

2017 2학기 인공지능 강의노트

3주차

강사: 양일호

강의 계획

주차	강의	실습
1	과목 소개 인공지능 및 python 소개 Python 프로그래밍 환경 조성 및 기본 문법	Python 개발 환경 구축 간단한 expert system 구현
2	Search (Blind Search, Heuristic Search)	DFS, BFS, A* 구현
3	Learning / Simulated Annealing	Simulated Annealing 구현
4	Genetic Algorithm	Genetic Algorithm 구현
5	보강 주간 (개천절 / 추석 연휴)	
6	Neural Network	Perceptron 구현
7	Neural Network	Single-Layer Perceptron 구현
8	Neural Network	Multi-Layer Perceptron 구현
9	Deep Neural Network	DNN 구현
10	Deep Neural Network	DNN 구현
11	DNN 심화 (혹은 Particle Swarm Optimization)	관련 내용 구현
12	DNN 심화 (혹은 Gaussian Mixture Model)	관련 내용 구현
13	DNN 심화 (혹은 Gaussian Mixture Model)	관련 내용 구현
14	DNN 심화 (혹은 Bayesian Networks)	관련 내용 구현
15	DNN 심화 (혹은 Decision Networks)	관련 내용 구현
16	기말고사	

Python
숙달을 위한
준비 기간

예비 기간
(대체 가능)

실습과제 (A* search 구현)

- 다음 기능을 구현(총 10점)
 1. 아래와 같은 미로 구조를 python 코드로 표현
 2. 11을 시작점으로 하여 15까지 A* search로 이동(5점)
 - 좌우상하 4방향으로만 이동 가능
 - 탐색하는 중간 과정 출력(현재 위치, $f(n)$, $g(n)$, $h(n)$ 등)
 - 최종적으로 탐색을 몇 번 수행한 끝에 도착했는지 탐색 횟수 출력
 3. python Tkinter 모듈을 이용하여 GUI 구현(5점)
 - 미로 출력
 - 최종적으로 찾아낸 하나의 이동 경로 출력
(중간 과정은 console에서만 출력해도 됨)

