

AI 특강

Prof. Jibum Kim

Department of Computer Science & Engineering

Incheon National University

-
- **김지범 (Ph.D):** 출신: 인천광역시 미추홀구, 서인천고등학교 졸업
 - Education: 연세대학교 전기전자공학부 학사, 석사
 - (미) 펜실베니아 주립대학교 컴퓨터공학과 박사 (2012)
 - (미) 로스알라모스 국립연구소 Post Doc (2013)
 - **현재: 인천대학교 컴퓨터공학부 정교수 (2013 – 현재)**

- **연구 분야: 수치 해석, 기하 딥러닝, 인공지능 응용**
- **수상: 2019, 2020 인천대 학술연구상 수상**
- **과제: 현재 연구재단 중견연구, 기초연구실 과제 수행중.
이전 연구재단 신진연구, 중기부 과제, 산업체과제등 수행
경험**
- **특허, 기술이전 실적**
- **최근 5년간 기술이전 2건, 총 3000만원 기술이전 실적**
- **최근 5년간 특허 등록: 10건 이상**
- **현재 인천과학예술평재학교 학생들 STEAM 과제 지도중**

■ 보유하고 있는 특허 기술이 한국전기연구원 통해 인천시 강소특구 사업화 스마트 ICT-E 유망기술로 소개

■ KERI, 창원·안산·청주·인천 강소특구 유망기술 설명회 개최 | 중앙일보 (joongang.co.kr)

The JoongAng 경제

중앙일보 | 입력 2022.10.07 18:52

서명수 기자



 한국전기연구원 Korea Electrotechnology Research Institute		보 도 자 료	
매주 즉시 보도를 요청드립니다.			
배포일시	2022. 10. 5. (수)	매수	본문 2매(별첨 1매)
문의	한국전기연구원 강소특구기획실 안솔기 기술원 T. 055-260-1068 / E-mail: sgahn@keri.re.kr		
창원·안산·청주·인천 강소특구 사업화 유망기술 총 집합			
스마트 ICT-E 분야 16종 핵심기술 소개 강소특구 간 연계 통한 공공기술 사업화·활성화 기대			

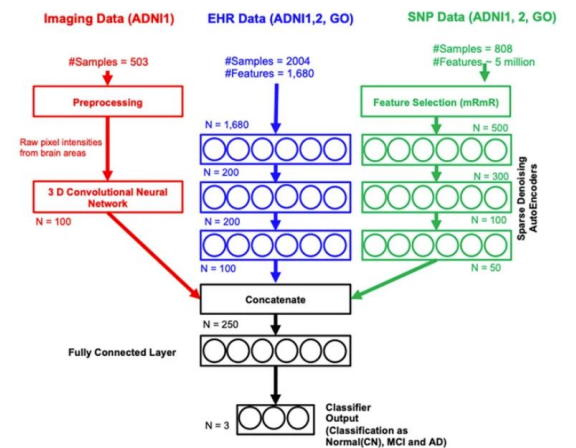
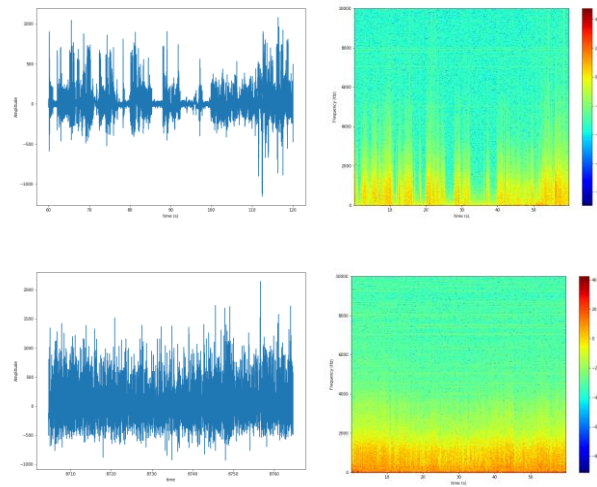
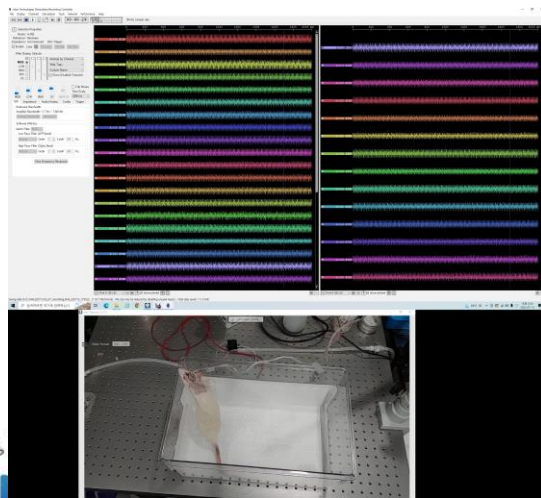
경남 창원 강소특구를 이끌어가는 한국전기연구원(KERI, 원장 직무대행 김남균)이 경기안산·충북청주·인천서구 강소특구와 함께 '스마트 ICT-E 유망기술 설명회'를 5일(수) 오후에 개최했다. 서울 코엑스에서 열린 이날 행사는 50개 기업 100여명의 관계자가 참석했다.

이번 설명회는 공공기술 사업화를 촉진하고, 산·학·연 혁신주체 간 유대 강화를 통해 강소특구 및 관련 기업들의 발전에 기여하기 위해 마련됐다.

설명회에는 4개 강소특구가 보유한 총 16개 기술이 소개됐다. '지능전기 기반 기계 융합'을 특화 키워드로 삼는 창원 강소특구는 KERI를 중심으로 ▲V2G 전기차를 이용한 최적 수요관리 기술(변길성 박사) ▲친환경·고효율 에너지 절감형 영구자석 유도가열기(배준한 박사) ▲전기차 충전제어 및 통신기술(이재조 박사) ▲고성능·저전력 스마트 보청기 시스템(박영진 박사) 기술 설명이 이루어졌다.

