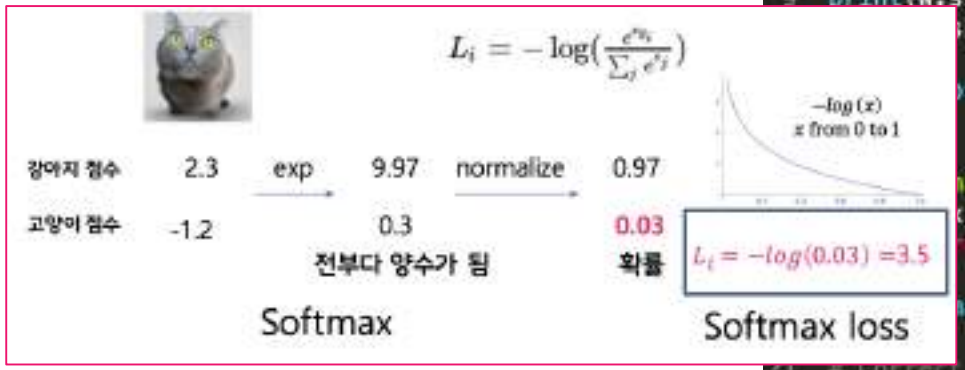
```
In [16]: t = [0, 0, 1, 0, 0, 0, 0, 0, 0, 0]

In [17]: y = [0.1, 0.05, 0.6, 0.0, 0.05, 0.1, 0.0, 0.1, 0.0, 0.0]

In [18]: cross_entropy_error(np.array(y), np.array(t))
Out[18]: 0.51082545709933802
```

정답이
인덱스 2, 추정은 7

```
In [19]: y = [0.1, 0.05, 0.1, 0.0, 0.05, 0.1, 0.0, 0.6, 0.0, 0.0]

In [20]: cross_entropy_error(np.array(y), np.array(t))
Out[20]: 2.3025840929945458
```

실습 : Softmax Loss

- 1_day/prac1_softmax_loss.py



$$E = -\sum_k t_k \log y_k$$

구현해야 할 부분

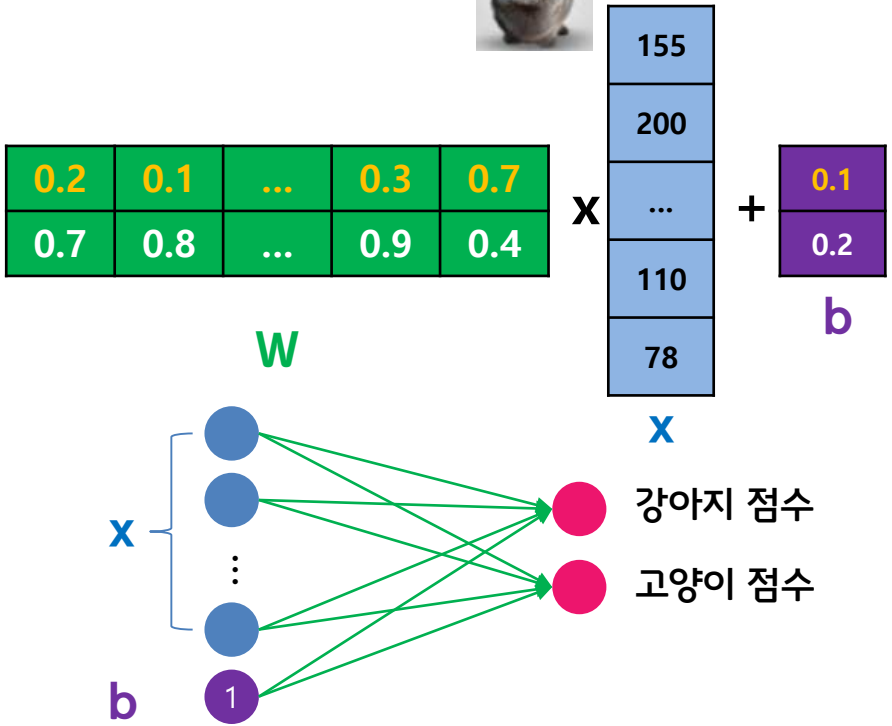
```python
# coding: utf-8
import numpy as np
#%%
X = np.array([1.0, 0.5])
W = np.array([[0.1, 0.3, 0.5], [0.2, 0.4, 0.6]])
B = np.array([0.1, 0.2, 0.3])
#%%
print(X.shape)
print(W.shape)
hape)

t(X, W)+B

ax(x):
  np.max(x) # 오버플로 대책
n np.exp(x) / np.sum(np.exp(x))

ax(A)

# correct one hot vector
t = [0, 0, 1]
#%%
# hint : np.log, np.sum, log(x + delta) for preventing lo
def cross_entropy_error(y, t):
    delta = 1e-7
    loss = 0 # your code
    return loss

#%%
lloss = cross_entropy_error(y, t)
```

강아지와 고양이 분류해보기

모두의연구소

- 분류기의 구성
  - Score function
  - Loss function

고양이가 입력이면 고양이 점수가 높아야 함



강아지와 고양이 분류해보기

모두의연구소

- 분류기의 구성
  - Score function
  - Loss function

0.2	0.1	...	0.3	0.7
0.7	0.8	...	0.9	0.4

W

x

155
200
...
110
78

x

+

0.1
0.2

b

=

Cross-entropy loss (Softmax)

강아지 점수	2.3	exp	9.97	normalize	0.97	강아지 확률
고양이 점수	-1.2		0.3		0.03	고양이 확률

$-log(0.03) = 3.5$

현재의 분류기는 3.5만큼 안 좋음. 이 loss 값을 줄이는게 목표

요약

모두의연구소

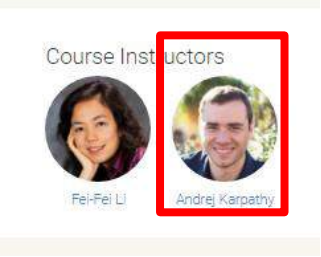
- 분류기의 구성
  - Score function
  - Loss function
  - Optimization

이제 Loss를 최소로 하는 W를 찾는 방법만 남았습니다

모두의연구소

기타 :

영상인식과 관련하여 우리는 어디까지 왔는가 ?



http://karpathy.github.io/2012/10/22/state-of-computer-vision/

박 은 수 Research Director

E-mail : es.park@modulabs.co.kr