1	2	3		5
6		8		10
11		13		15
16	17	18	19	20
21	22	23		25

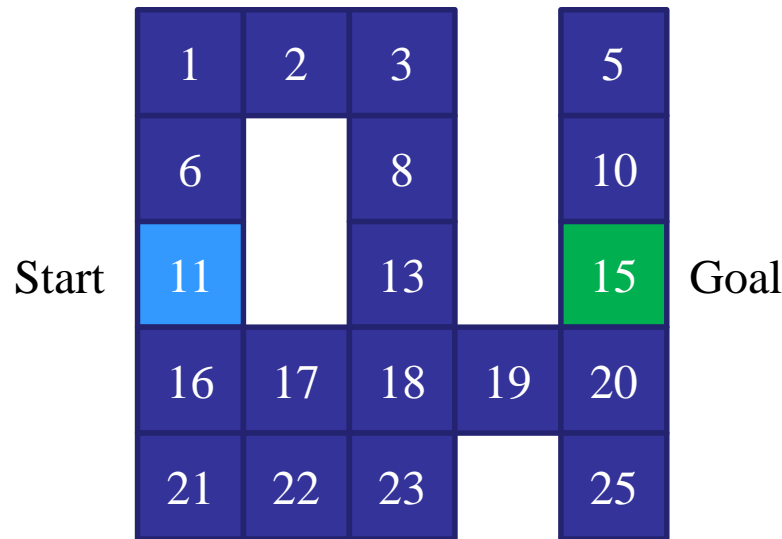
A* search

- Heuristic function
 - $f(n) = g(n) + h(n)$

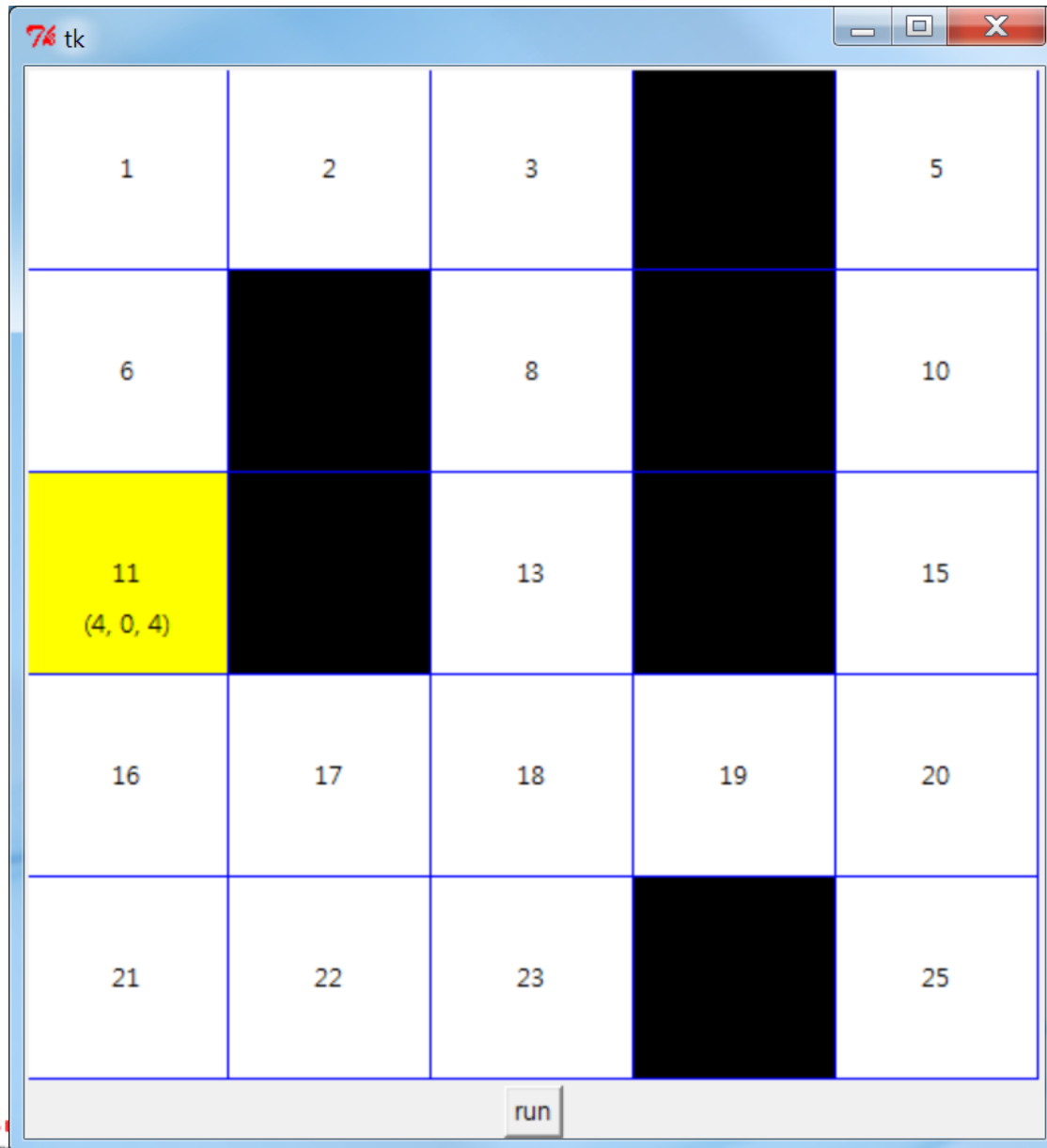
$g(n)$ = 현재 위치(n)까지 이동한 횟수

$h(n)$ = 현재 위치(n)에서 목적지까지 적어도 최소한 이동해야 할 것 같은 횟수

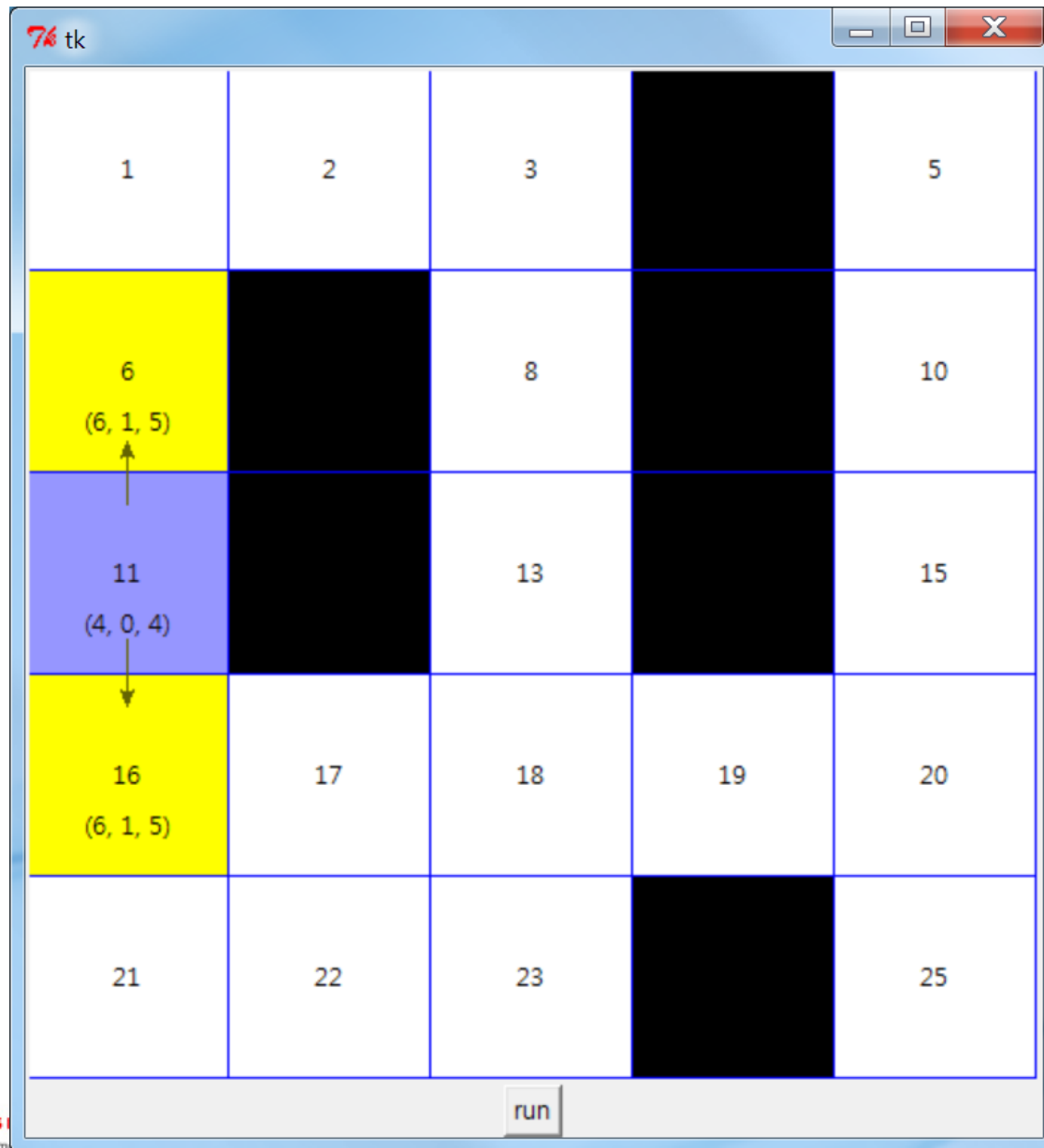
실제 거리보다 $h(n)$ 이 크면 안됨



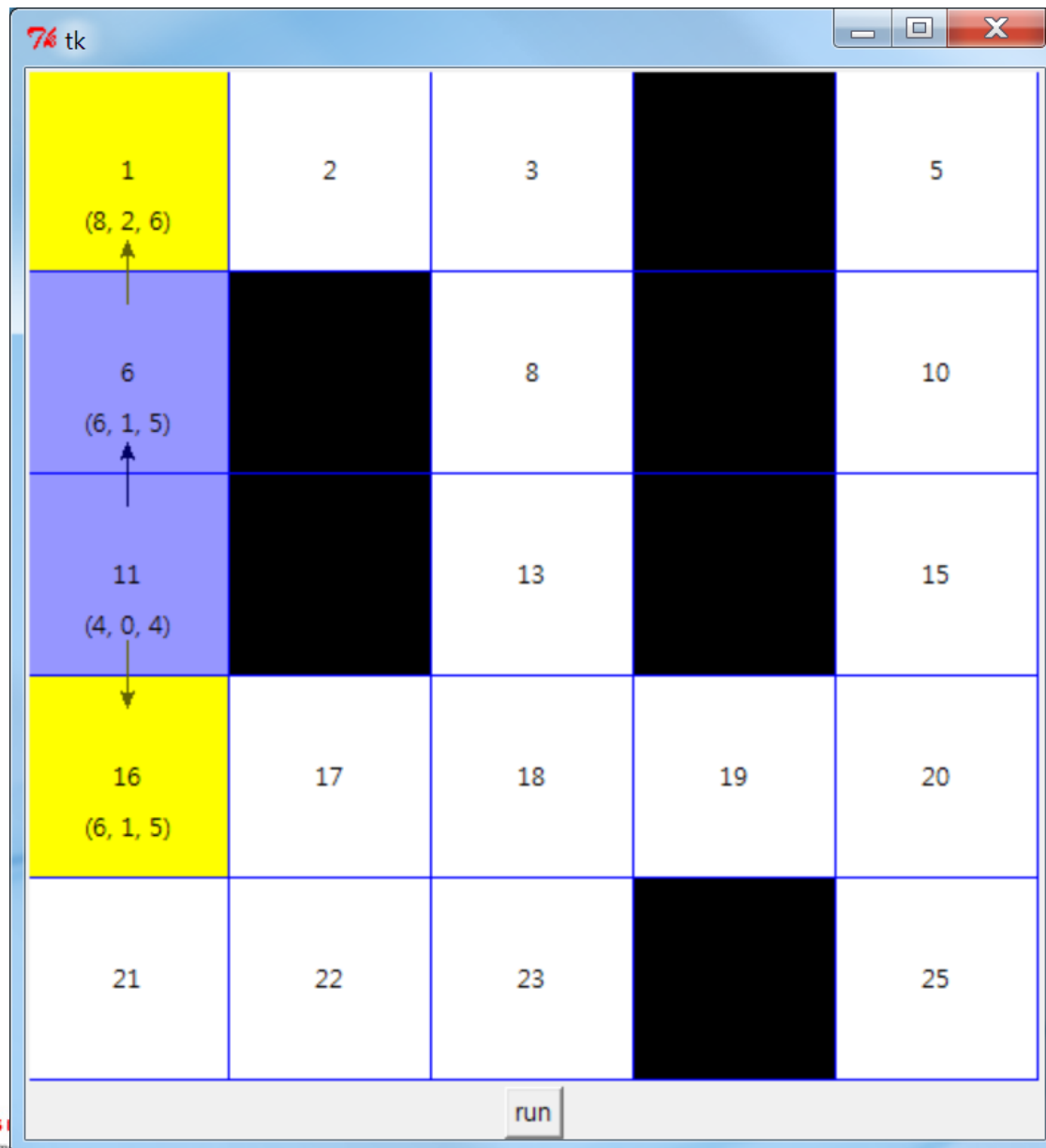
A* search



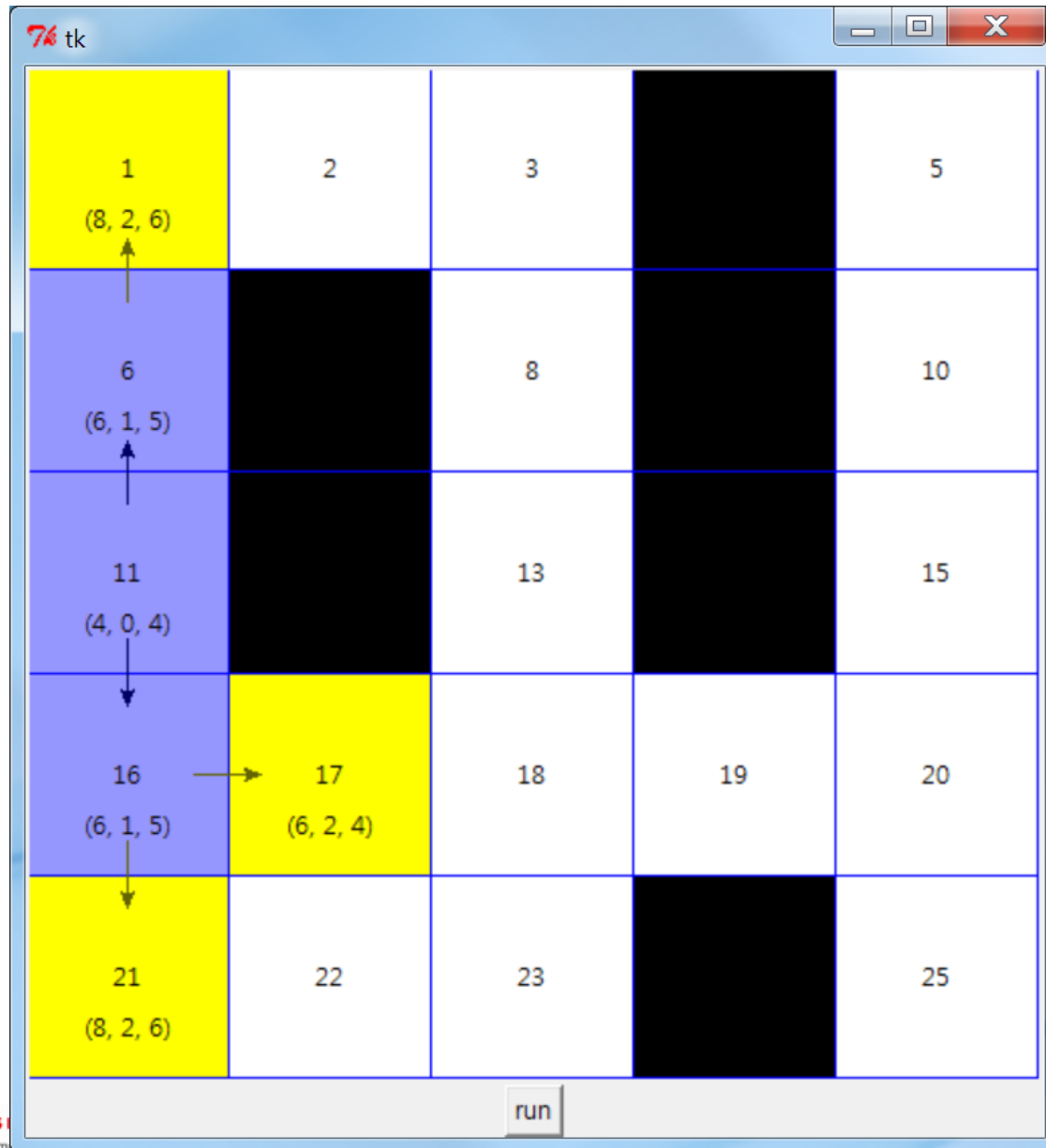
A* search



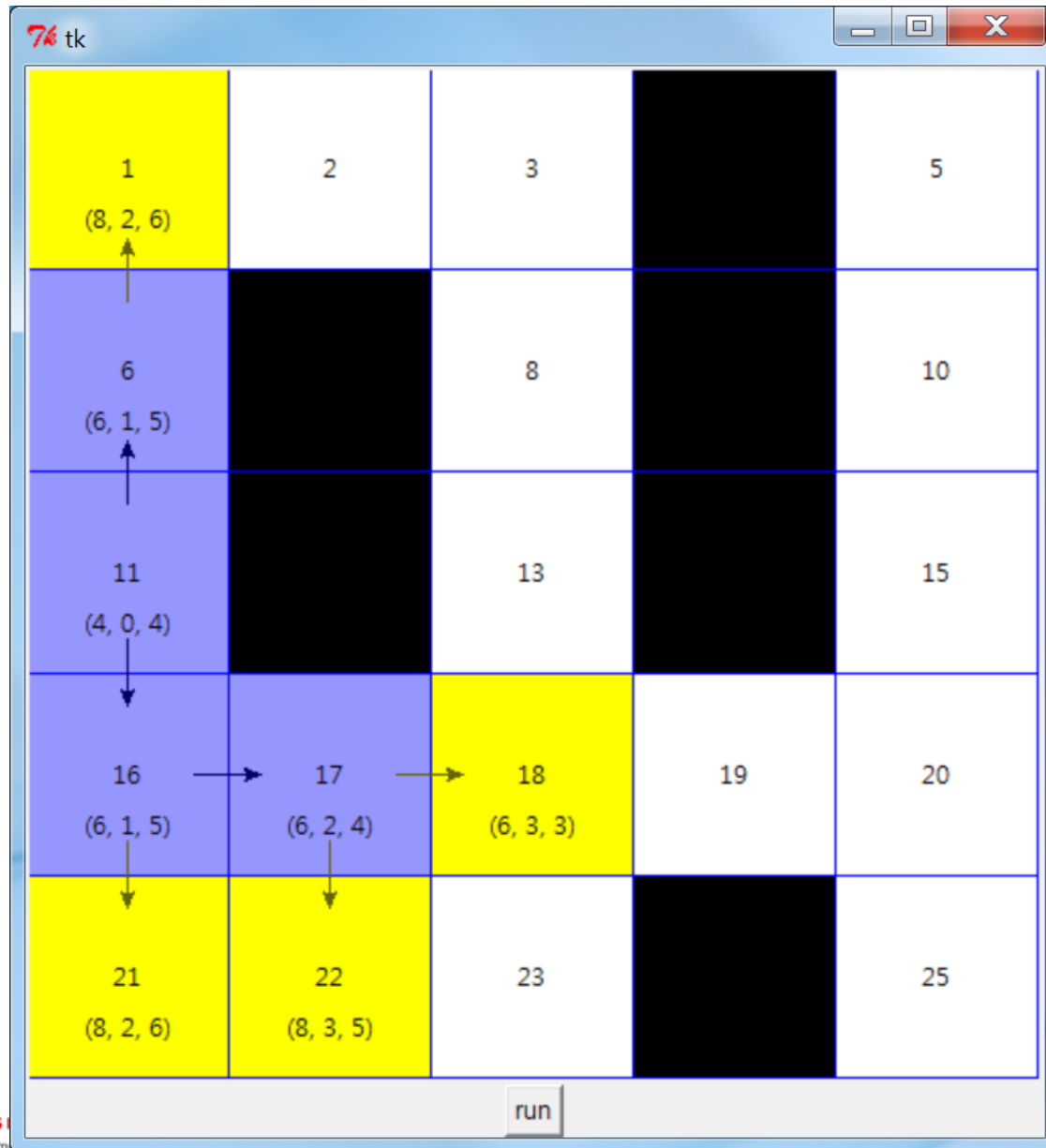
A* search



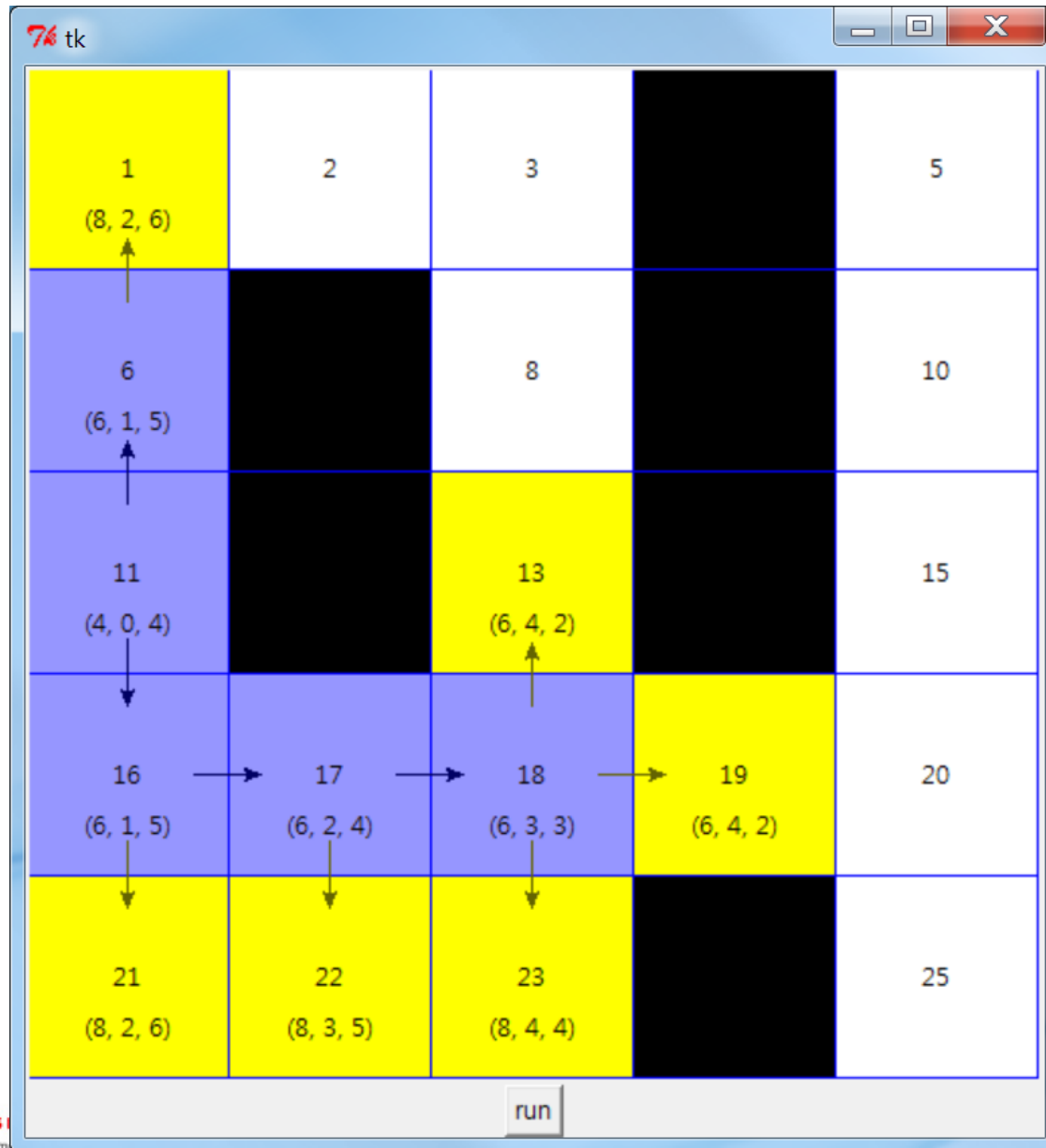
A* search



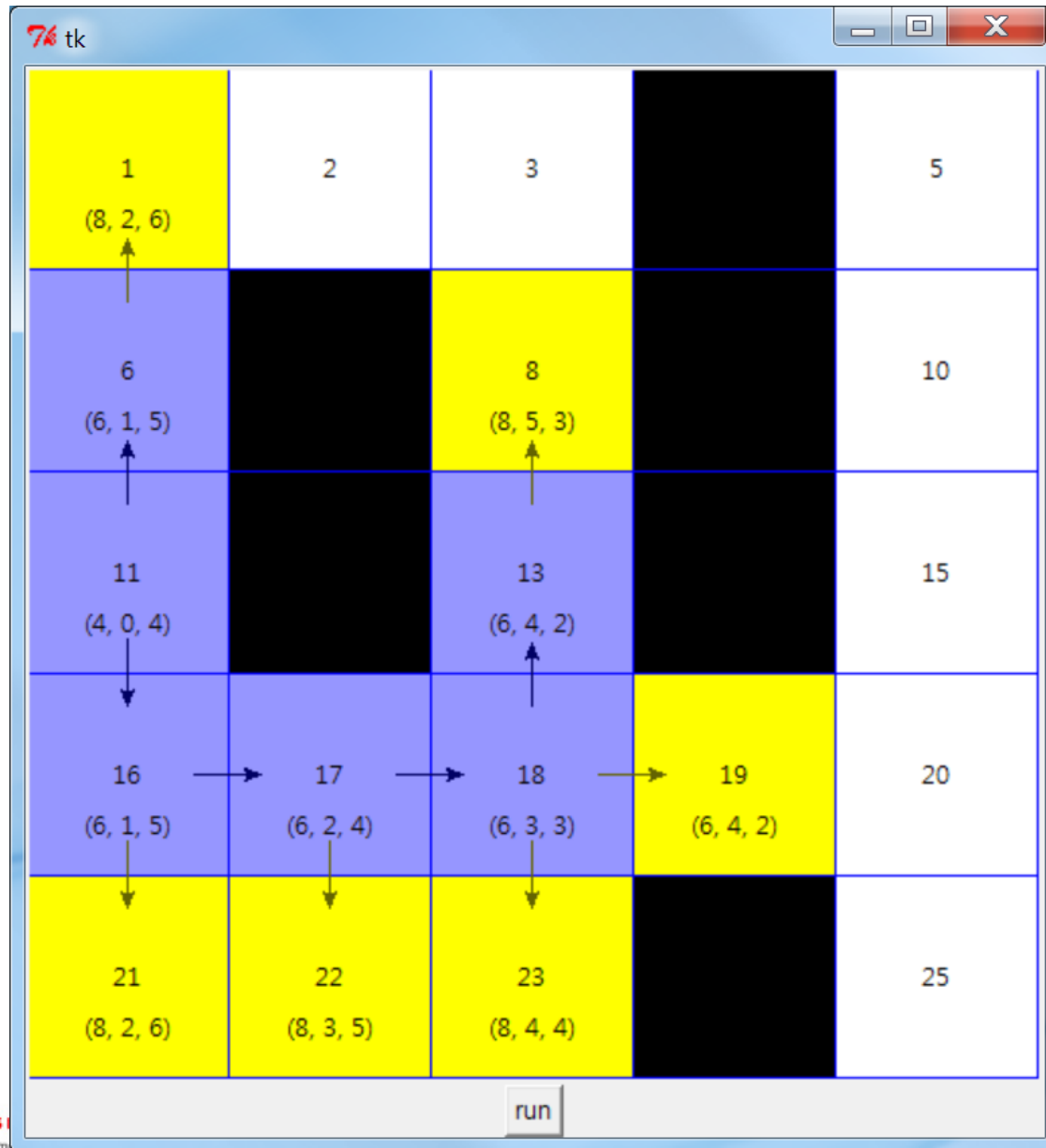
A* search



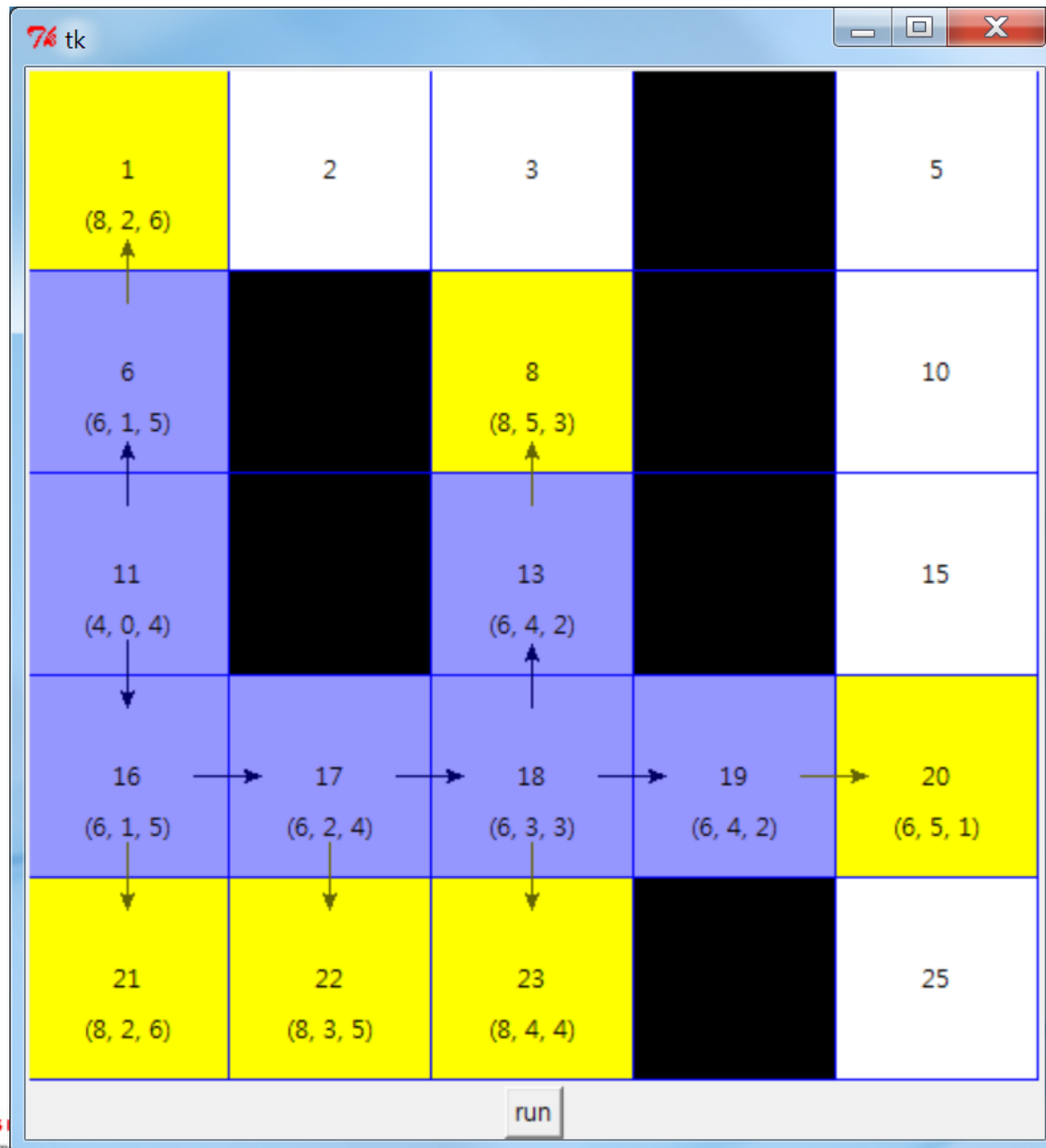
A* search



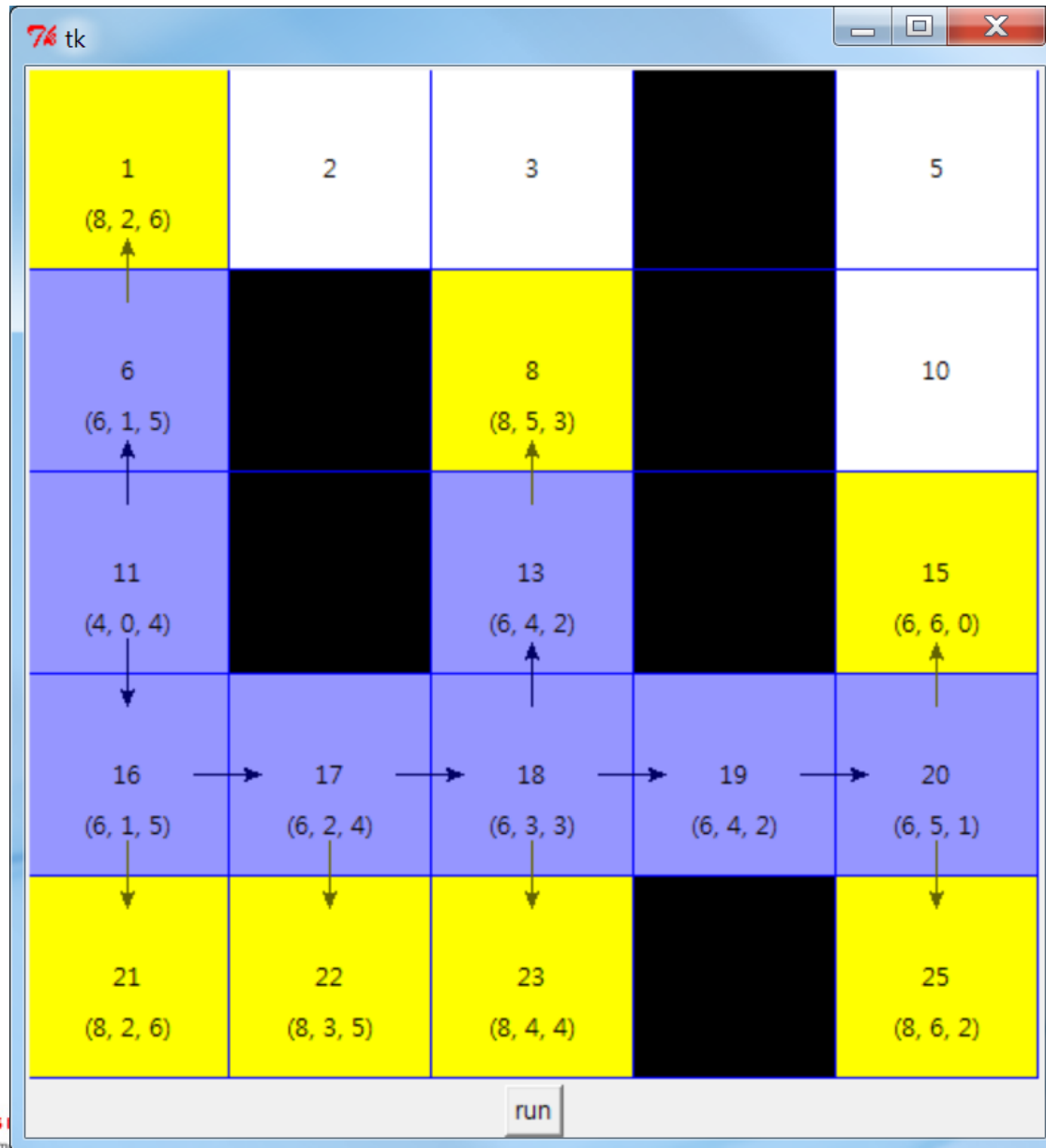
A* search



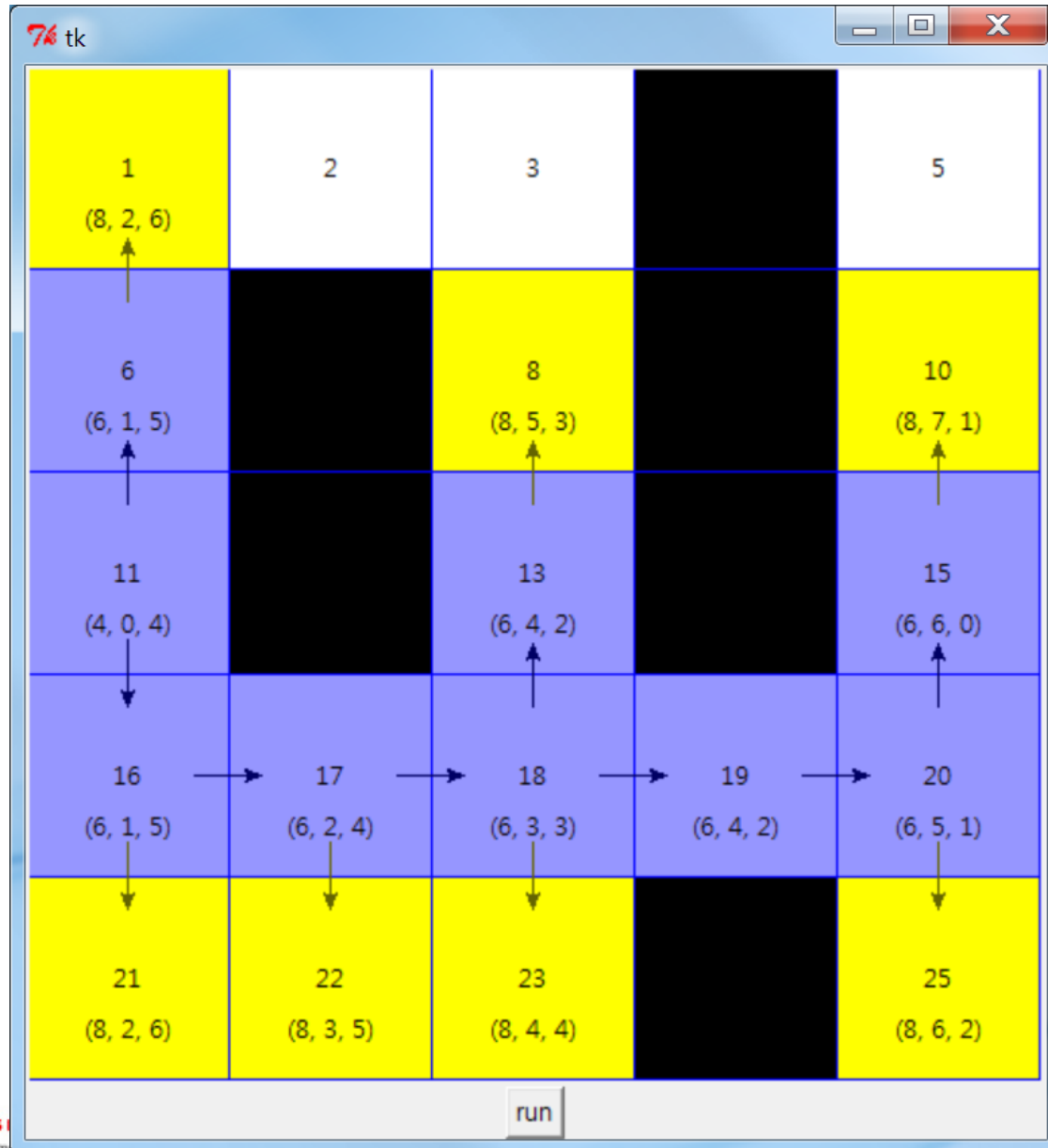
A* search



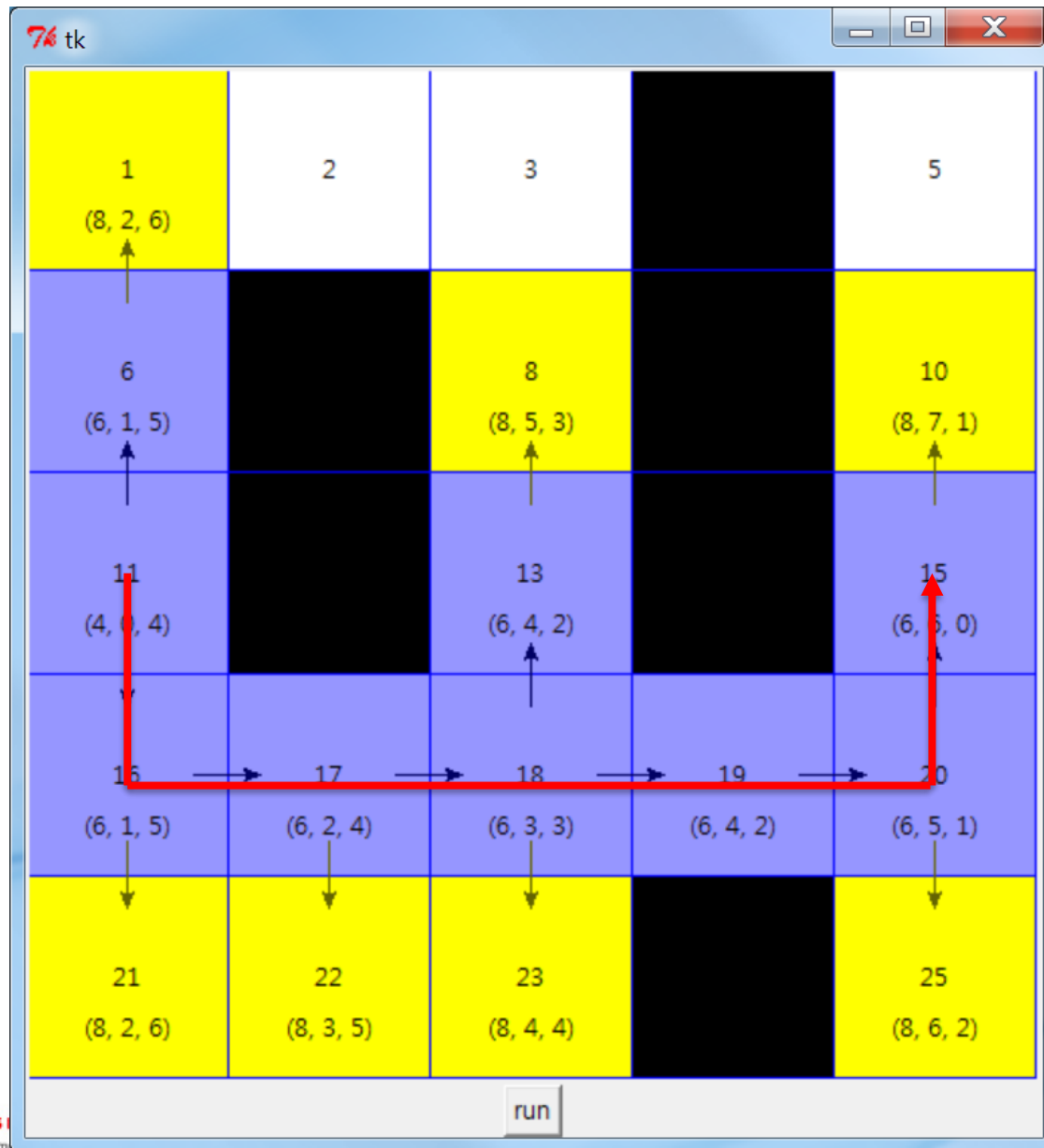
A* search



A* search



A* search



공지

- 에듀클래스 과제제출시 내용 작성하지 말 것
 - 중요한 내용이라면 소스코드 최상단 주석으로 작성
 - Ex) 별도의 설치가 필요한 라이브러리
- 외부 라이브러리 사용
 - 가급적이면 강의에서 다루었던 라이브러리를 위주로 사용
 - 부득이한 경우 pip로 쉽게 설치 가능한 라이브러리를 사용
 - + 소스코드 최상단 주석에 명시



강의 계획

주차	강의	실습
1	과목 소개 인공지능 및 python 소개 Python 프로그래밍 환경 조성 및 기본 문법	Python 개발 환경 구축 간단한 expert system 구현
2	Search (Blind Search, Heuristic Search)	DFS, BFS, A* 구현
3	Learning / Simulated Annealing	Simulated Annealing 구현
4	Genetic Algorithm	Genetic Algorithm 구현
5	보강 주간 (개천절 / 추석 연휴)	
6	Neural Network	Perceptron 구현
7	Neural Network	Single-Layer Perceptron 구현
8	Neural Network	Multi-Layer Perceptron 구현