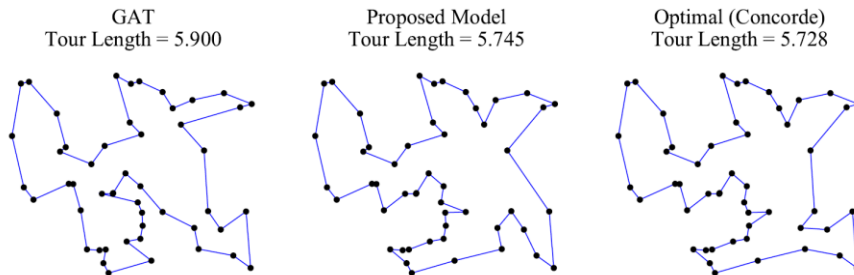
'ICT융복합 부품소재' 특화인 안산 강소특구는 한양대 예리카 캠퍼스를 중심으로 ▲시속각 로봇 그리퍼(최영진 교수) ▲가상/증강현실 공간 항조방법 및 시스템(최명렬 교수) 등의 기술을, '스마트 IT부품·시스템' 개발을 추진하는 청주 강소특구는 충북대를 중심으로 ▲정밀지도를 이용한 라이다센서 캘리브레이션 방법(박태형 교수) ▲서버기반 부품 검사방법 및 그를 위한 시스템 및 장치(박태형 교수) 등의 기술을, 'ICT융복합 환경오염 처리 및 관리' 분야를 다루는 인천서구 강소특구는 인천대를 중심으로 ▲표면오염 추정 기법을 이용한 결로 예측 시스템 및 방법(황광일 교수) ▲수도 관망 내부 누수 여부 모니터링 장치(김지범 교수) 등의 기술을 소개했다.

- **현재 진행중인 연구**
- **1. 뇌파, 행동데이터, 뇌 image등을 사용한 뇌졸중 사전진단 multi-modal 딥러닝 모델 설계 연구 (기초연구실)**
- **동물 실험 기반으로 뇌파 분석, 딥러닝 모델 설계, 기전 파악**

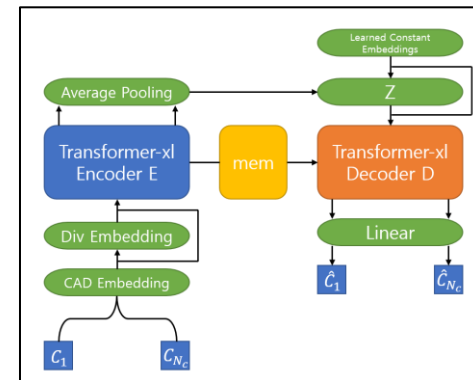
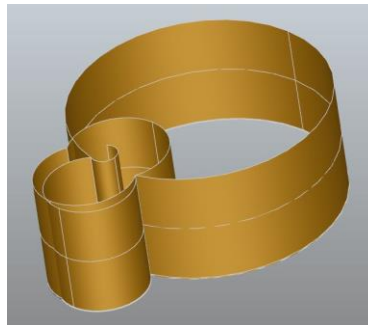
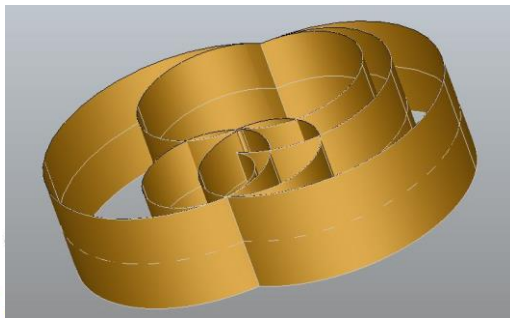


■ 2. 데이터 기반의 기하 딥러닝 기술 연구 (중견연구)

■ 1) 딥러닝 기반의 최적화 문제의 근사해 찾기



■ 2) 딥러닝 기반의 3D CAD 생성 모델 연구



-
- 이재승 (GAI Lab 석사 3학기)
 - E-mail: hunni10@inu.ac.kr
 - 경력: ACM-ICPC (대학생 프로그래밍 경시대회) Korea 2020 본선 26위

- **연구실 인공지능 장비**
- **Multi-GPGPU Data Processing System**
- **1. Nvidia Tesla GPU-V100 6기 (학과 공동 운용)**
- **[NVIDIA V100 | NVIDIA](#)**
- **2. Nvidia Titan XP 8기|x2 (16기)**
- **총 1억원 이상의 장비. 7호관 서버실에 위치함**



■ 특강 내용

10월 22일 특강

1. 인천대 컴퓨터공학과와 AI 관련과목 소개
2. Machine learning이란?
3. Python Pandas를 이용한 data 시각화
4. Scikit-learn과 MNIST dataset을 이용한 이진 분류기 실습 (SVM 선형 분류기)
5. 비지도학습과 K-means 군집화 방법 소개 및 실습

10월 29일 특강

1. Perceptron 개념과 실습
2. 간단한 신경망 구성
3. 경사하강법과 역전파 원리
4. Iris dataset과 신경망을 이용한 분류기 실습

■ 1. 인천대 컴퓨터공학부 소개




-
- 인천대 컴퓨터공학부 역사
 - 1984년: 10월 인천대 전자계산학과 신설 인가로 시작 (정원 50명)
 - 1985년: 제 1회 입학
 - 2015년: 설립 30주년
 - 2017년: 컴퓨터공학부 (주간 78명, 야간 30명)
 - 2022년: 컴퓨터공학부 (주간 108명)

- 인천대학교 컴퓨터 공학부
- 7호관 4층, 5층



- **전임교수: 총 17명**
- **직원 (조교): 4명**
- **재학생 (주/야) – 2022.4.6 기준**
- **총합: 436/102명**
- **4개의 컴퓨터 실습실**
- **자체 서버실 보유**

• 실습실 환경

	컴퓨터실습실1
	위치: 정보기술대 408호 규모: 54석
	컴퓨터실습실2
	위치: 정보기술대 415호 규모: 48석
	컴퓨터실습실3
	위치: 정보기술대 501호 규모: 47석
	컴퓨터실습실4
	위치: 정보기술대 511호 규모: 48석