9	Deep Neural Network	DNN 구현
10	Deep Neural Network	DNN 구현
11	DNN 심화 (혹은 Particle Swarm Optimization)	관련 내용 구현
12	DNN 심화 (혹은 Gaussian Mixture Model)	관련 내용 구현
13	DNN 심화 (혹은 Gaussian Mixture Model)	관련 내용 구현
14	DNN 심화 (혹은 Bayesian Networks)	관련 내용 구현
15	DNN 심화 (혹은 Decision Networks)	관련 내용 구현
16	기말고사	

Python
숙달을 위한
준비 기간

예비 기간
(대체 가능)

강의개요

- 강의
 - (machine) learning
 - Pattern recognition을 중심으로
- 실습
 - Simulated annealing 구현



강의

전문가 시스템(expert system)

- 간단한 인공지능 예시

- 물고기 종류 구분

- 또 다른 특징을 함께 고려

- Ex) 꼬리 길이

연어

농어

몸길이 = 60 꼬리길이 = 11

몸길이 = 68 꼬리길이 = 18

몸길이 = 80 꼬리길이 = 15

몸길이 = 75 꼬리길이 = 13

몸길이 = 84 꼬리길이 = 20

몸길이 = 99 꼬리길이 = 12

몸길이 = 102 꼬리길이 = 9

몸길이 = 110 꼬리길이 = 5

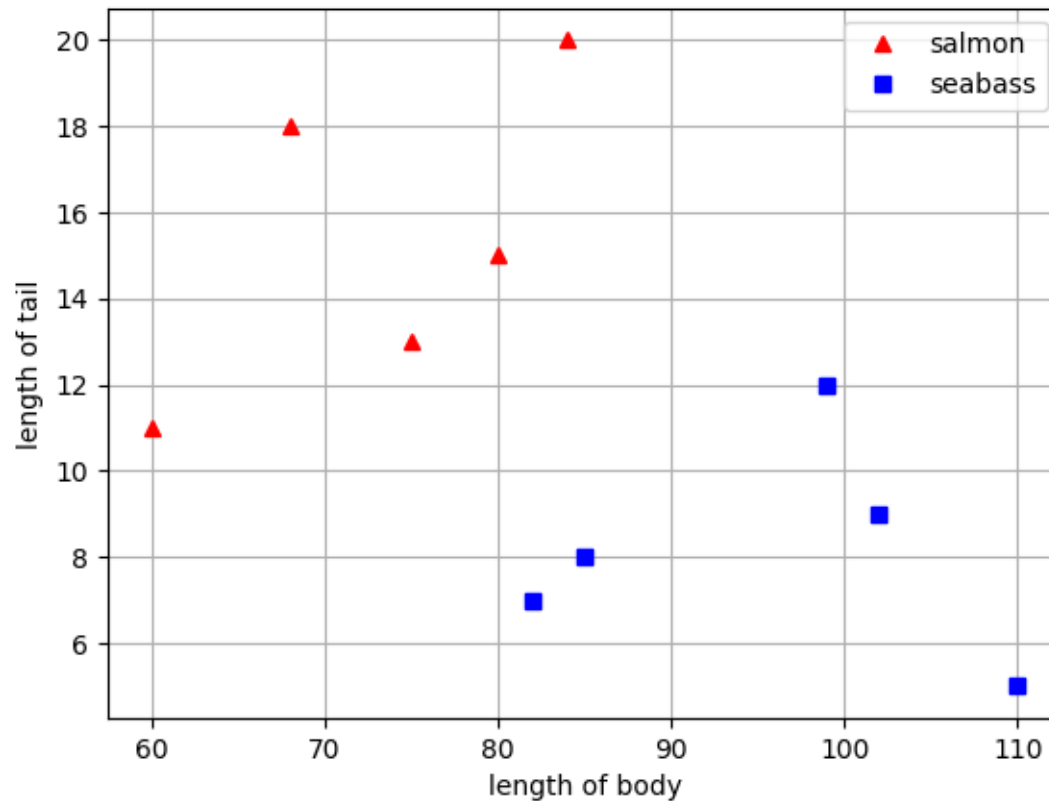
몸길이 = 85 꼬리길이 = 8

몸길이 = 82 꼬리길이 = 7

```
if 몸길이 < x and 꼬리길이 > y:  
    return '연어'  
else if 몸길이 >= x and 꼬리길이 <= y:  
    return '농어'
```

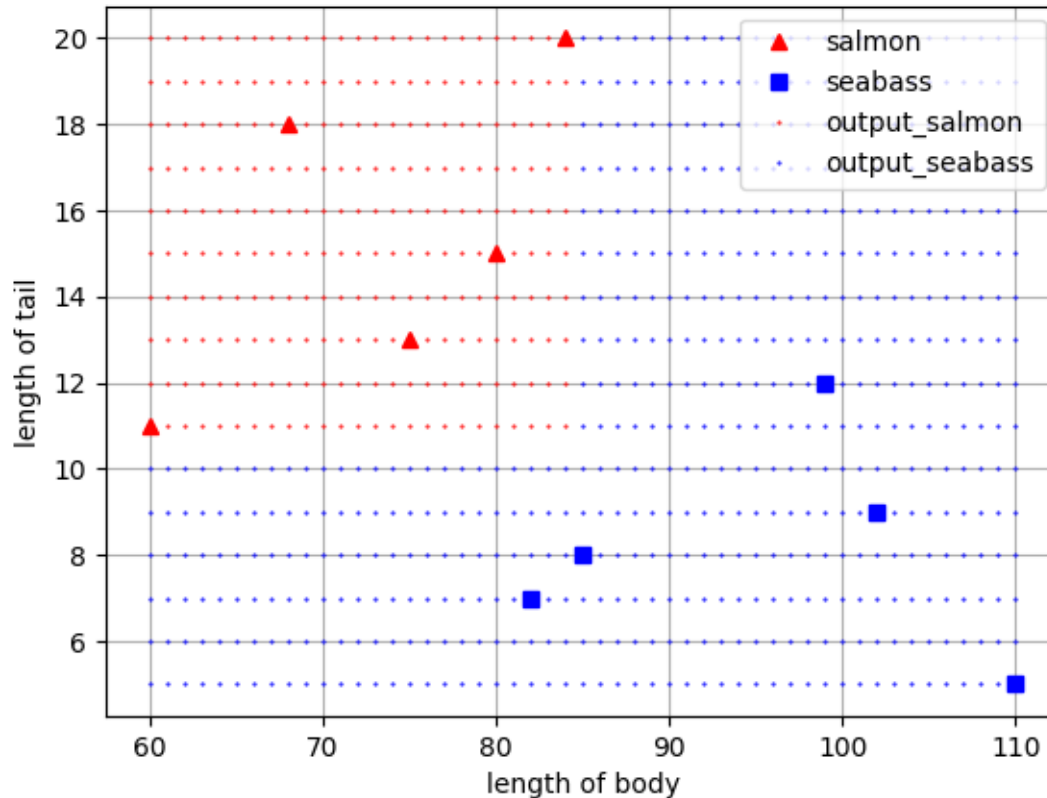


물고기 종류 구분하기(데이터 분포)



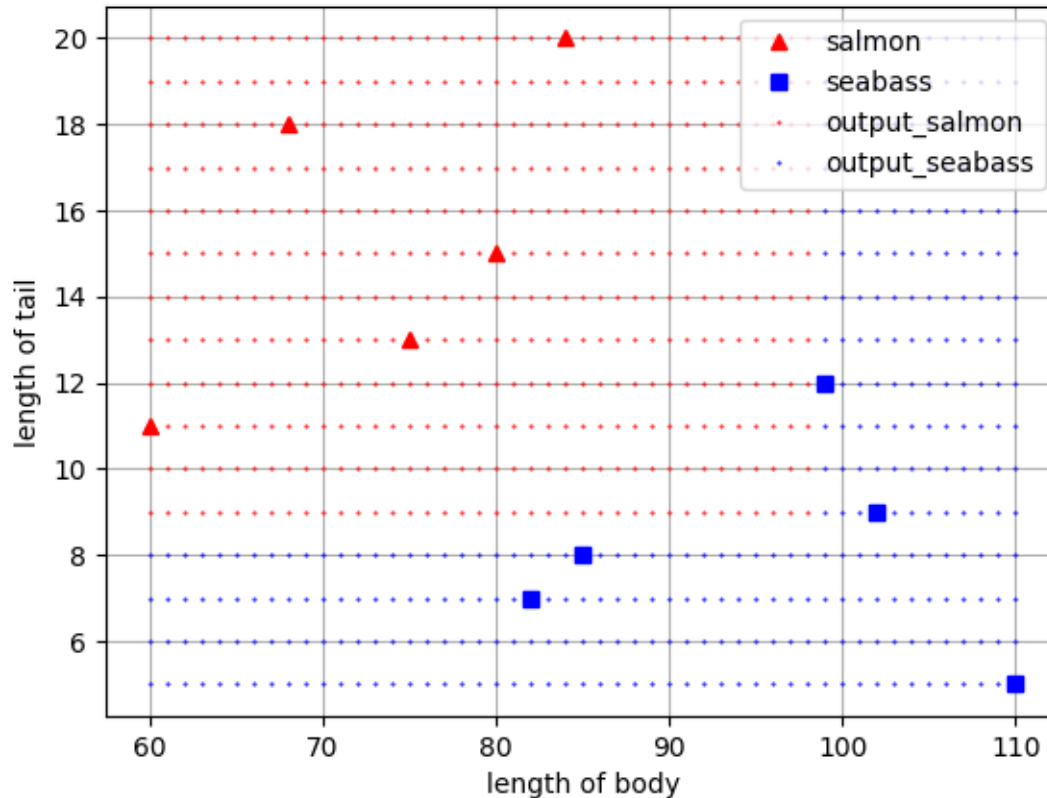
물고기 종류 구분하기(이진 분류)

```
def getClass(x, y):  
    if x < 85 and 10 < y:  
        return 'salmon'  
    return 'seabass'
```



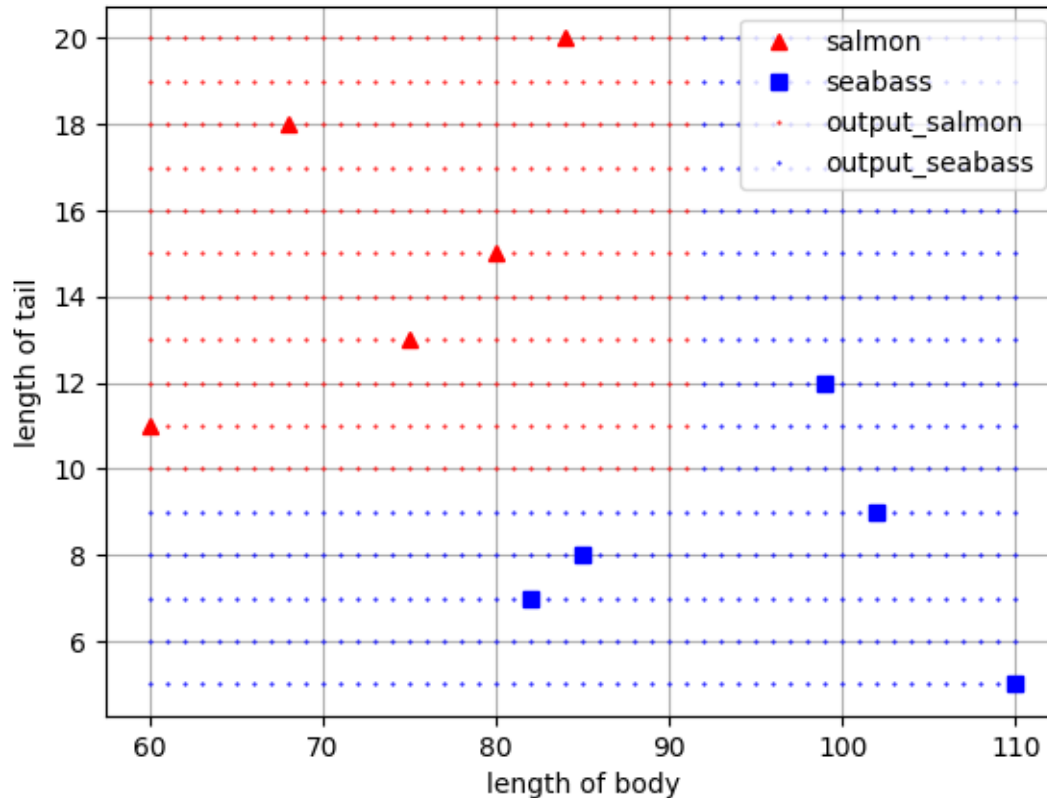
물고기 종류 구분하기(이진 분류)

```
def getClass(x, y):  
    if x < 99 and 8 < y:  
        return 'salmon'  
    return 'seabass'
```



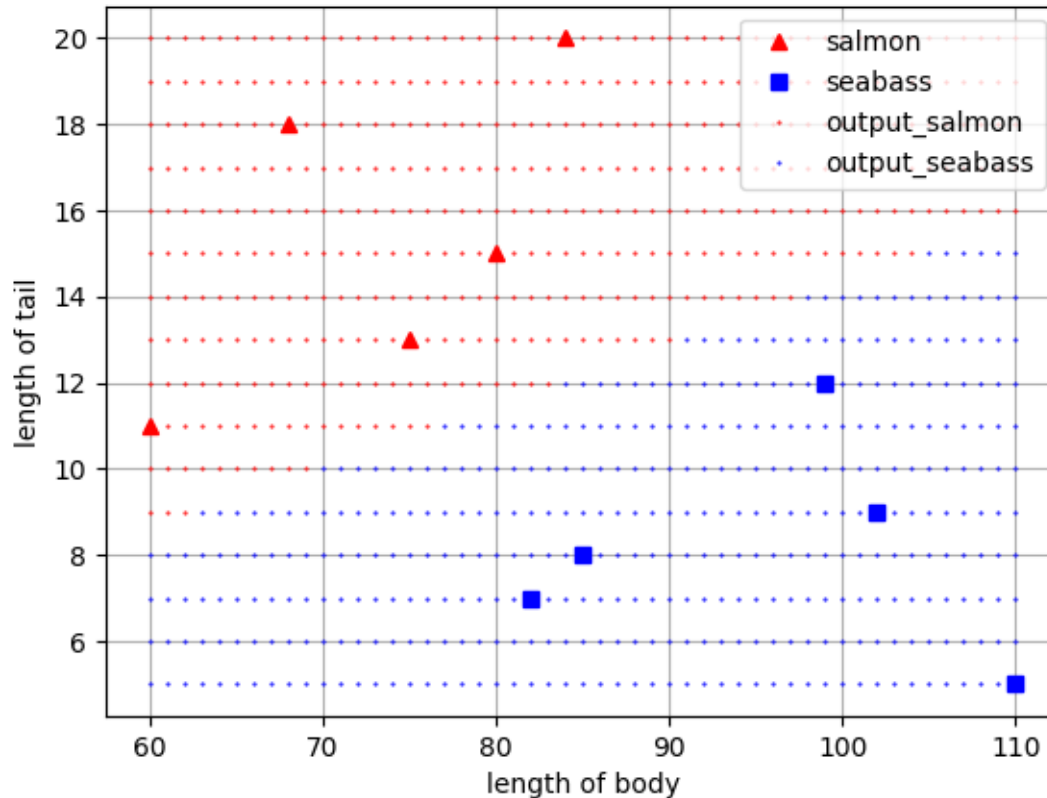
물고기 종류 구분하기(이진 분류)

```
def getClass(x, y):  
    if x < 92 and 9 < y:  
        return 'salmon'  
    return 'seabass'
```



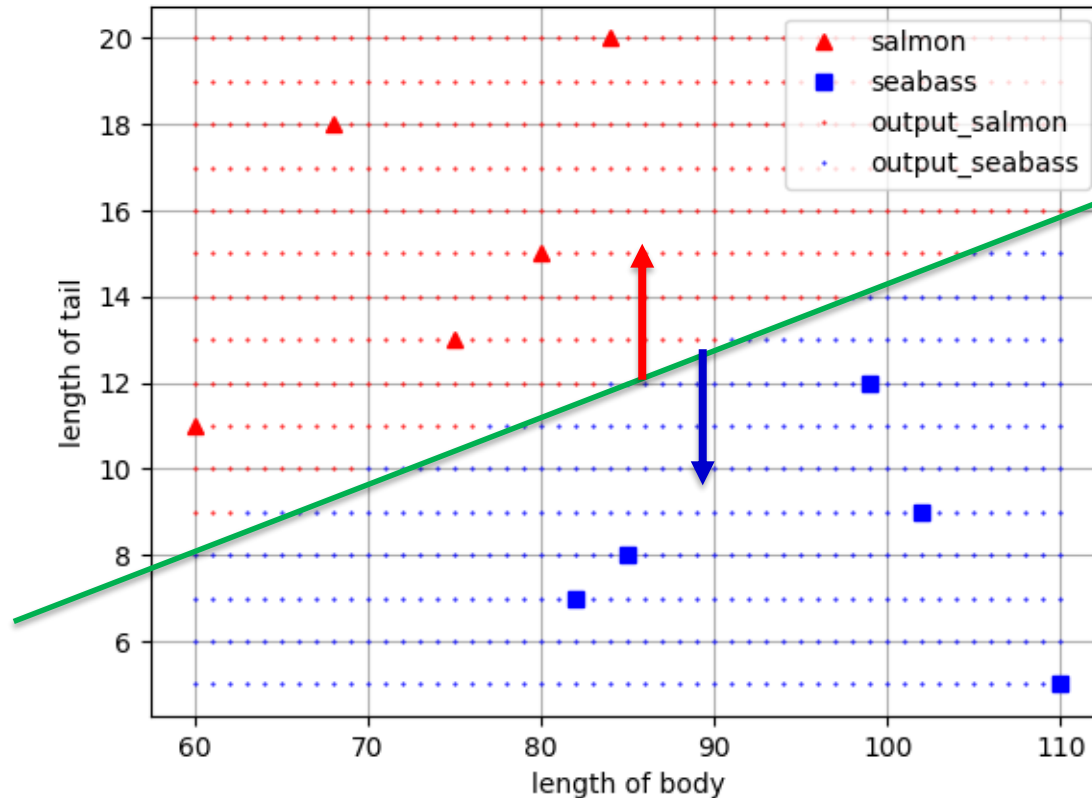
물고기 종류 구분하기(이진 분류)

```
def getClass(x, y):  
    if (x / y) < 7:  
        return 'salmon'  
    return 'seabass'
```



물고기 종류 구분하기(이진 분류)

```
def getClass(x, y):  
    if (x / y) < 7:  
        return 'salmon'  
    return 'seabass'
```



$$\frac{x}{y} = 7$$

$$x = 7y$$

$$y = \frac{1}{7}x$$

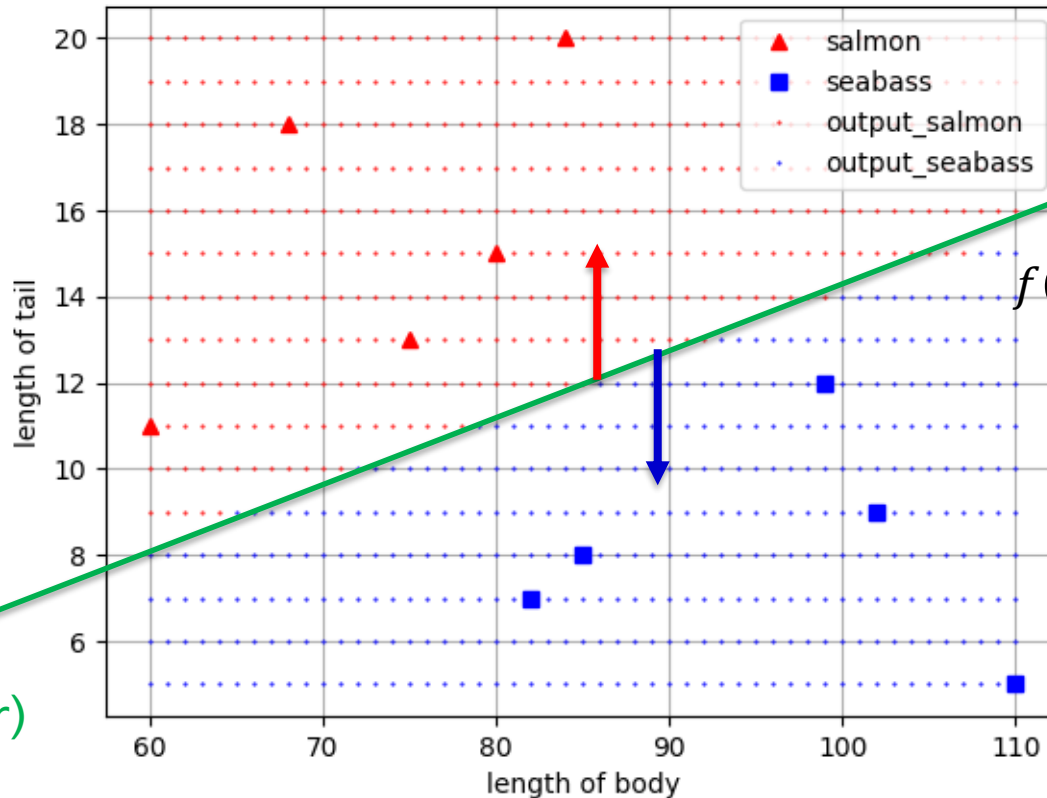
$$y = 0.14x$$

$$0.14x - y = 0$$



물고기 종류 구분하기(이진 분류)

```
def getClass(x, y):  
    if (0.14 * x - y) < 0:  
        return 'salmon'  
    return 'seabass'
```



선형 분류기
(linear classifier)

$$0.14x - y = 0$$

$$f(x) = ax + by + c = 0$$

0.14 -1 0

parameters

어떻게 정할 건가?



Learning

- 기계 학습(machine learning)

기계 학습(機械學習) 또는 머신 러닝(영어: machine learning)은 인공 지능의 한 분야로, 컴퓨터가 학습할 수 있도록 하는 알고리즘과 기술을 개발하는 분야를 말한다. 가령, 기계 학습을 통해서 수신한 이메일이 스팸인지 아닌지를 구분할 수 있도록 훈련할 수 있다.



Learning

- 어떻게 적당한 파라미터를 정할 수 있을까?

해결해야 할 문제: 연어와 농어를 완벽히 분류하는 선형 분류기의 파라미터 정하기

(pattern)
classification pattern
recognition의
한 종류

주어진 것: 연어 / 농어의 몸 길이, 꼬리 길이, 어종명 학습 데이터

특징 레이블
(feature) (label)

60	11	salmon
68	18	salmon
80	15	salmon
75	13	salmon
84	20	salmon
99	12	seabass
102	9	seabass
110	5	seabass
85	8	seabass
82	7	seabass

Learning

- 어떻게 적당한 파라미터를 컴퓨터가 정하게 할 수 있을까?

해결해야 할 문제: 연어와 농어를 완벽히 분류하는 선형 분류기의 파라미터 정하기

(pattern)
classification pattern
recognition의
한 종류

주어진 것: 연어 / 농어의 몸 길이, 꼬리 길이, 어종명 학습 데이터

특징 레이블
(feature) (label)

한 번에 정하게 할 수 있는 알고리즘을 만들 수 있는가?

60	11	salmon
68	18	salmon
80	15	salmon
75	13	salmon
84	20	salmon
99	12	seabass
102	9	seabass
110	5	seabass
85	8	seabass
82	7	seabass

Learning

- 어떻게 적당한 파라미터를 컴퓨터가 정하게 할 수 있을까?

해결해야 할 문제: 연어와 농어를 완벽히 분류하는 선형 분류기의 파라미터 정하기

(pattern)
classification pattern
recognition의
한 종류

주어진 것: 연어 / 농어의 몸 길이, 꼬리 길이, 어종명 학습 데이터

특징 레이블
(feature) (label)

한 번에 정하게 할 수 있는 알고리즘을 만들 수 있는가?



가능한 경우도 있지만 보통은 어려움

60	11	salmon
68	18	salmon
80	15	salmon
75	13	salmon
84	20	salmon
99	12	seabass
102	9	seabass
110	5	seabass
85	8	seabass
82	7	seabass

Learning

- 어떻게 적당한 파라미터를 컴퓨터가 정하게 할 수 있을까?

해결해야 할 문제: 연어와 농어를 완벽히 분류하는 선형 분류기의 파라미터 정하기

한 번에 정하기



여러 단계에 걸쳐 찾아가기

완벽하게 분류하기

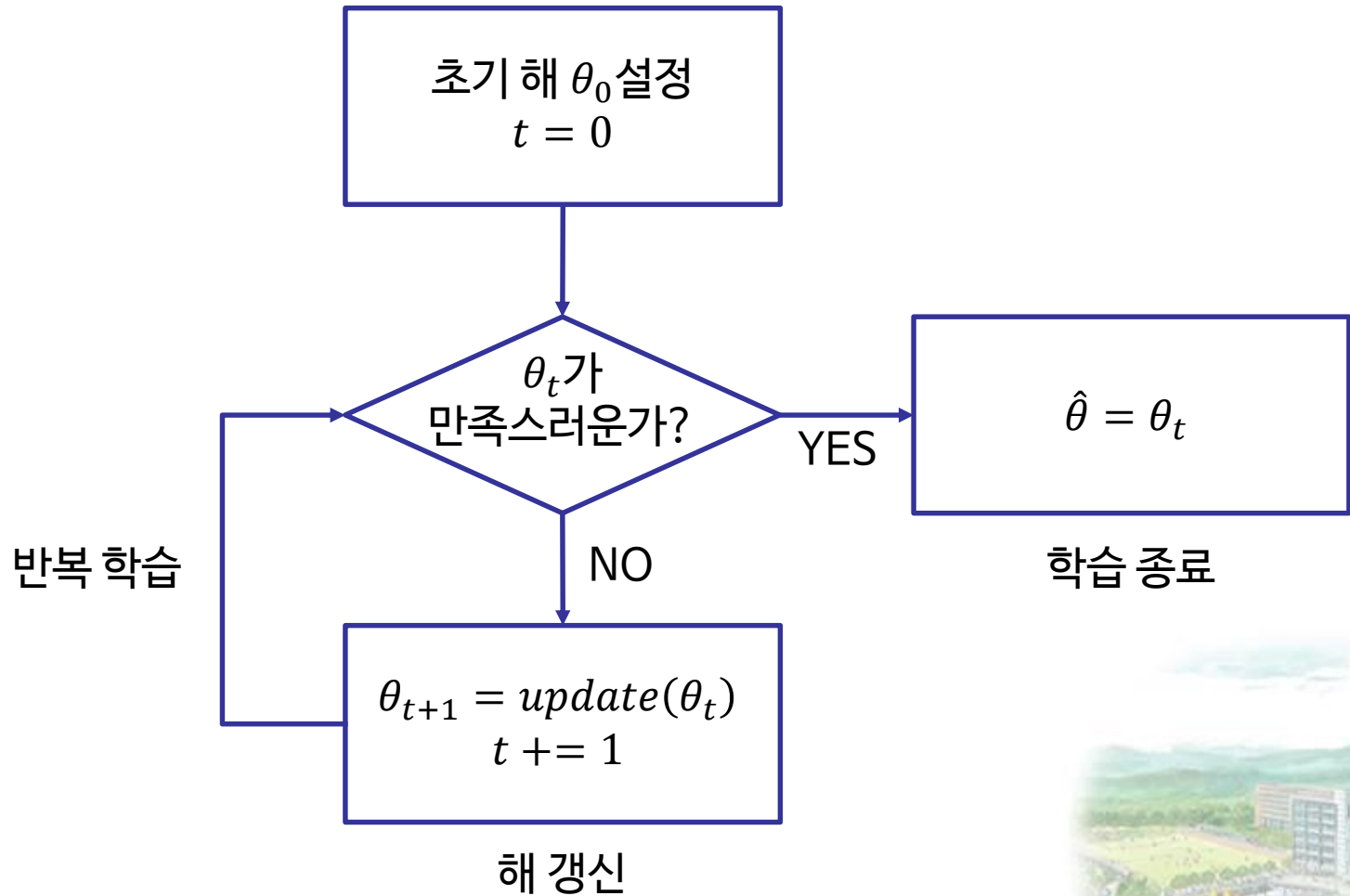


적당히 괜찮은 결과를 찾으면 멈추기



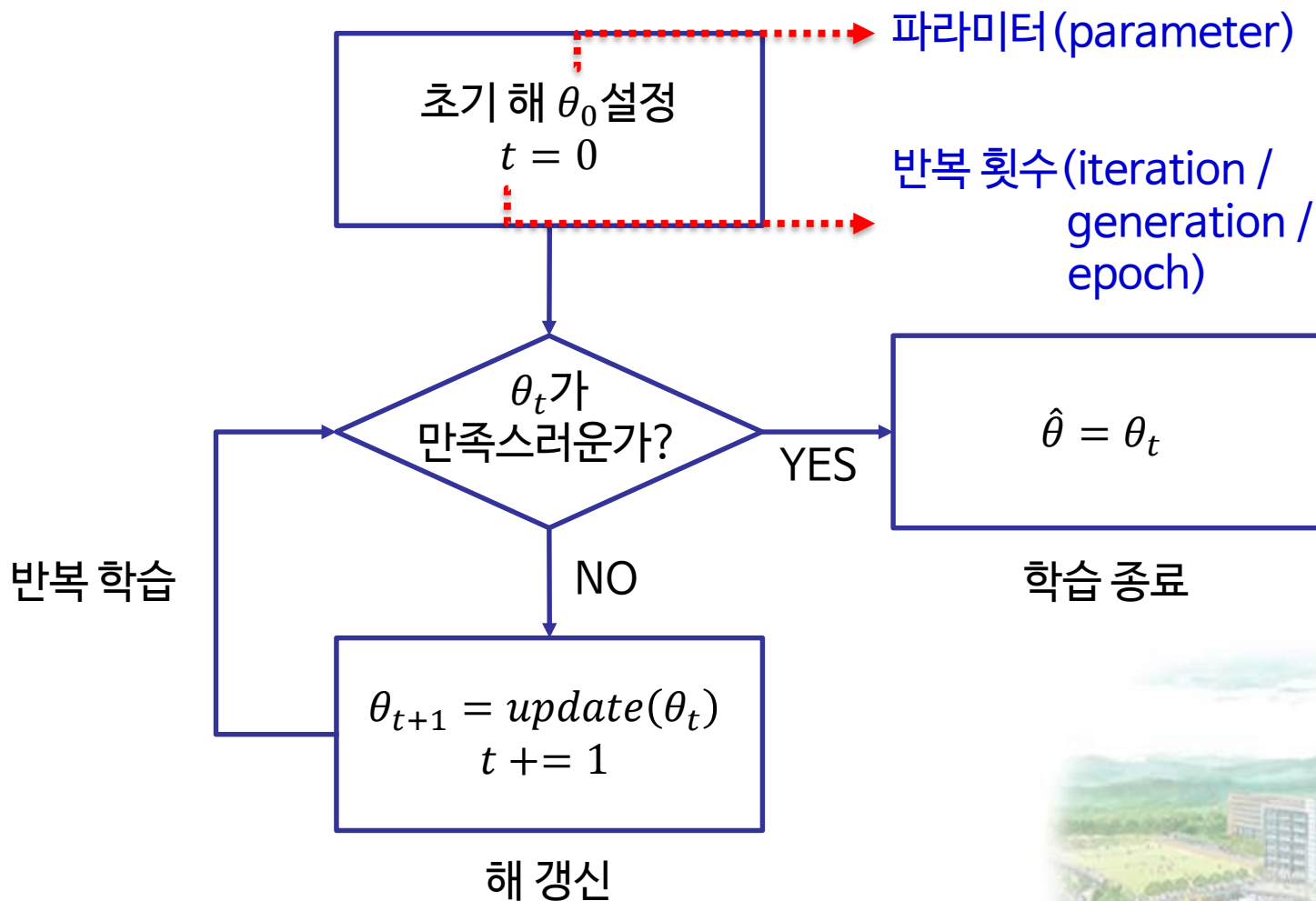
Learning

- 최적의 해를 찾아가는 반복 알고리즘



Learning

- 최적의 해를 찾아가는 반복 알고리즘



Learning

- 최적의 해를 찾아가는 반복 알고리즘

최적화(optimization)

탐색(search)

Search algorithms based on optimization

Gradient descent

Simulated annealing

Genetic algorithms

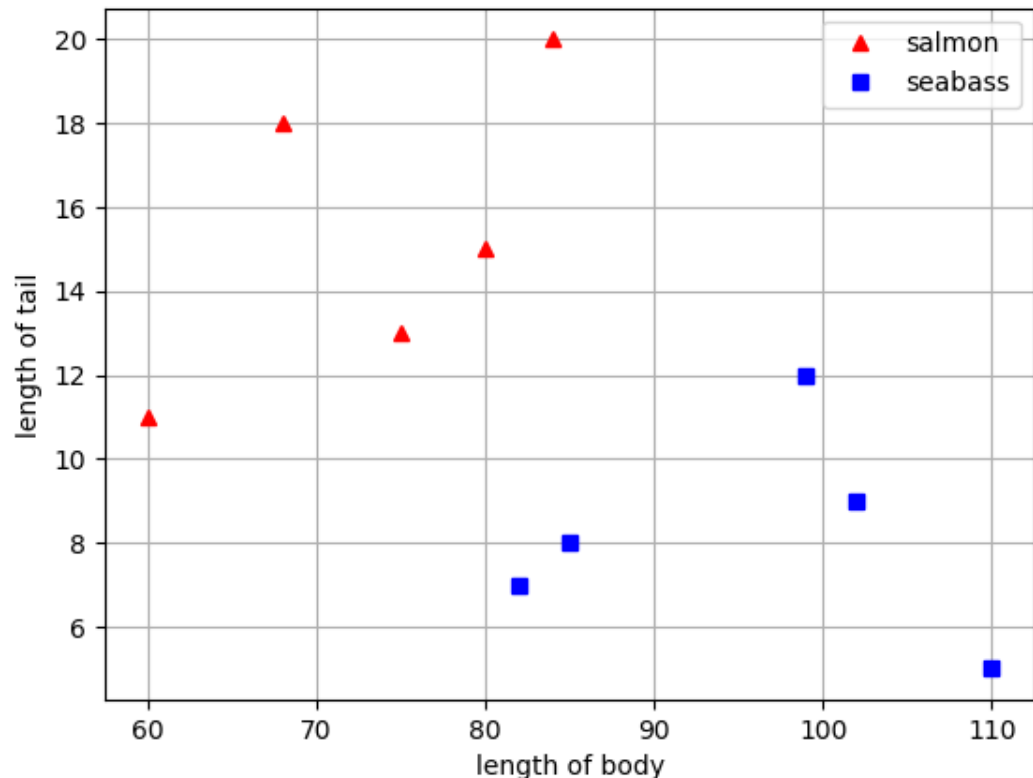
Particle swarm optimization



Learning의 종류

- 레이블의 유무에 따른 종류
 - 지도학습(supervised learning)
 - 학습 과정에서 입력 데이터에 대한 올바른 출력 (label)을 알려줌
 - Ex) classification

		label
60	11	salmon
68	18	salmon
80	15	salmon
75	13	salmon
84	20	salmon
99	12	seabass
102	9	seabass
110	5	seabass
85	8	seabass
82	7	seabass