■ 학과에 AI 관련된 연구를 하시는 교수님들이 다수

■ 컴퓨터공학부 재직 교수진 정보 및 연구 분야

교수명	연구실명	연구 분야
홍윤식	모바일컴퓨팅 응용 연구실	IoT, 빅데이터 기반 모바일 컴퓨팅 응용
성미영	MARVEL 연구실	햅틱, 인간-컴퓨터상호작용, 가상현실
민병준	분산 시스템 연구실	병렬 분산 실시간 시스템, 통신망 관리 기술
채진석	인터넷 SW 연구실	모바일, 인터넷 데이터 기반 SW응용
이면섭	인공지능 게임 연구실	인공지능 게임
이선정	자연어처리 연구실	자연어처리, 인공지능
박종승	엔터테인먼트 컴퓨팅 연구실	게임공학, 증강현실, 시각정보처리
최승식	무선 정보 네트워크 연구실	무선 MAC, 자원관리, 네트워크 성능분석
박문주	시스템 소프트웨어 연구실	인공지능 하드웨어 가속, 저전력 SW, 운영체제
김우일	IMPRESS 연구실	음성처리, 음성인식
김지범	그래픽스 AI 연구실	그래픽스 모델링, 기하 딥러닝
안재균	생물정보학 및 데이터 분석 연구실	인공지능 기반 생물정보학
백형부	실시간 AI 시스템 연구실	인공지능, 실시간 시스템, 비전 시스템
신유현	데이터 인텔리전스 연구실	딥러닝 기반 자연어 처리
구민석	뉴로모픽 컴퓨팅 시스템 연구실	뉴로모픽 컴퓨팅, 인공신경망
최대진	인공지능 및 데이터마이닝 연구실	기계학습 응용, 데이터마이닝
이승수	사이버보안 융합 연구실	클라우드 컴퓨팅 보안, 인공지능 기반 보안

[표 1] 컴퓨터공학부 재직 교수진 및 연구 분야

■ 학부생 대표 기업 취업 현황

- 2022년: 우아한형제들, 현대HT, 한전KDN, 스마일게이트
- 2021년: 삼성 리서치(**Samsung Research**), 네이버, 카카오엔터프라이즈, 쿠팡
- 2020년: 삼성전자, 우아한 형제들, 쿠팡, AJ 네트워크, IBK시스템
- 2019년: KT, 삼성전자, 카카오, GS 리테일, 농협은행, 라인플러스, 롯데정보통신, 삼성 SDS, LG CNS, 예금보험공사, 네오위즈 게임즈, 한전 KDN, 한화 시스템
- 2018년: 삼성전자, 카카오, 카카오게임즈, SK인포섹, 롯데정보통신, GS ITM, 관세청, 동아제약

- 대학원 대표 기업 취업 현황
- 삼성전자, 안랩, NHN, SK 하이닉스, 네이버
파이낸셜등



- 2022학년도 학생부교과성적우수자 전형
- 경쟁률: 13.46대 1
- 경쟁률, 성적 모두 인천대 내에서 최상위권

단과 대학	모집단위	지원현황			성적				교과 환산점			총원 합격
					지원 자	최초 합격자	최종 등록자	최종 등록자	최종등록자			예비 합격 순위
		모집 인원	지원 인원	경쟁률	평균			70% cut	평균	70% cut	가산점 평균	
정보 기술 대학	컴 퓨 터 공 학 부	24	323	13.46	3.40	2.21	2.52	2.75	343.38	342	4.95	55
	정 보 통 신 공 학 과	19	213	11.21	3.55	2.68	2.91	2.99	340.06	339.5	4.96	26
	임베디드시스템공학과	6	58	9.67	3.89	2.35	3.11	3.27	336.93	334.7	4.92	12

- 2024년도 입학생 부터 인천대 컴퓨터공학부 대폭개편
- 교육부 특성화 사업 선정 (사업 기간 2022-2024)
- ‘컴퓨터인공지능’ 학부로 개편하여 AI 부분 강화 계획
- 1. 인공지능 전공
- 2. 컴퓨터전공으로 전공을 2개로 운영 계획 (커리큘럼 2개)



■ 2. 인천대 컴퓨터공학부 AI 관련과목 소개

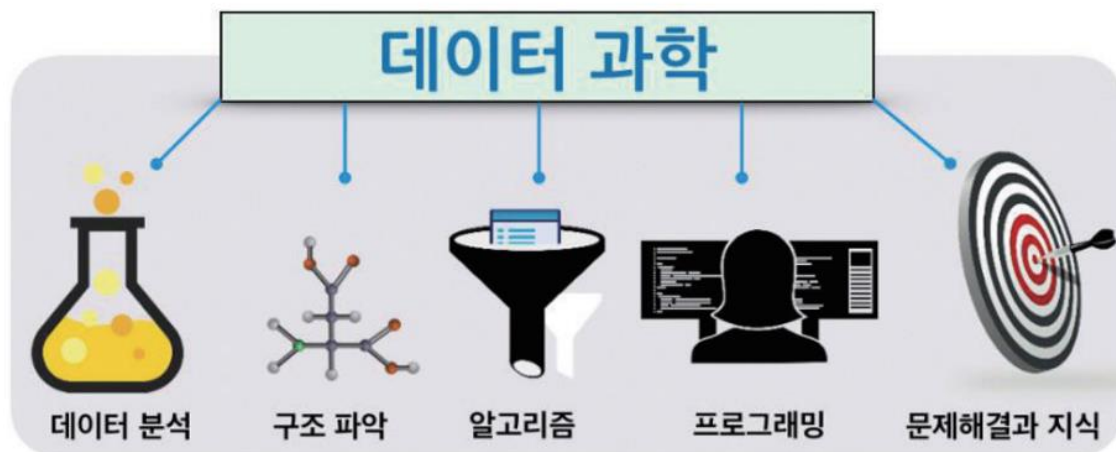
■ 인공지능과 관계된 교과목들

1학년	2학년	3학년	4학년
데이터 사이언스 입문	데이터 사이언스, 시뮬레이션 기초 및 실습	지능정보 시스템	인공지능과 딥러닝, 빅데이터 입문 컴퓨터 비전