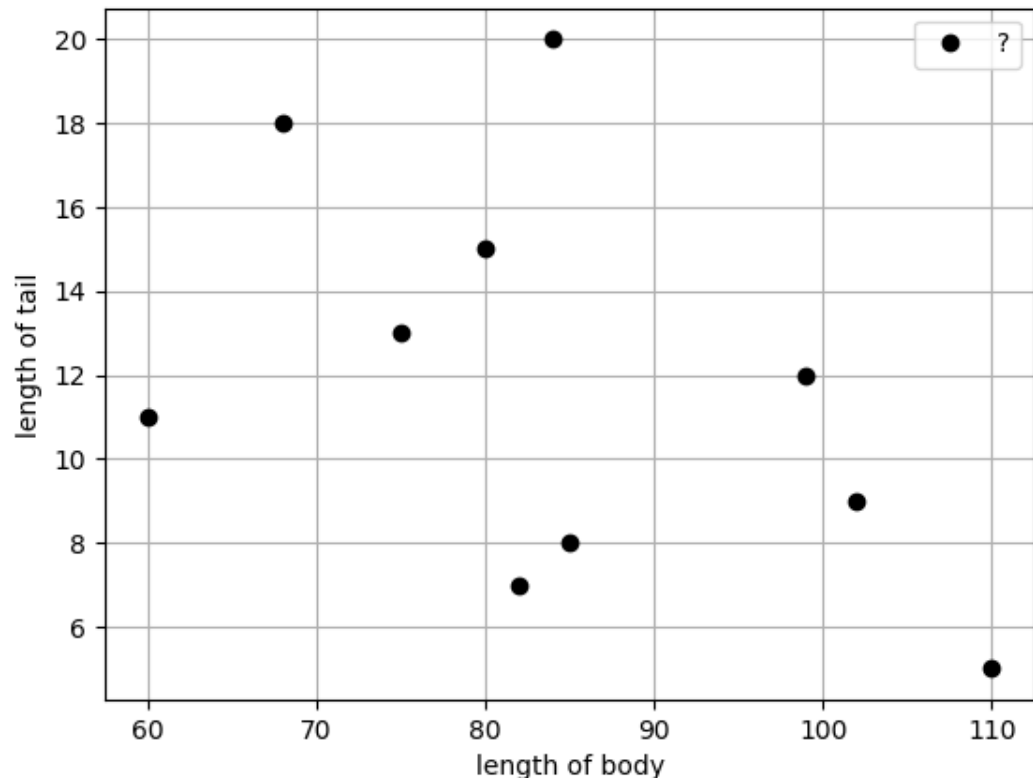
Learning의 종류

- 레이블의 유무에 따른 종류

- 비지도학습(**un**supervised learning)

- 학습 과정에서 입력 데이터에 대한 올바른 출력 (**label**)을 알려주지 **않음**
- Ex) clustering

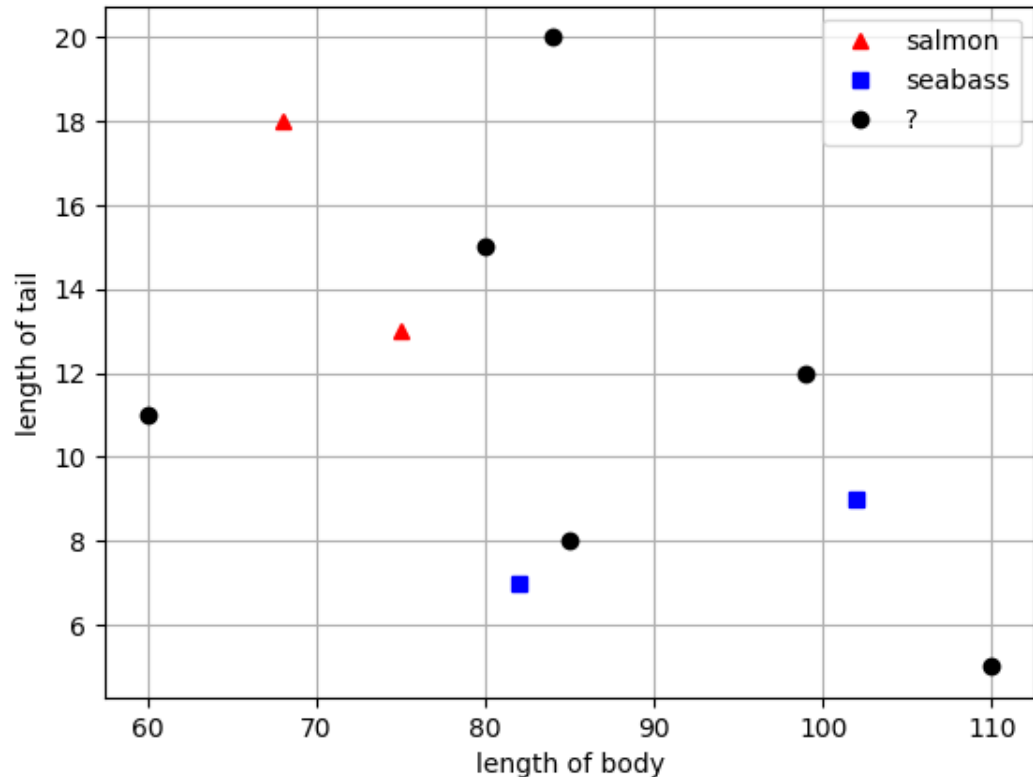
		label
60	11	
68	18	
80	15	
75	13	
84	20	
99	12	
102	9	
110	5	
85	8	
82	7	



Learning의 종류

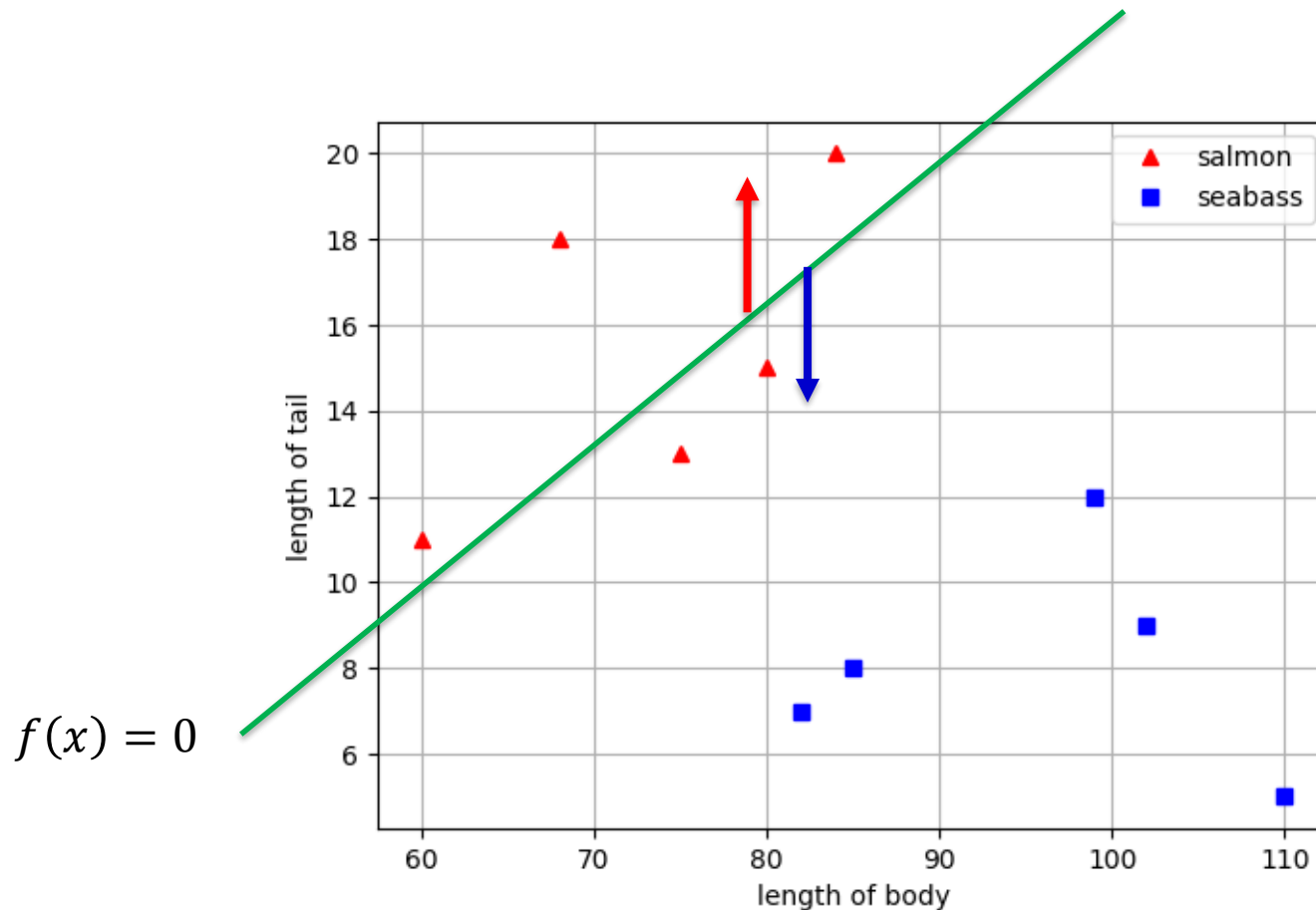
- 레이블의 유무에 따른 종류
 - 준지도학습(semi-supervised learning)
 - 학습 과정에서 입력 데이터에 대한 올바른 출력(label)을 일부만 알려줌

		label
60	11	
68	18	salmon
80	15	
75	13	salmon
84	20	
99	12	
102	9	seabass
110	5	
85	8	
82	7	seabass



성능 평가

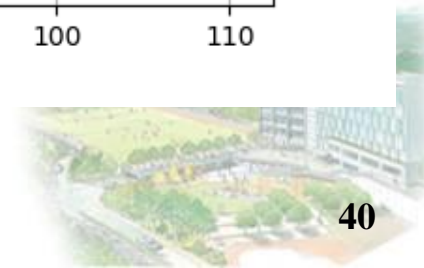
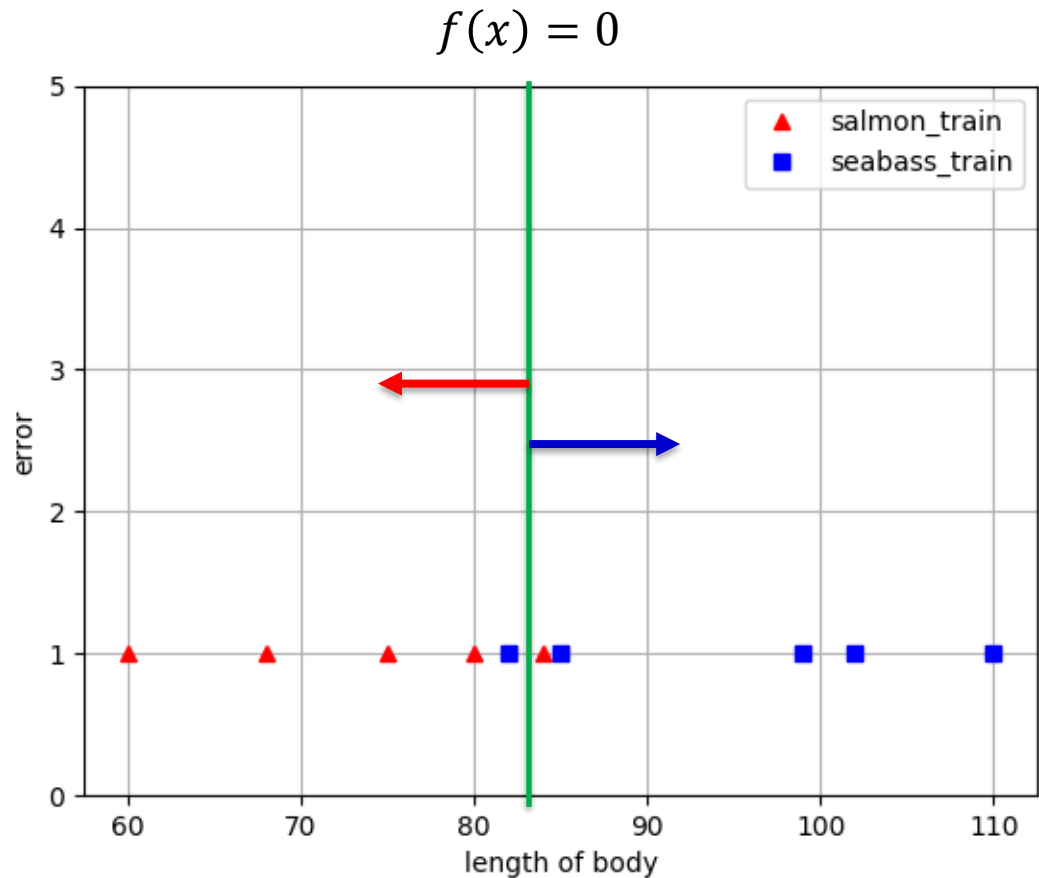
- 다음 선형 분류기의 학습 데이터에 대한 분류 오류율은 얼마인가?



성능 평가

- 다음 선형 분류기의 학습 데이터에 대한 분류 오류는 얼마인가?

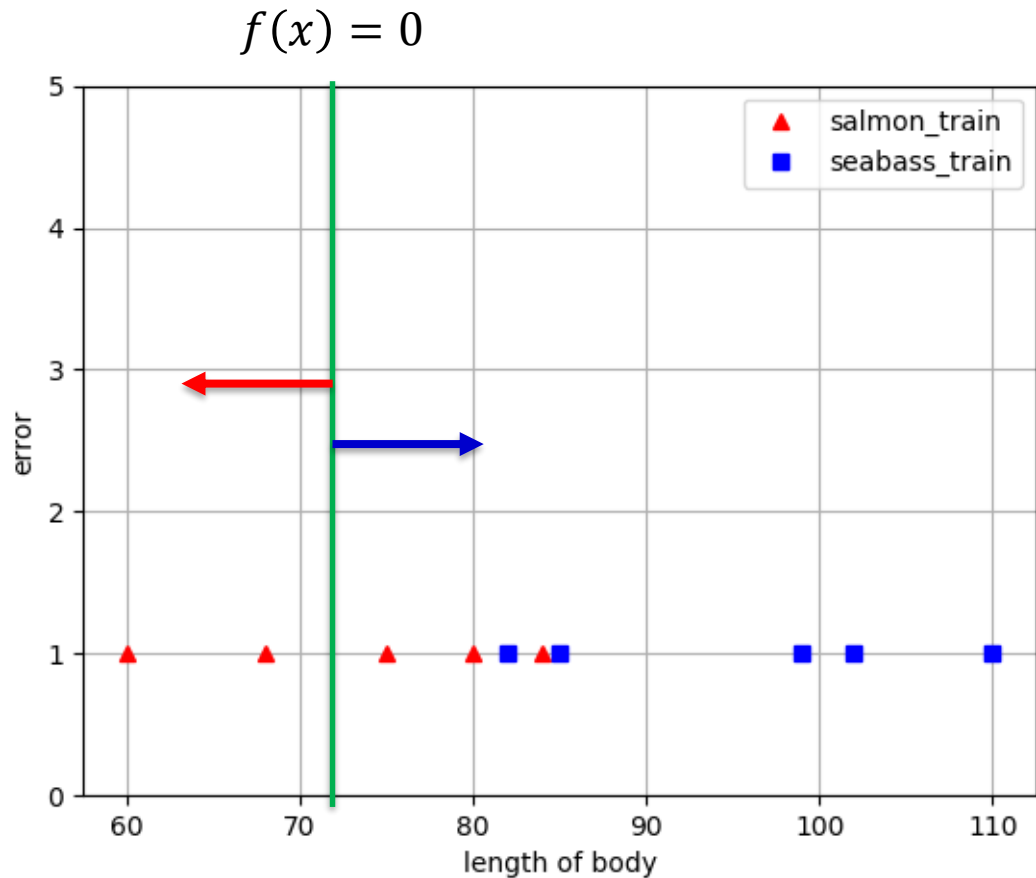
```
def getClass(x, param):  
    if x < param:  
        return 'salmon'  
    return 'seabass'
```



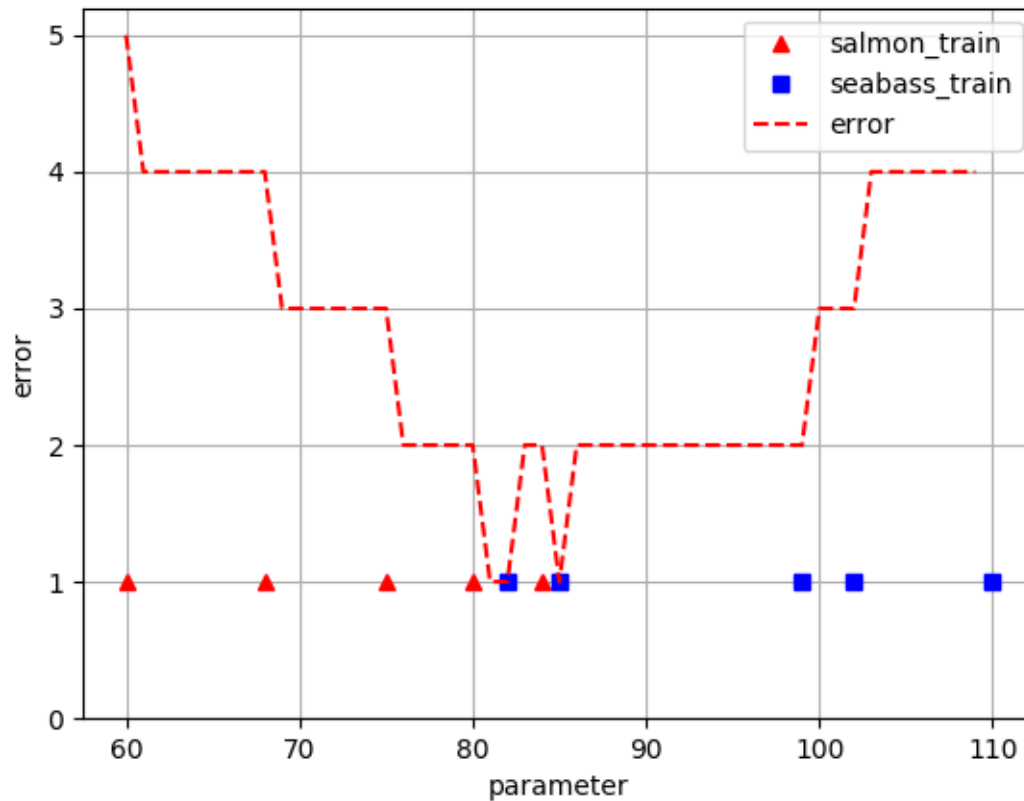
성능 평가

- 다음 선형 분류기의 학습 데이터에 대한 분류 오류는 얼마인가?

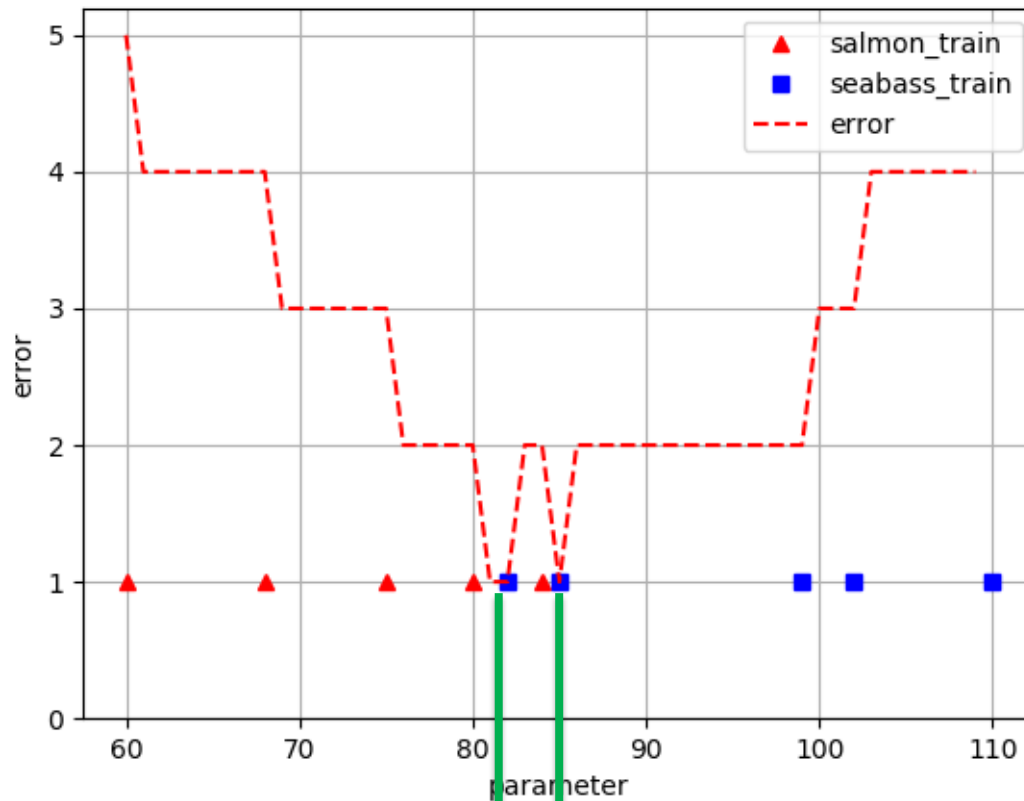
```
def getClass(x, param):  
    if x < param:  
        return 'salmon'  
    return 'seabass'
```



성능 평가



성능 평가

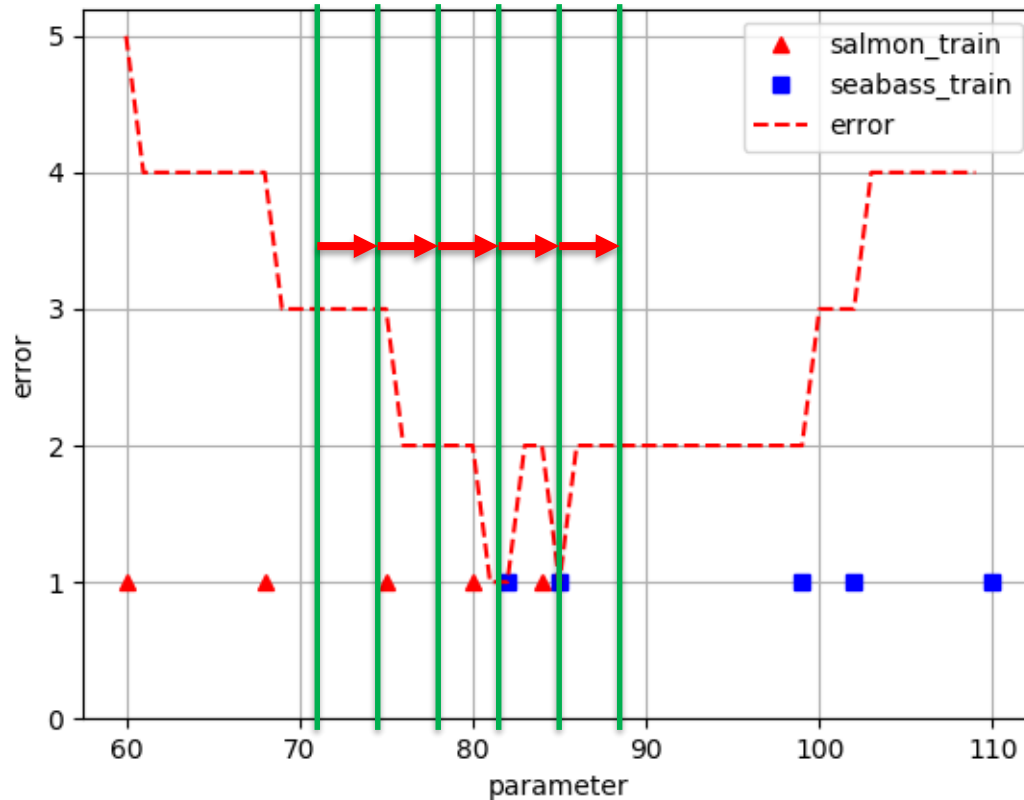


최적 파라미터

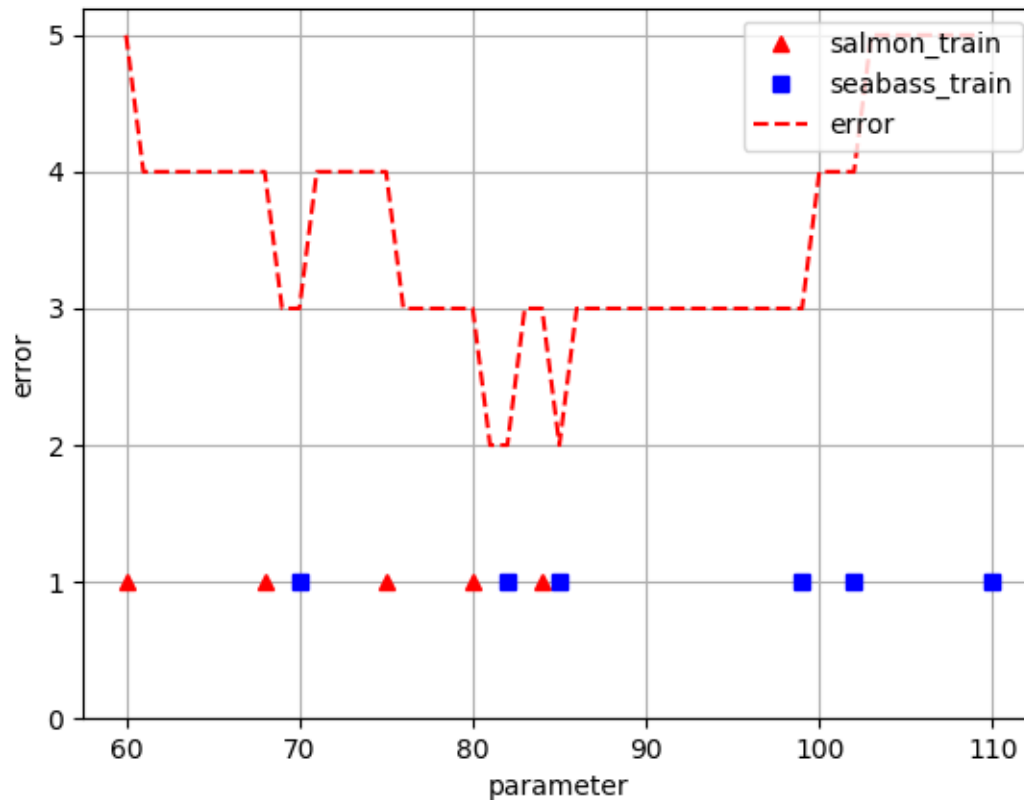


간단한 학습 전략

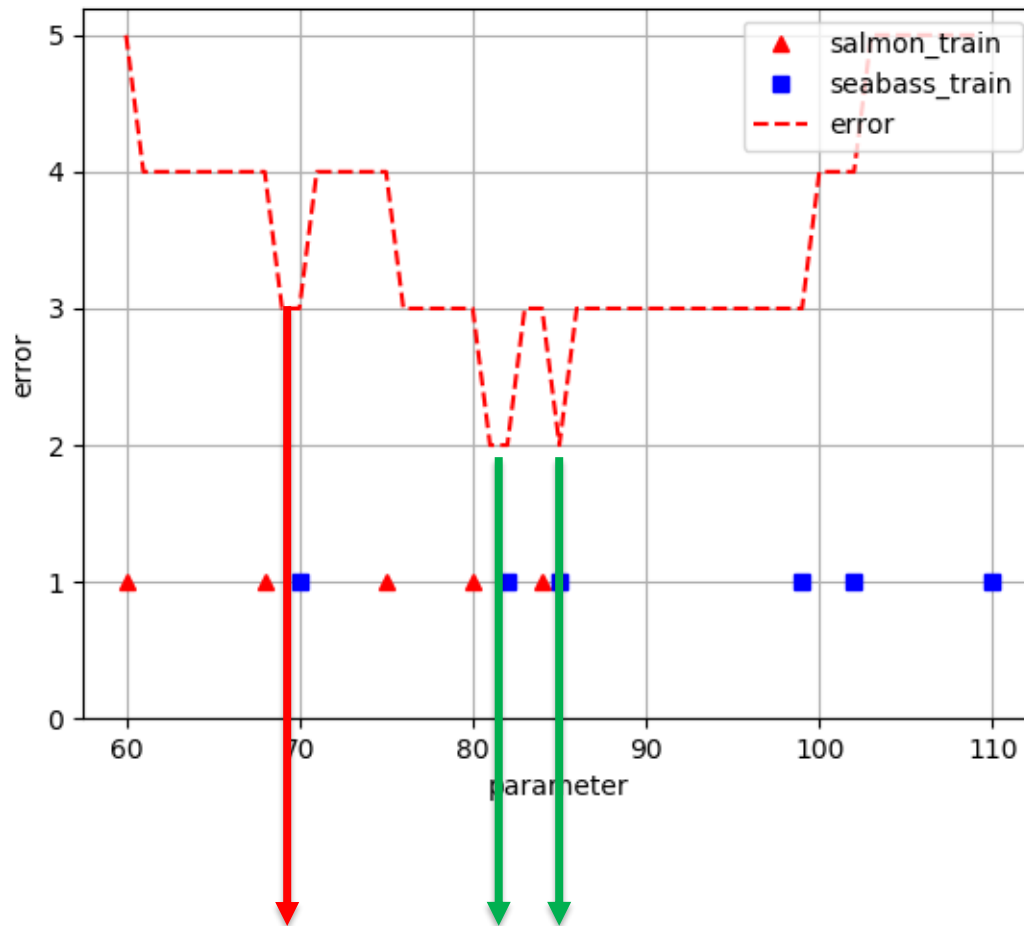
1. 랜덤 파라미터로 초기화
2. 현재 파라미터의 오류 계산
3. 파라미터를 약간 키우거나 줄임
4. 오류가 줄었거나 동일하면 2로
5. 오류가 줄지 않는다면 최소 오류 파라미터를 선택하고 학습 종료



Global / local minimum



Global / local minimum



local minimum

Global minima



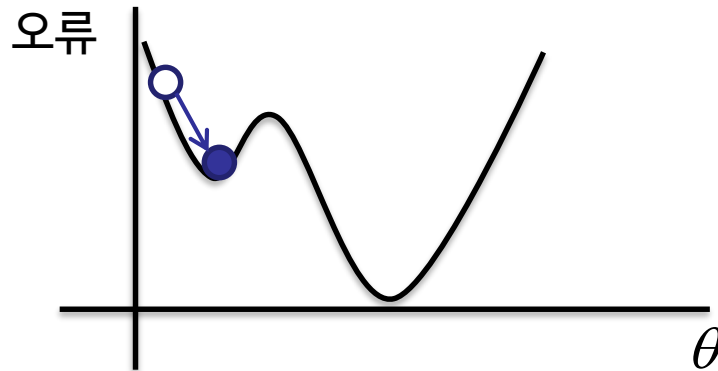
Global / local minimum

- 전역 최소점 (global minimum / optimum)
 - 전체 탐색 공간 중 **가장 오류가 적은 지점**
- 지역 최소점 (local minimum / optimum)
 - 인접한 탐색 영역에 비해 **상대적으로 오류가 적은 지점**

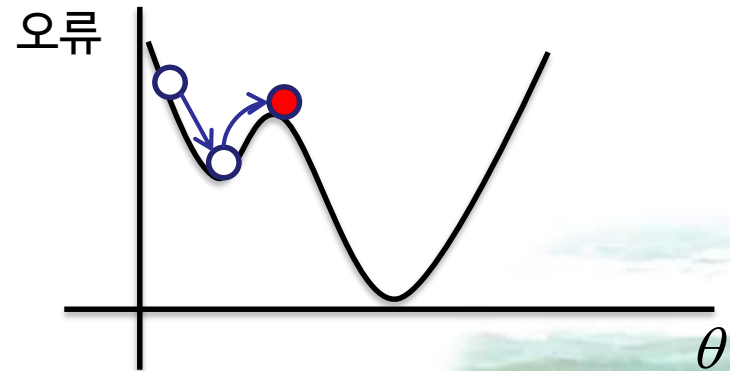


SA (simulated annealing)

- 오류가 감소하는 방향으로만 탐색을 수행하는 경우
 - Ex) gradient descent
 - Local minimum에 빠질 수 있음
- Simulated annealing의 아이디어
 - 조건에 따라 오류가 증가하는 방향으로도 탐색을 수행
 - → local minimum에서 빠지더라도 벗어날 가능성이 있음



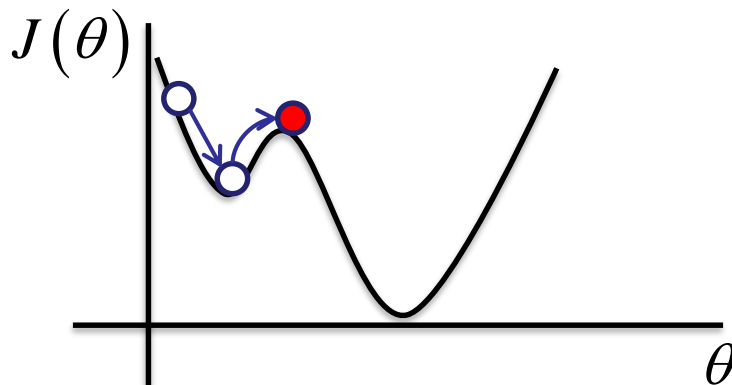
Gradient descent method



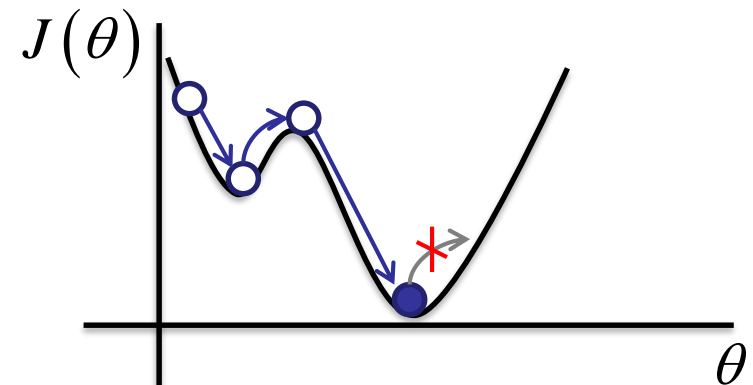
Simulated annealing

SA (simulated annealing)

- 온도가 높은 환경에서는 분자 운동이 활발하고 온도가 낮아질 수록 점차 침체됨
 - -> 초기에 온도가 높을 때에는 해가 상승하는 경우가 많지만 점차 온도를 낮추어가며 global minimum을 탐색



높은 온도



낮은 온도

Global minimum을 찾을 수 있을 만큼
충분한 시간 동안 탐색하도록
서서히 온도를 낮춤

SA (simulated annealing)

- 알고리즘

온도 T 초기화
현재 위치 θ 초기화
현재 위치의 오류 E 계산

while $T \neq 0$:

 탐색 지점 θ_{new} = 현재 위치에서 인접한 곳 중 랜덤 선택

 탐색 지점의 오류 E_{new} 계산

 현재 위치와의 오류 차이 $\Delta E = E_{new} - E$

 if $\Delta E \leq 0$:

 위치 이동 $\theta = \theta_{new}$

 else:

 0~1 사이의 랜덤값 r 생성

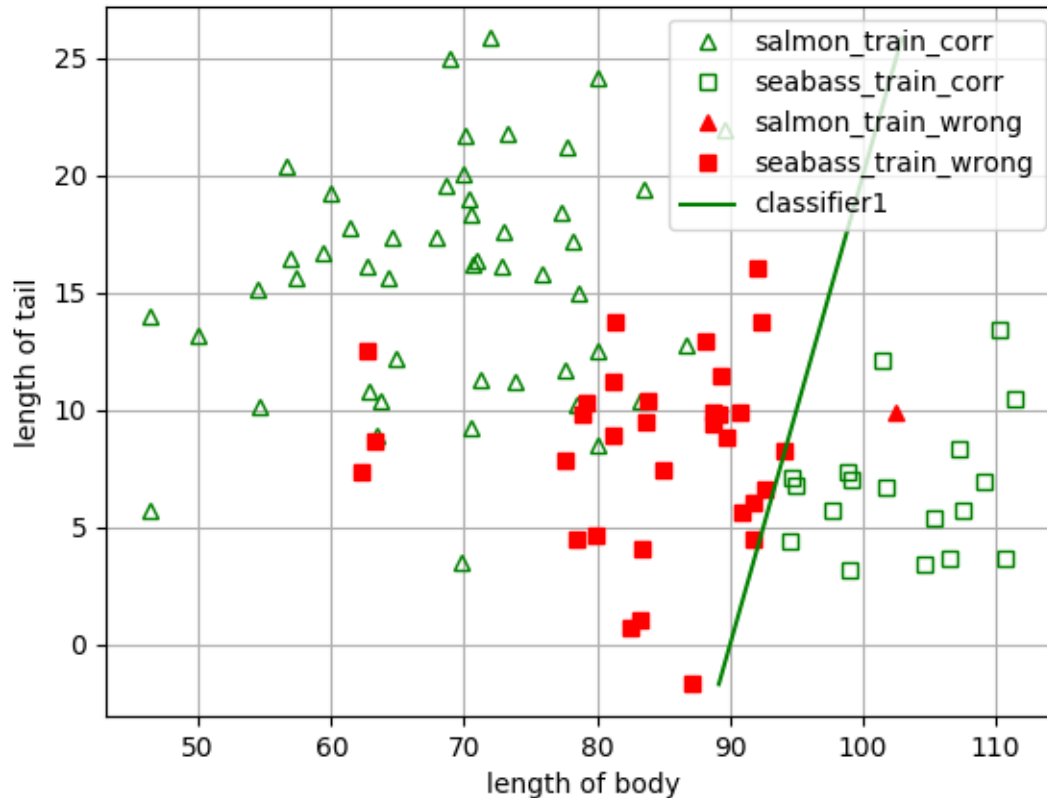
 if $r < \exp(-\Delta E/T)$:

 위치 이동 $\theta = \theta_{new}$

$T = \alpha T$ (일반적으로 $\alpha = 0.8 \sim 0.99$)

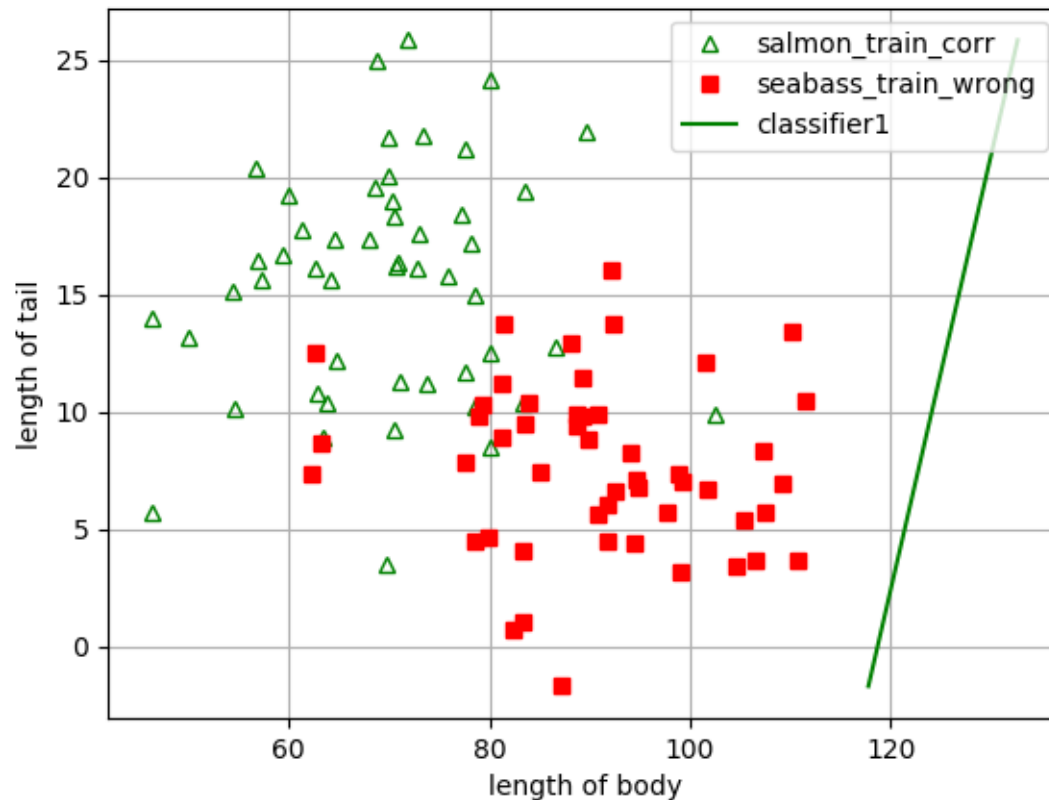
SA (simulated annealing)