-
- 과목 이름: 데이터 사이언스 입문 (1학년)
 - 배우는 내용들
 - Python 기초
 - Python 라이브러리 활용 (NumPy, Matplotlib, Pandas 등)
 - 데이터 전처리, 분석, 활용 (시각화) 방법

데이터 과학 (Data Science)

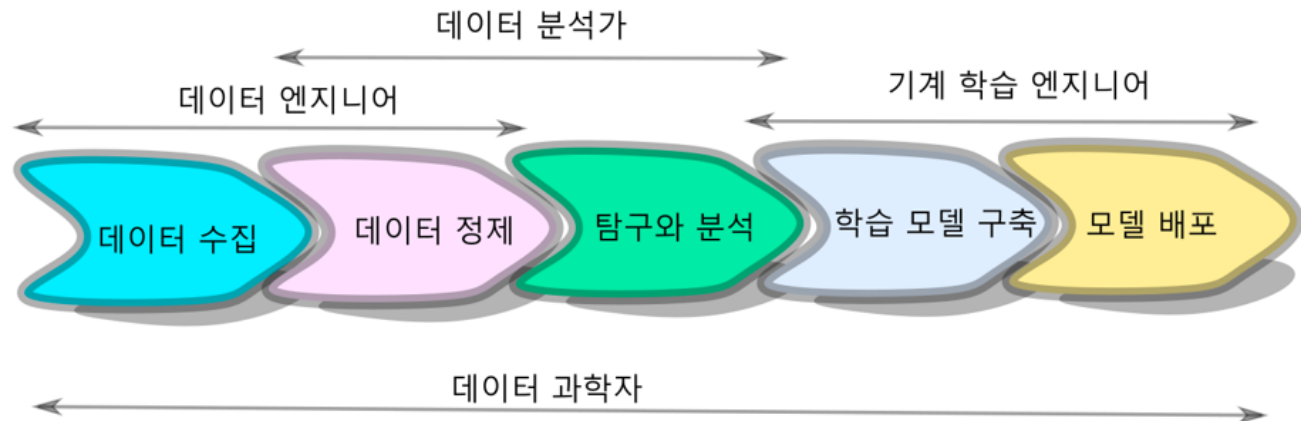
- 데이터과학: data에서 과학적 방법으로 정보나 지식을 추출하는 학문
- 데이터란? 구조화된 정보
 - 다차원 배열 (행렬)
 - 일정하거나 일정하지 않은 간격의 시계열
 - 스프레드시트와 비슷한 데이터 (엑셀에 저장되는 형태)



필수 파이썬 라이브러리

- Numpy: numerical python 의 줄임말
 - ndarray, 행렬계산, 선형대수, 난수, C API 등
 - 데이터 컨테이너 역할, 많은 경우 numpy배열을 기본 자료구조로 가정
- Pandas: panel data + python data analysis
 - 표 형태의 row와 column 이름을 갖는 dataframe + 1D series
 - 색인 기능 제공 -> 데이터 변형, 자르기, 취합, 부분집합 선택 가능
- Matplotlib: 가장 기본이 되는 시각화 도구
- Scipy: 과학 계산, numpy와 사용하면 전통적인 과학계산 대부분 가능
 - 미적분, 선형대수, 행렬 분해, 최적화, 희소 행렬, 확률분포
- Scikit-learn: 범용 머신러닝 도구
 - 분류, 회귀, 클러스터링, 차원축소, 모델 선택, 전처리 등의 모듈 포함

데이터 처리과정



-
- 과목 이름: 데이터 사이언스 (2학년)
 - 배우는 내용들
 - 경사하강법 (GD), 확률적 경사하강법 (SGD)
 - 인공신경망 동작원리
 - 합성곱 신경망 (CNN)
 - 순환 신경망 (RNN)
 - 오토 인코더 (Autoencoder)등

- 데이터사이언스는 데이터를 분석하고 연구하는 학문입니다.
- 데이터사이언스는 데이터에 대한 정리, 변환, 패턴 분석, 시각화, 예측 모델 개발 등의 포괄적이고 종합적인 절차를 포함합니다.
- 전반부에는, 데이터 분석의 기본 과정과 딥러닝의 기초인 **인공신경망(artificial neural network)**의 동작원리에 대하여 충분히 이해합니다.
- 후반부에는, 데이터를 분석하는 효과적인 방법인 **딥러닝**의 다양한 기법을 실습을 통해 체득합니다.
- **(수업목표)**
 1. 데이터사이언스 개념을 이해합니다.
 2. 인공신경망의 기본 개념과 이론을 학습합니다.
 3. 파이토치를 이용하는 딥러닝 기법들을 실습을 통하여 익힙니다.
 4. 학습한 이론과 방법을 응용하여 실제 문제를 해결하는 나만의 데이터사이언스 모델을 설계하고 구현해 봅니다.

- **(수강조건)**

파이썬 프로그래밍 기본 능력을 갖추시고 본 교과목을 수강하시기를 권합니다.

선형대수학(linear algebra) 교과목을 수강하셨거나, 기초수학(미분, 벡터, 행렬, 확률, 통계 등)에 대한 기본 개념을 이해하시고 본 교과목을 수강하시기를 권합니다.