학습 0회째, 온도 $T=100.0$, 오류율=33.0%



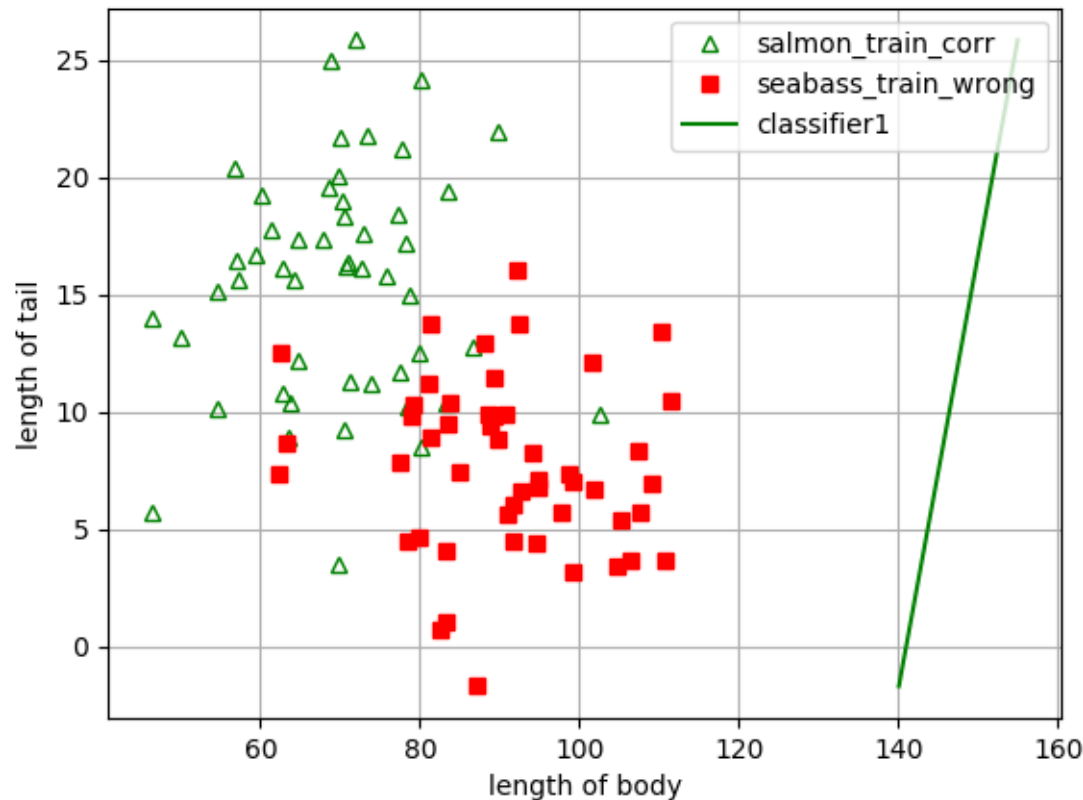
SA (simulated annealing)

학습 100회째, 온도 $T=36.6$, 오류율=50.0%



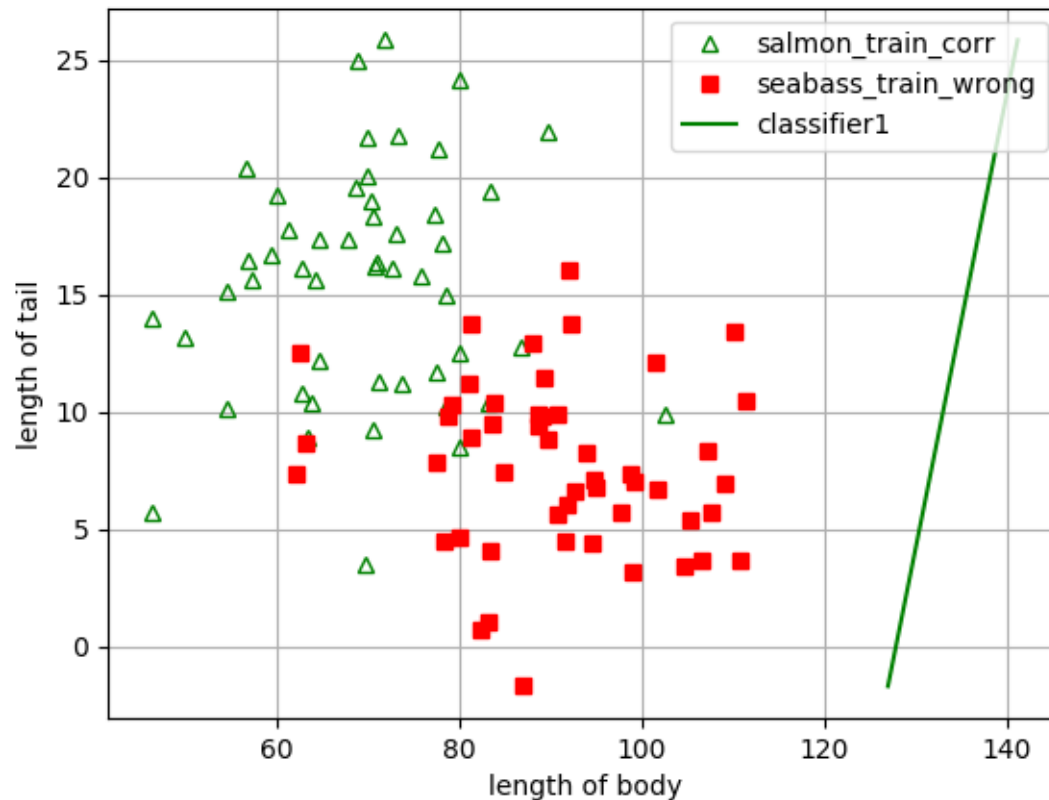
SA (simulated annealing)

학습 200회째, 온도 $T=13.4$, 오류율=50.0%



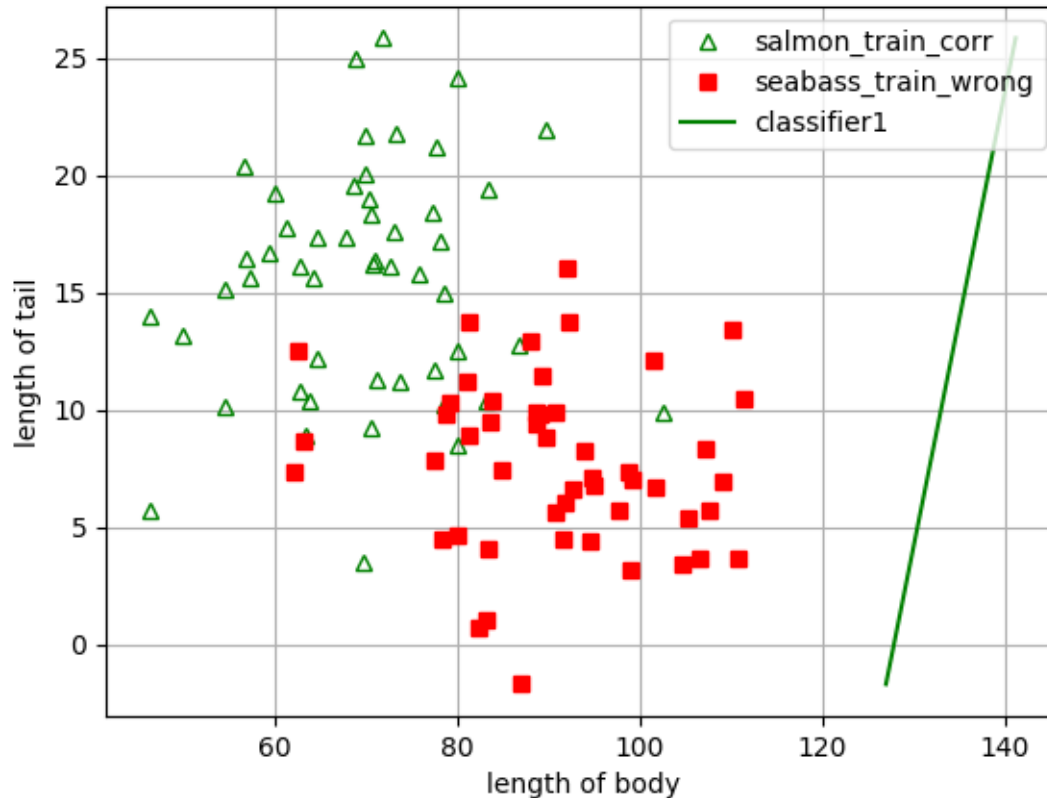
SA (simulated annealing)

학습 300회째, 온도 $T=4.9$, 오류율=50.0%



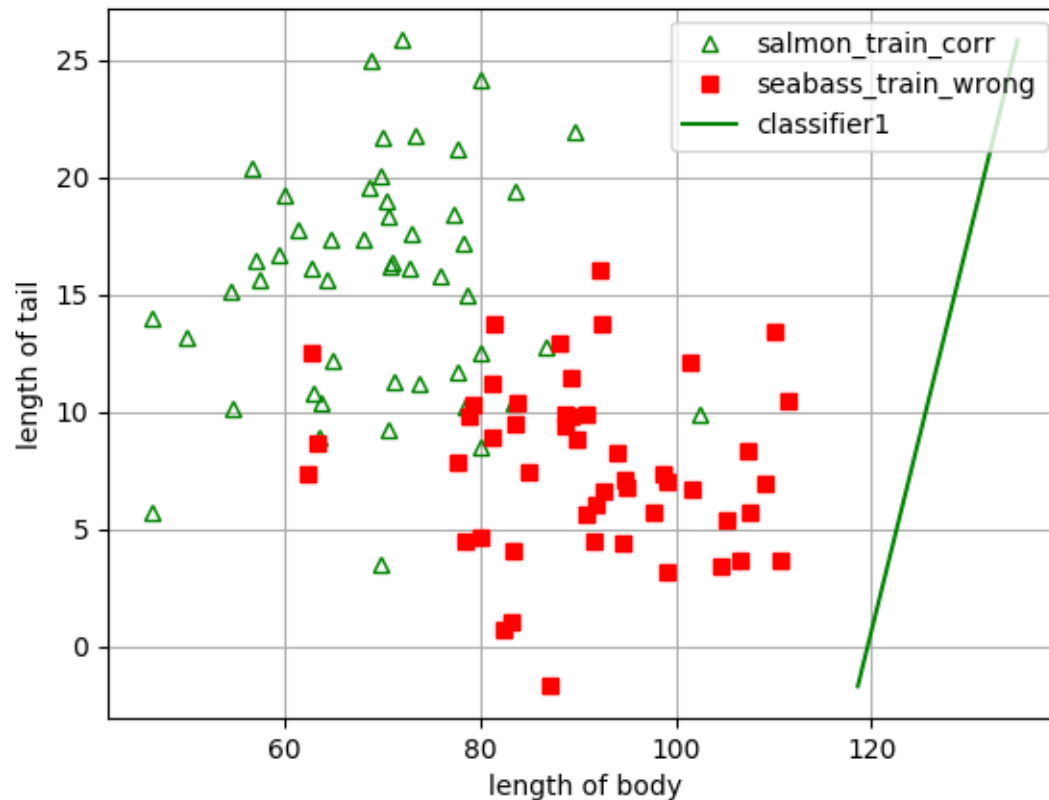
SA (simulated annealing)

학습 400회째, 온도 $T=1.8$, 오류율=50.0%



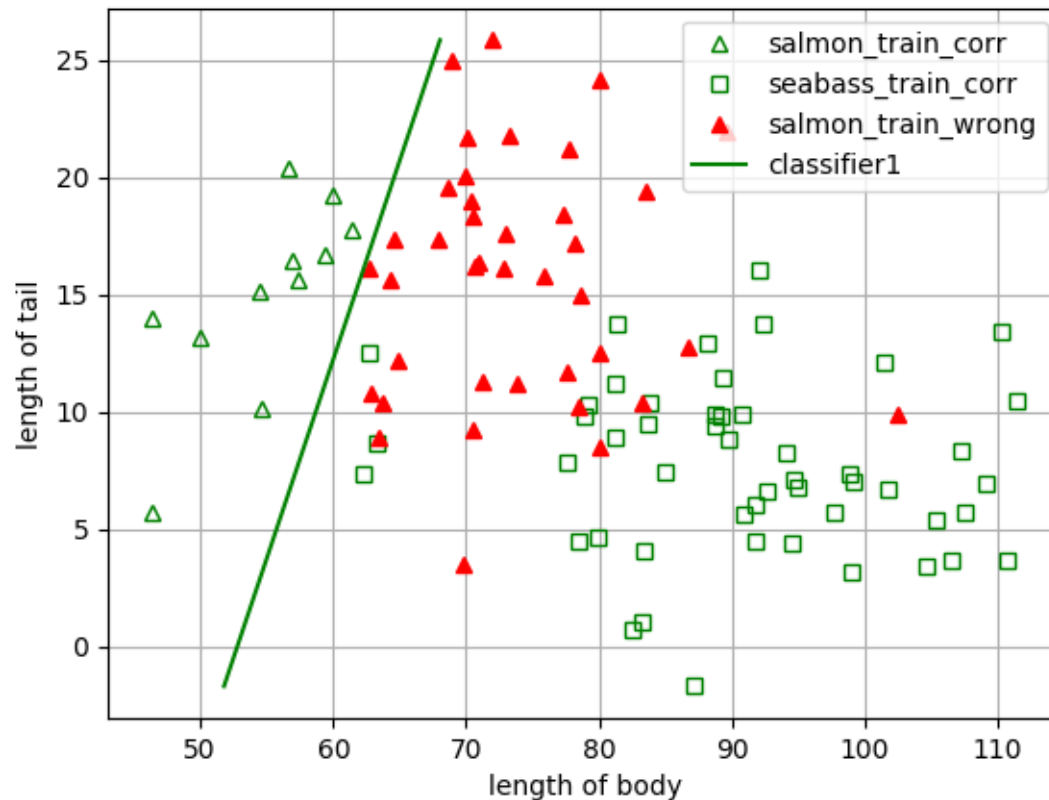
SA (simulated annealing)

학습 500회째, 온도 $T=0.7$, 오류율=50.0%



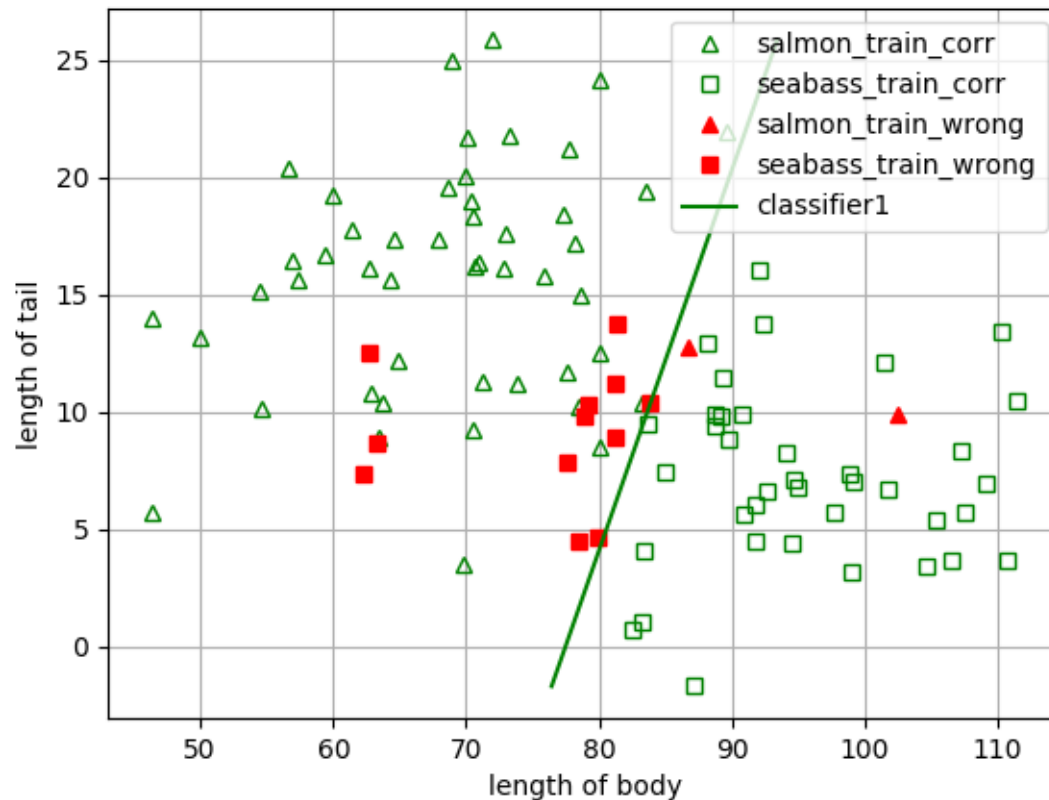
SA (simulated annealing)

학습 600회째, 온도 $T=0.2$, 오류율=39.0%



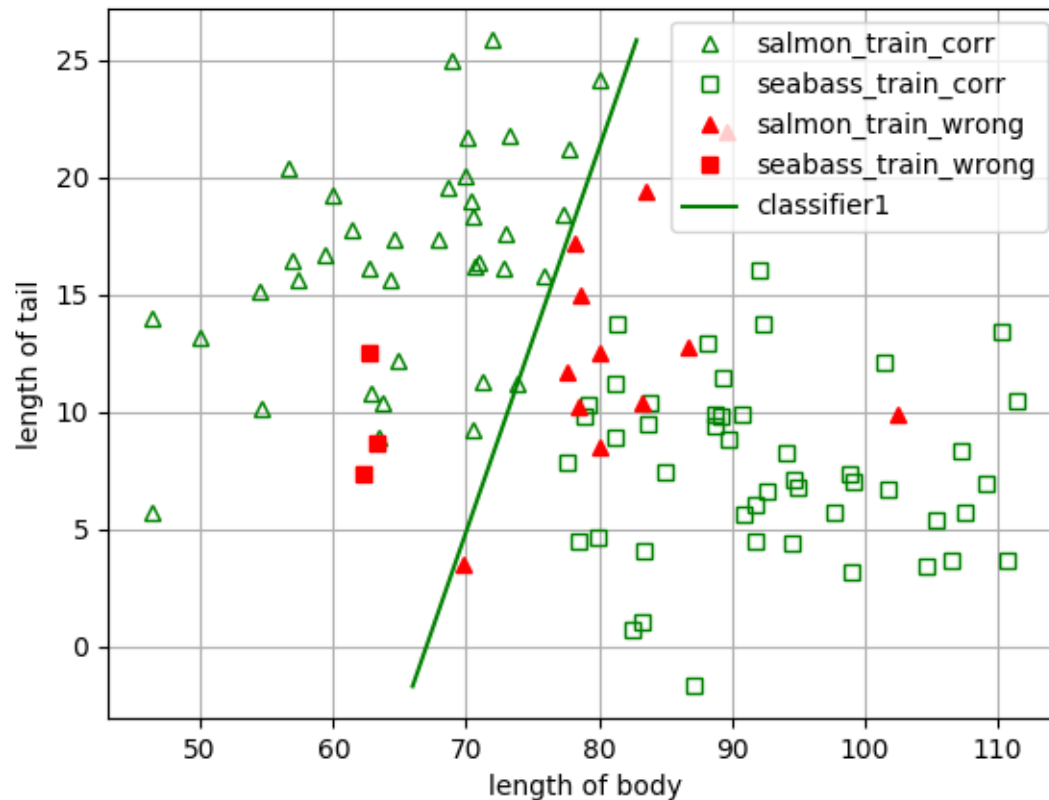
SA (simulated annealing)

학습 700회째, 온도 $T=0.1$, 오류율=14.0%



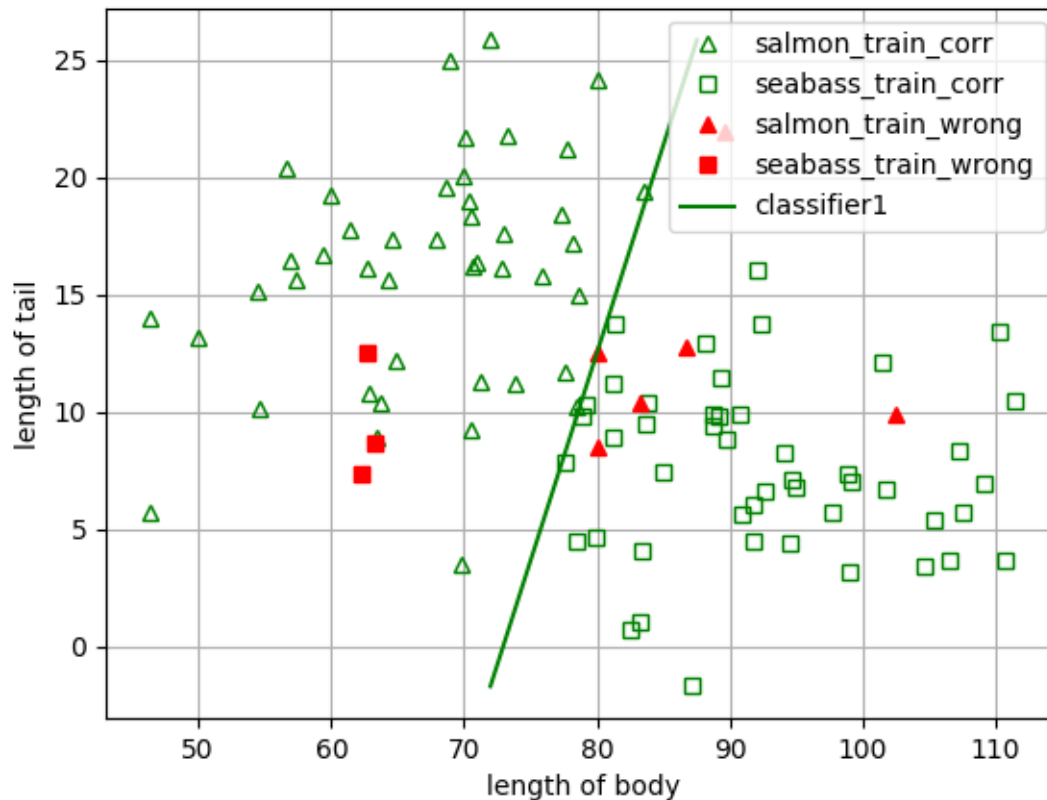
SA (simulated annealing)

학습 800회째, 온도 $T=0.0$, 오류율=15.0%



SA (simulated annealing)

학습 900회째, 온도 $T=0.0$, 오류율=9.0%



SA (simulated annealing)

- 논의



QnA

실습

전문가 시스템(expert system)

- 간단한 인공지능 예시

- 물고기 종류 구분

- 또 다른 특징을 함께 고려

- Ex) 꼬리 길이

연어

농어

몸길이 = 60 꼬리길이 = 11

몸길이 = 68 꼬리길이 = 18

몸길이 = 80 꼬리길이 = 15

몸길이 = 75 꼬리길이 = 13

몸길이 = 84 꼬리길이 = 20

몸길이 = 99 꼬리길이 = 12

몸길이 = 102 꼬리길이 = 9

몸길이 = 110 꼬리길이 = 5

몸길이 = 85 꼬리길이 = 8

몸길이 = 82 꼬리길이 = 7

```
if 몸길이 < x and 꼬리길이 > y:  
    return '연어'  
else if 몸길이 >= x and 꼬리길이 <= y:  
    return '농어'
```



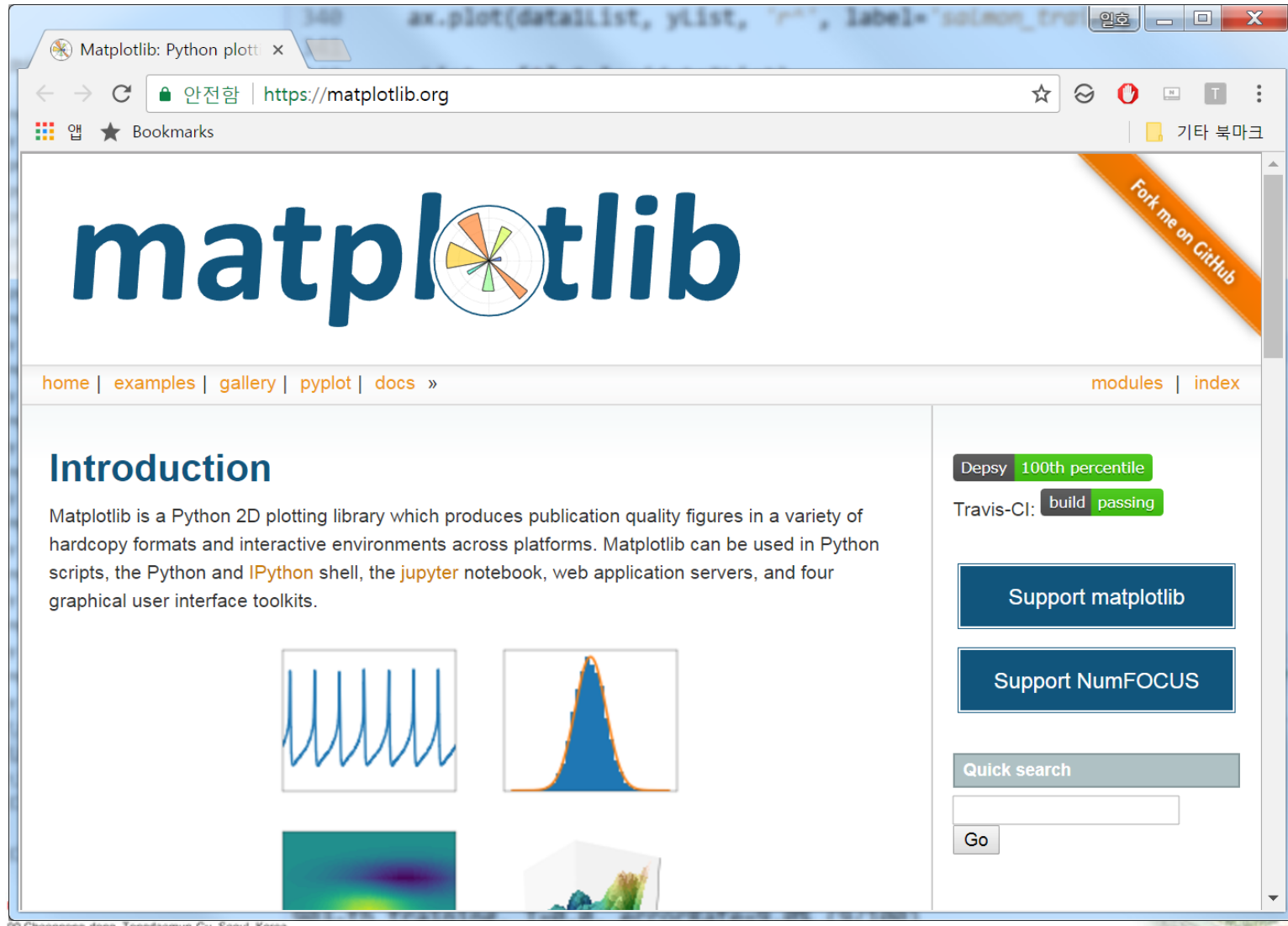
공지

- 에듀클래스 과제제출시 zip 압축 제출할 것
 - Python 코드 하나인 경우도 압축

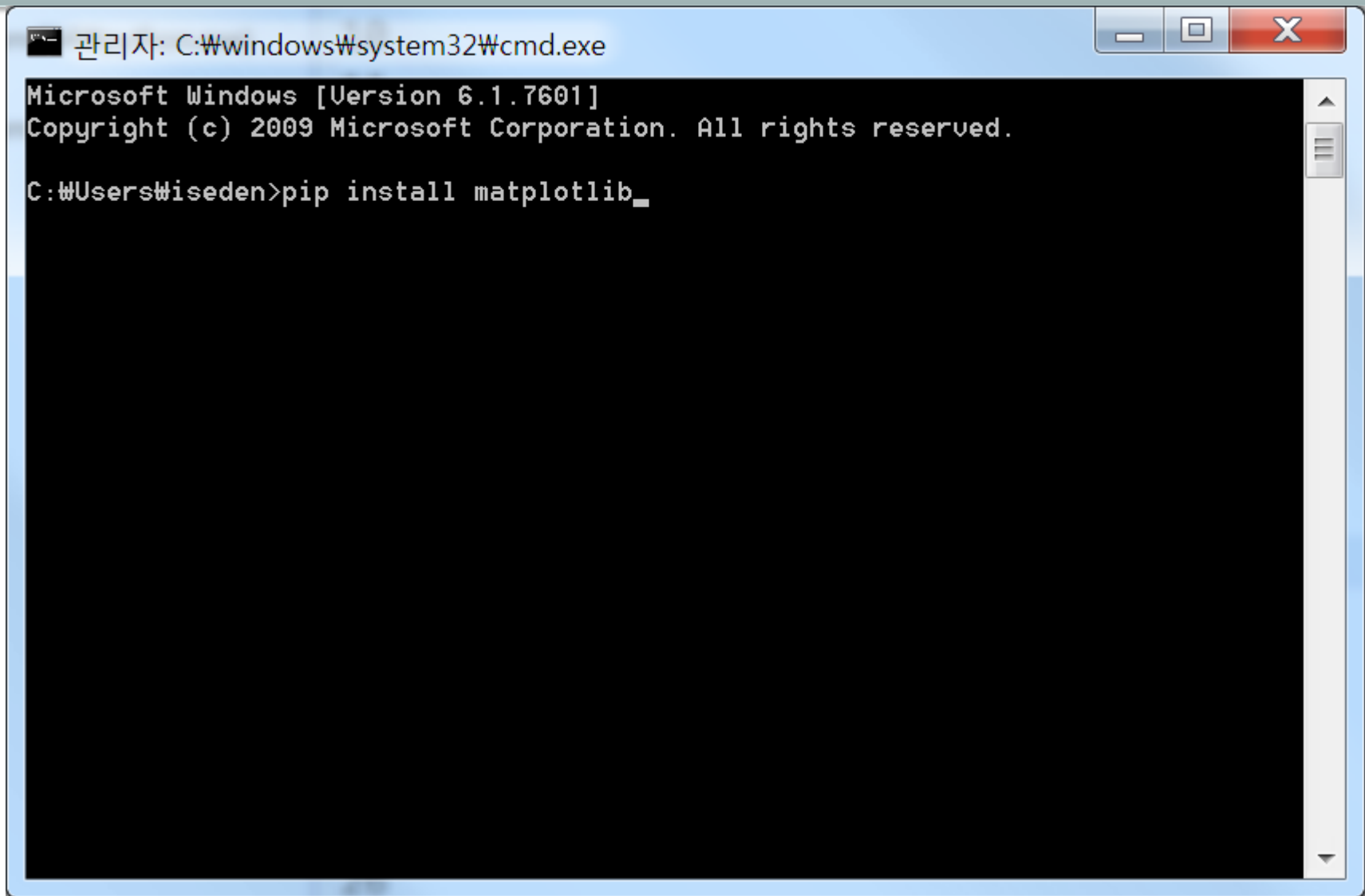


matplotlib

- <https://matplotlib.org/>



matplotlib



A screenshot of a Windows command prompt window. The title bar shows the path '관리자: C:\Windows\system32\cmd.exe'. The window contains the following text: 'Microsoft Windows [Version 6.1.7601]', 'Copyright (c) 2009 Microsoft Corporation. All rights reserved.', and the command 'C:\Users\wiseden>pip install matplotlib_'. The command prompt is currently at the end of the command line, waiting for further input.

```
관리자: C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\wiseden>pip install matplotlib_
```



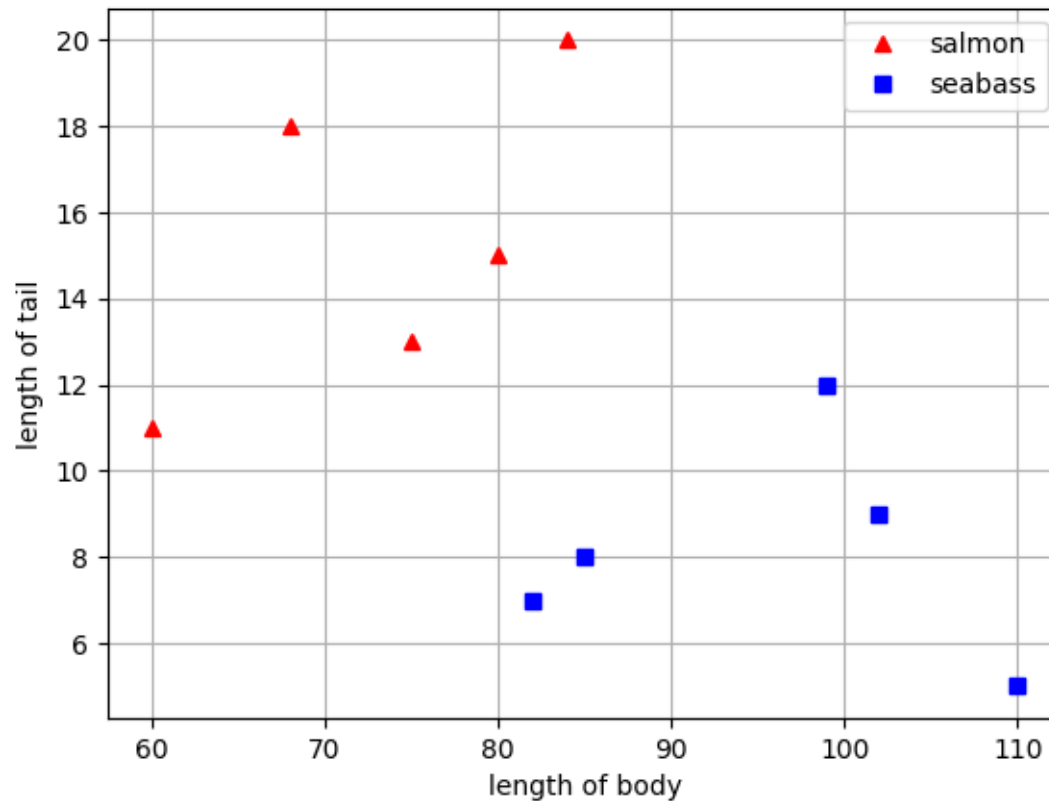
matplotlib

```
관리자: C:\windows\system32\cmd.exe

100% |#####| 8.6MB 87kB/s
Requirement already satisfied: six>=1.10 in c:\scisoft\winpython-64bit-2.7.13.1zero\python-2.7.13.amd64\lib\site-packages (from matplotlib)
Collecting pytz (from matplotlib)
  Downloading pytz-2017.2-py2.py3-none-any.whl (484kB)
100% |#####| 491kB 975kB/s
Collecting cycler>=0.10 (from matplotlib)
  Downloading cycler-0.10.0-py2.py3-none-any.whl
Collecting python-dateutil (from matplotlib)
  Downloading python_dateutil-2.6.1-py2.py3-none-any.whl (194kB)
100% |#####| 194kB 1.2MB/s
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=1.5.6 in c:\scisoft\winpython-64bit-2.7.13.1zero\python-2.7.13.amd64\lib\site-packages (from matplotlib)
Requirement already satisfied: numpy>=1.7.1 in c:\scisoft\winpython-64bit-2.7.13.1zero\python-2.7.13.amd64\lib\site-packages (from matplotlib)
Collecting funtools32 (from matplotlib)
  Downloading funtools32-3.2.3-2.zip
Installing collected packages: pytz, cycler, python-dateutil, funtools32, matplotlib
Running setup.py install for funtools32 ... done
Successfully installed cycler-0.10.0 funtools32-3.2.3.post2 matplotlib-2.0.2 python-dateutil-2.6.1 pytz-2017.2

C:\Users\wisden>
```

matplotlib



matplotlib

```
import matplotlib.pyplot as plt

if __name__ == '__main__':
    data1List = []
    data1List.append([60, 11])
    data1List.append([68, 18])
    data1List.append([80, 15])
    data1List.append([75, 13])
    data1List.append([84, 20])

    data2List = []
    data2List.append([99, 12])
    data2List.append([102, 9])
    data2List.append([110, 5])
    data2List.append([85, 8])
    data2List.append([82, 7])
```



matplotlib

```
fig, ax = plt.subplots()

xList = []
yList = []
for data in data1List:
    x, y = data
    xList.append(x)
    yList.append(y)
ax.plot(xList, yList, 'r^', label='salmon')

xList = []
yList = []
for data in data2List:
    x, y = data
    xList.append(x)
    yList.append(y)
ax.plot(xList, yList, 'bs', label='seabass')
```



matplotlib

```
ax.grid(True)
ax.legend(loc='upper right')
ax.set_xlabel('length of body')
ax.set_ylabel('length of tail')
ax.set_xlim(None, None)
ax.set_ylim(None, None)

plt.savefig('fish1.png')
plt.show()
```



그 밖의 python 트릭들

- `b = copy.deepcopy(a)`
 - `a`의 모든 원소를 `b`로 복사(call by reference 아님)
- `global a`
 - 전역 변수 `a`를 사용할 수 있도록 선언
- `random.uniform(a, b)`
 - $a \sim b$ 사이의 실수 랜덤 생성
- `exp(a)`
 - `math.exp(a)`
 - 간혹 특정 범위 넘어가면 문제 생김
 - `numpy.exp(a)`
 - 사용 추천



Python 코드 구현시 문제 해결 방법

- 발생한 오류 문구를 그대로 입력해서 구글 검색
- 라이브러리별 공식 홈페이지 및 레퍼런스 페이지 참조
 - 파이썬 기본 문법: <https://www.python.org/>
 - Tkinter: <http://effbot.org/tkinterbook/tkinter-index.htm>
 - Matplotlib: <https://matplotlib.org/>
- 라이브러리 코드 리뷰
 - [파이썬 설치 경로] \Lib\site-packages
- 다음 코드로 사용법 확인
 - help([함수명])
 - dir([함수명])



실습과제 (1 / 2, 5점)

- 다음 학습용 2차원 어종 데이터를 파일로부터 읽기
 - “salmon_train.txt” / “seabass_train.txt”
- 선형 분류기로 이를 분류하고 학습 데이터에 대한 오류율 계산
 - 초기 $T = 100$
 - 선형 분류기의 초기 파라미터: $[2.0, -1.0, -180.0]$
 - SA를 이용하여 100회 반복 학습
 - 파라미터 탐색시 현재 위치를 기준으로 다음 구간 사이에서 랜덤 이동:
 - $[(-0.01, +0.01), (-0.01, +0.01), (-10.0, +10.0)]$
 - 매 반복마다 오류율 / 분류 결과등을 콘솔 출력 + **텍스트 파일로 저장**
 - “train_log.txt”
 - $T \neq 0.99$ 로 감쇄
 - T 가 0.001보다 작아지면 중단



실습과제 (2 / 2, 5점)

- 다음 테스트용 2차원 어종 데이터를 파일로부터 읽고 분류
 - “salmon_test.txt” / “seabass_test.txt”
 - 앞에서 학습한 선형 분류기(가장 좋은 파라미터)로 분류할 것
 - 분류 결과를 출력 + 텍스트 파일로 저장
 - “test_output.txt”
 - 결과를 **이미지(test_output.png)**로 저장
 - Matplotlib 사용
 - 선형 분류기는 안 그려도 됨
 - 연어 샘플 (맞은거 / 틀린거)
 - 농어 샘플 (맞은거 / 틀린거)
 - 구분만 할 수 있도록



QnA