Week	Date	Chapter	Topic	Remark
1 st week		이러닝 자료	수업 소개 파이썬 기초 (Lab1) 파이썬 속성 강좌 (→Quiz1)	
2 nd week		이러닝 자료	파이썬 라이브러리 활용 연습 numpy matplotlib pandas (Lab2,3,4)	
3 rd week		이러닝 자료	생기초 수학 선형대수 (Lab5) 생기초 수학 통계 (Lab6) 생기초 수학 확률 (Lab7)	Quiz1
4 th week		이러닝 자료 이러닝 자료 교재 APPENDIX A Chapter 1~5 Chapter 6~9	경사하강법 (Lab8) 계산그래프 (Lab9) 기초 미분 연습 인공 신경망의 동작 원리 1 인공 신경망의 동작 원리 2	
5 th week		Chapter 10~16	인공 신경망의 동작 원리 3 인공 신경망의 순전파, 역전파, 가중치 업데이트 (NNLab1)	
6 th week		Chapter 17~19 Chapter 20	파이썬 시작하기 파이썬으로 인공신경망 만들기	
7 th week		퀴즈 Chapter 21	신경망 첫걸음 PART 1 (→Quiz2) MNIST 손글씨 데이터 인식하기 (NNLab2)	Quiz2
8 th week		Chapter 22~24	더 재미있는 것들 (NNLab3)	
9 th week		이러닝 자료 중간시험	신경망을 이용한 붓꽃 데이터 분류 (Homework) 중간시험 필기+실기	HW Exam
10 th week		1장~2장 3장	딥러닝, 파이토치 선형회귀 분석	
11 th week		4장 5장	인공 신경망 합성곱 신경망 CNN	
12 th week		6장 7장	순환 신경망 RNN 모델 최적화	
13 th week		8장 9장	뉴럴 스타일 트랜스퍼 오토인토더	
14 th week		10장 기말시험	생성적 적대 신경망 GAN 기말시험 필기+실기	Exam
15 th week		프로젝트 발표	나만의 데이터사이언스 모델 제작 발표	Project Report

교재

1. 신경망 첫걸음, 타리크 라시드 지음, 송교석 옮김, 한빛미디어, 2017.

- <http://preview2.hanbit.co.kr/books/bwrx/>
- <https://github.com/makeyourownneuralnetwork/makeyourownneuralnetwork>

Make Your Own Neural Network, Tariq Rashid, CreateSpace Independent Publishing Platform, 2016.

- <https://www.amazon.com/Make-Your-Own-Neural-Network-ebook/dp/B01EER4Z4G>
- 한글판 pdf available

2. 파이토치 첫걸음, 최건호, 한빛미디어, 2019.

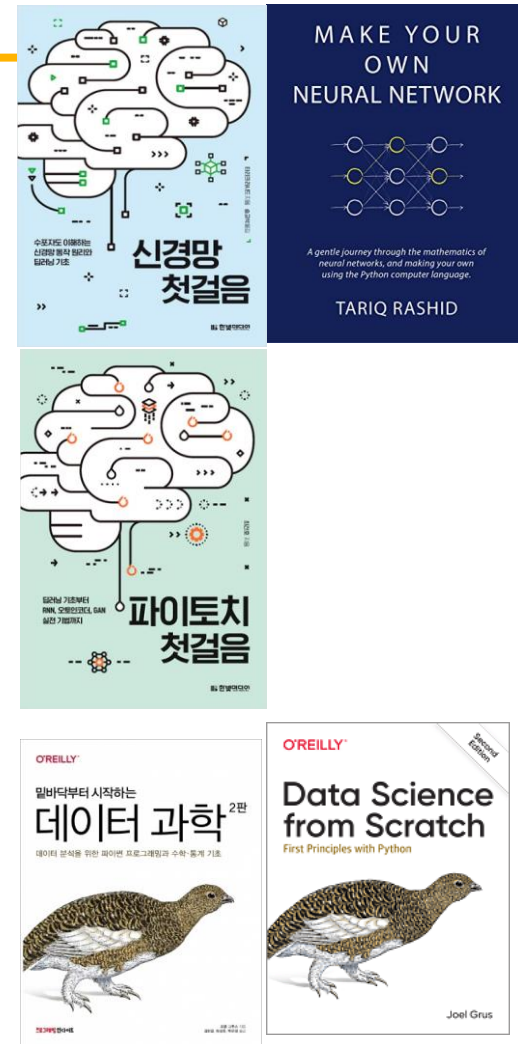
- http://www.hanbit.co.kr/store/books/look.php?p_code=B7818450418

3. (일부사용) 밑바닥부터 시작하는 데이터 과학 2판, 조엘 그루스 지음 | 김한결, 하성주, 박은정 옮김 | 프로그래밍인사이트 | 2020년 3월.

- <https://github.com/insightbook/Data-Science-from-Scratch>

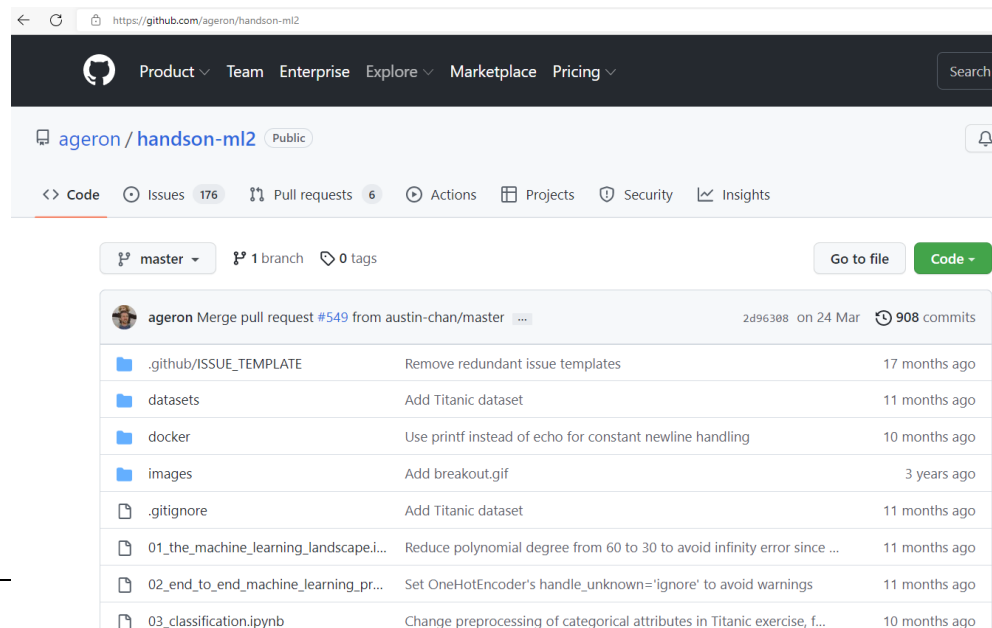
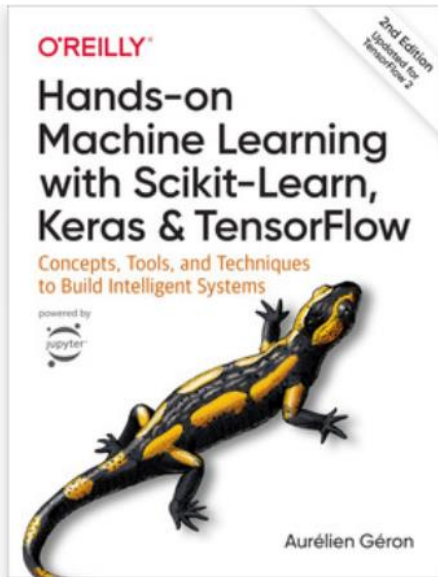
Data Science from Scratch, 2nd Edition, Data Science from Scratch, 2nd Edition, by Joel Grus, O'Reilly Media, Inc., May 2019.

- <https://github.com/joelgrus/data-science-from-scratch>



-
- 3. (이론) Machine learning 이란?
 - 핸즈온 책 chapter 1

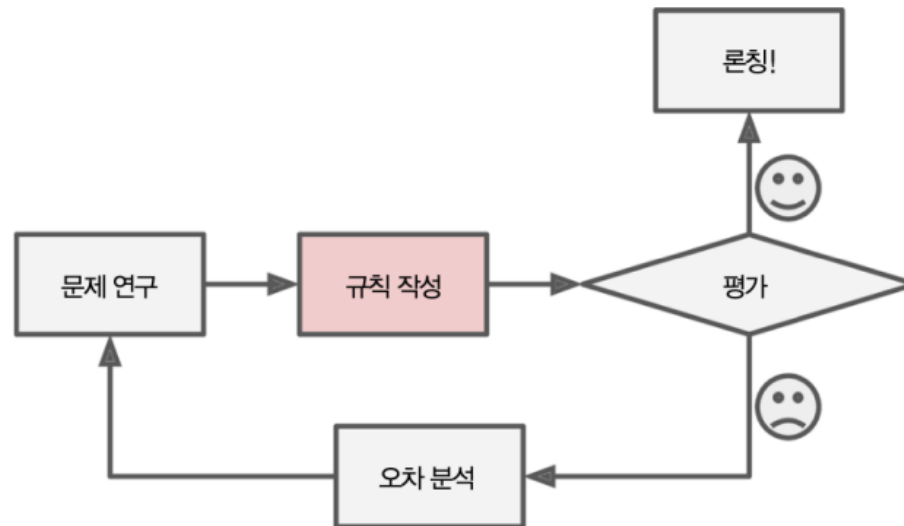
- 데이터 사이언스나 인공지능 입문으로 좋은 책인
- “Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition”의 챕터 1, 2와 3의 일부를 오늘 실습에 소개
- 구글에서 위의 이름으로 검색하면 깃허브도 있음



-
- **Machine learning (ML)? 한글로 기계 학습**
 - **1. Machine learning is the science of programming computers so they can learn from data**
 - **2. Machine learning is the field of study that gives computers the ability to learn without being explicitly programmed (명시적으로 규칙을 정하기보다는 학습을 통해서 배움)**

- 스팸 필터 예
- 1. 기존의 프로그래밍 기술
- 1) 스팸 메일이 어떻게 보이는지 관찰. 특정 단어나 구절 (예: “credit card”, “amazing”)이 많이 보인다
- 2) 각각의 패턴에 대해서 규칙을 작성해서 1)에서 관찰한 패턴이 여러 개 보이면 스팸 메일이라고 판단
- 3) 작성한 프로그램을 평가 (test)하고 성능이 좋을때까지 1)과 2) 반복

전통적인 프로그래밍

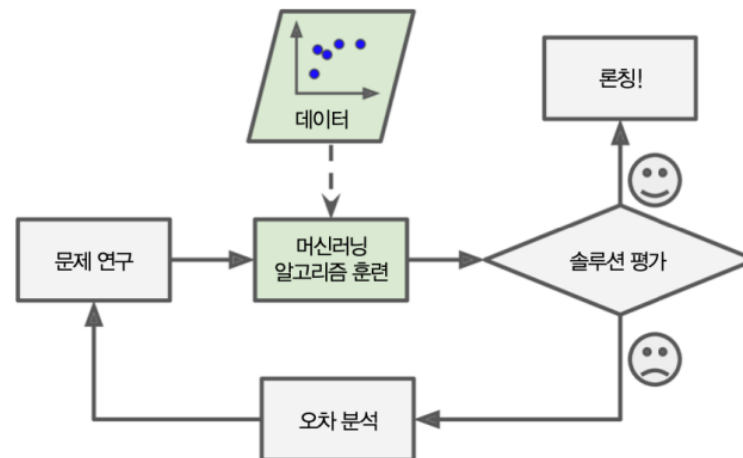


- 전통적인 프로그래밍 접근 방법은 다음과 같다.
 - **문제 연구**: 누군가가 문제를 해결하기 위해 해결책을 찾음
 - **규칙 작성**: 결정된 규칙을 개발자가 프로그램을 작성
 - **평가**: 만들어진 프로그램을 테스트
 - 문제가 없다면 **론칭**, 문제가 있다면 **오차**를 분석한 후 처음 과정부터 다시 실시

- 이러한 전통적인 방법의 문제점?
- 스팸 분류 프로그램을 작성되고 론칭된 후에 **새로운 스팸단어가 생기면** 프로그램이 이 단어를 자동으로 분류할 수 없음
- 예를 들어 최초에 스팸메일들에서 “4U”라는 단어가 포함된 스팸메일을 보냈고 이것이 차단되고 다시 “for you”가 포함된 스팸메일을 보낸 경우
- **개발자가 새로운 규칙을 매번 새롭게 업데이트 해주어야 함**
- 새로운 규칙이 생겼을 때 사용자가 매번 업데이트를 해주어야하므로 **유지 보수가 어렵다**

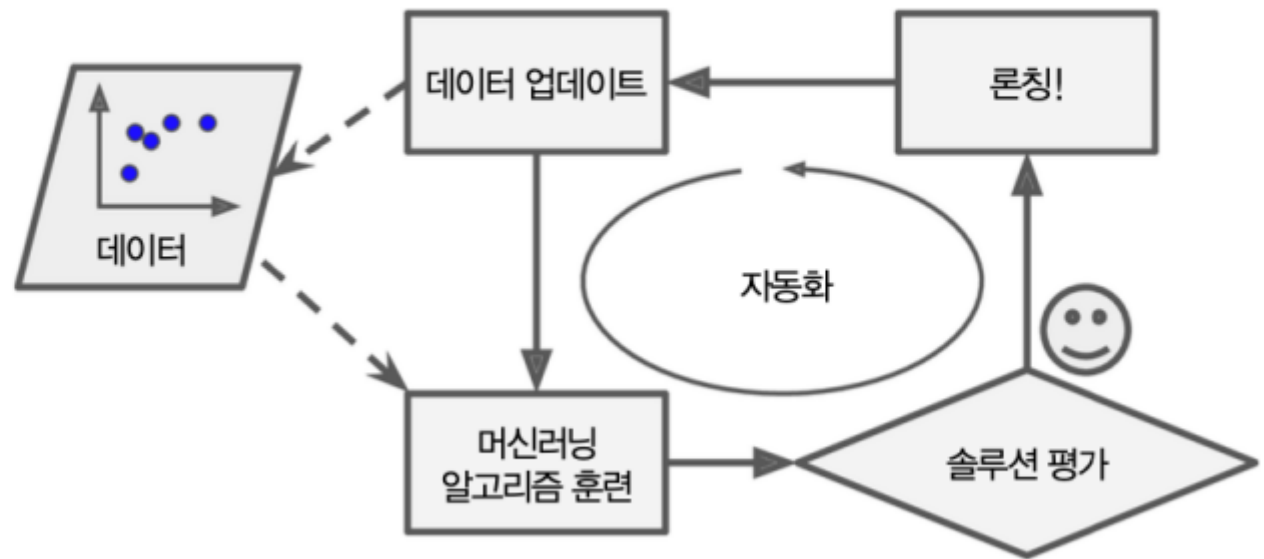
- Machine learning 방법
- 데이터로부터 스스로 스팸메일의 특징을 찾음
- 차이: 문제 연구후에 규칙이 아닌 머신 러닝 알고리즘 훈련 (학습)을 통해 주어진 데이터를 바탕으로 훈련함

머신러닝



- 예를 들어 최초에 스팸 메일들에서 “4U”라는 단어가 포함된 스팸메일을 보냈고 이것이 차단되고 다시 “for you”가 포함된 스팸메일을 보낸 경우
- Machine learning의 경우 사용자가 스팸으로 지정한 메일에 'For U'가 자주 등장하는 경우 **이 단어가 포함된 이메일을 스팸 으로 분류하도록 스스로 학습**
- **장점: 규칙을 매번 새롭게 정할 필요가 없고 자동화 가능**

머신러닝 학습 자동화



-
- 실제 데이터를 가지고 ML 모델을 학습시키고 테스트 시킬때는 전체 데이터를 나누어서 사용
 - **Training set:** 머신 러닝 프로그램이 훈련 (학습)하는데 사용하는 데이터 집합
 - **Test set:** 실제 머신 러닝 프로그램이 얼마나 잘 동작하는지 테스트 (평가)하는 데이터 집합

-
- **Supervised learning (지도 학습)**, unsupervised learning (비지도 학습), semi-supervised learning (준지도 학습), Reinforcement learning (강화 학습)
 - In supervised learning, the training data you feed to the algorithm includes the **desired solutions, called labels**
 - Label의 예: 스팸 메일이면 1, 정상 메일이면 0

-
- A typical supervised learning task is **classification (분류)**
 - The spam filter is a good example of this: it is trained with many example emails along with **their class (spam or not)**, and it must learn how to classify new emails

분류



- 특성을 사용한 데이터 분류
- 예제: 스팸 필터

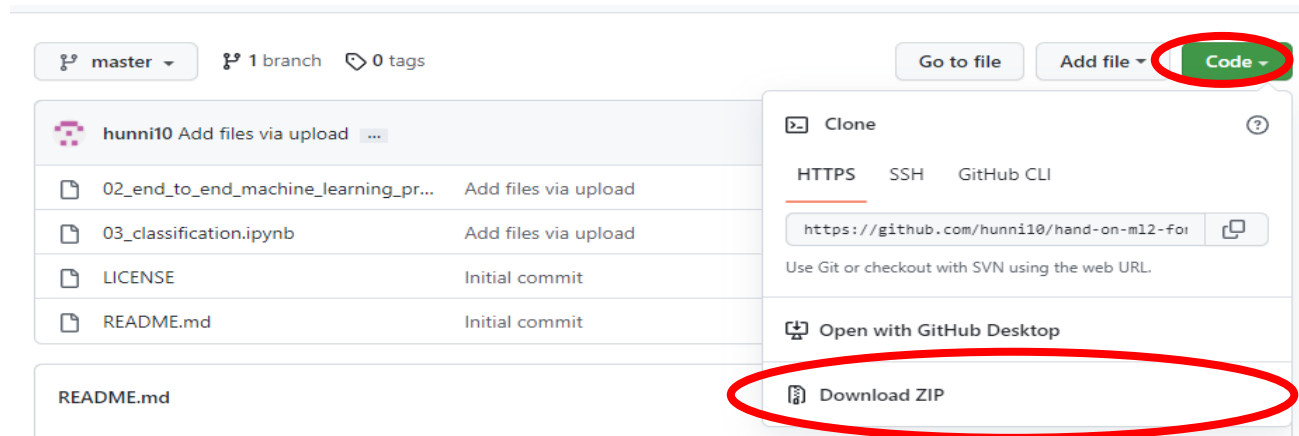
-
- 3. (실습) Python의 Pandas를 이용하여 실제 데이터를 갖고 간단한 data 시각화 및 상관 관계 분석 (핸즈온 책 chapter 2)

-
- 유명한 실제 공개 데이터 저장소
 - UC 얼바인 머신러닝 저장소
 - UCI Machine Learning Repository
 - Kaggle dataset
 - Find Open Datasets and Machine Learning Projects | Kaggle

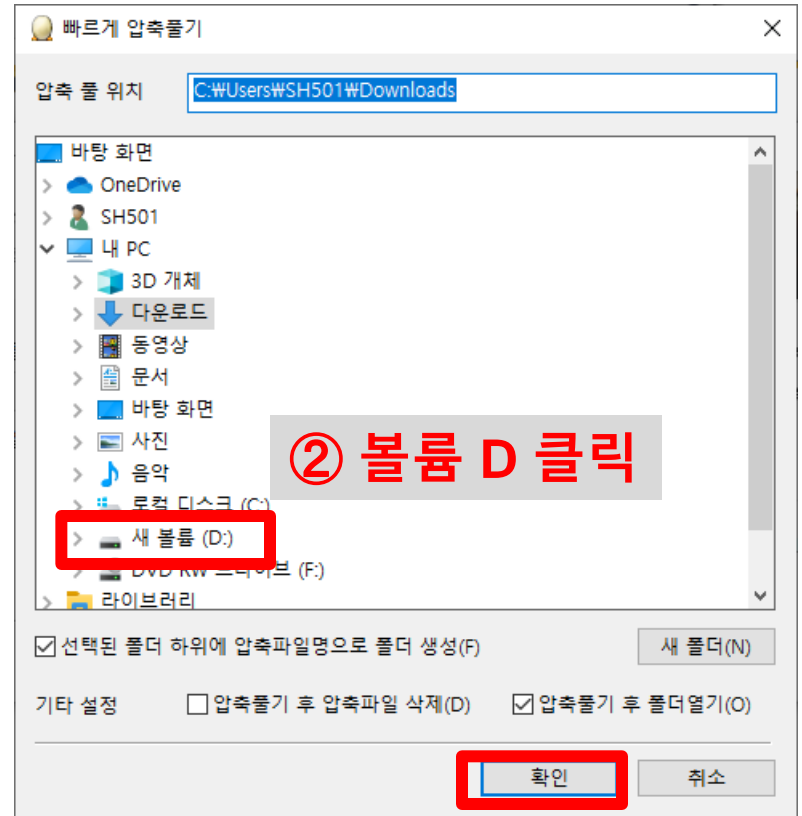
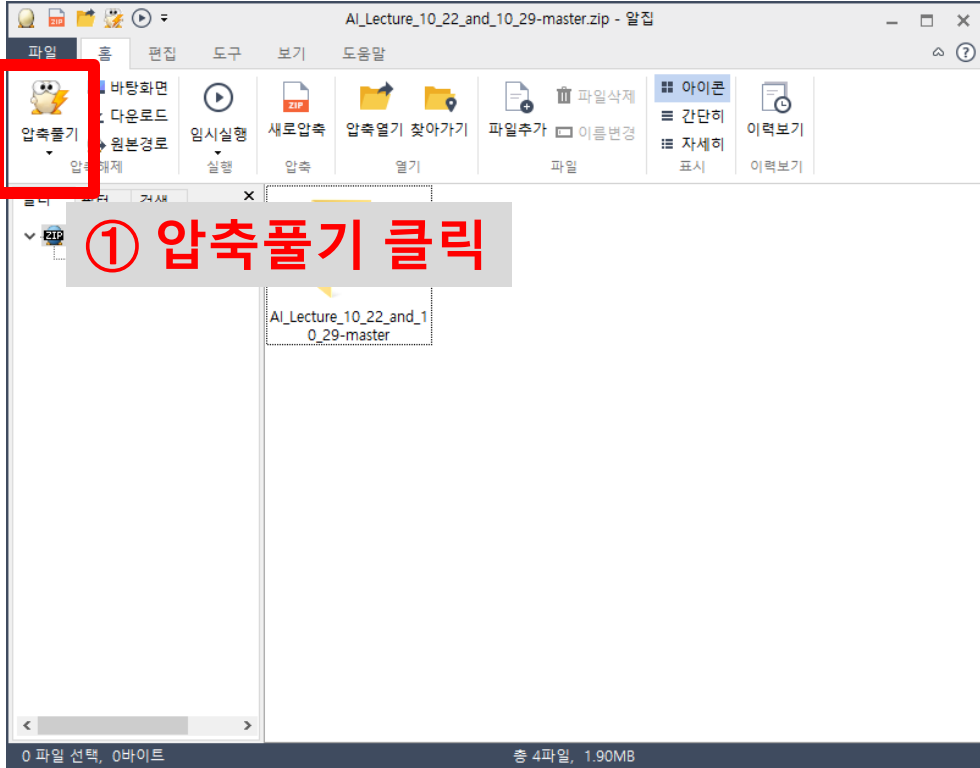
- 오늘 최초 실습에서 사용할 실제 데이터
- 주어진 데이터: 미국 캘리포니아 인구조사 데이터
- 특징: 구역(block)별 인구, 중간 소득, 경도, 위도 등
- Label: 중간 주택 가격
- 학습 내용: 데이터 시각화, 특징간 상관관계
- 사용할 라이브러리: Python Pandas (데이터 처리 분석), NumPy (행렬 수치 연산) Matplotlib (시각화), Scikit-learn (머신러닝 알고리즘 구현) 등

■ 실습 시작

- GitHub -
hunni10/Al_Lecture_10_22_and_10_29:
수업자료 공유용입니다. source:
<https://github.com/ageron/handson-ml2>



다운로드받은 ZIP 파일 열기



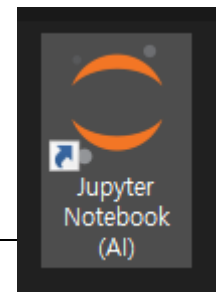
③ 확인 클릭

- 오늘 실습에서는 Python 코드를 사용 예정
- Jupyter Notebook을 사용 예정

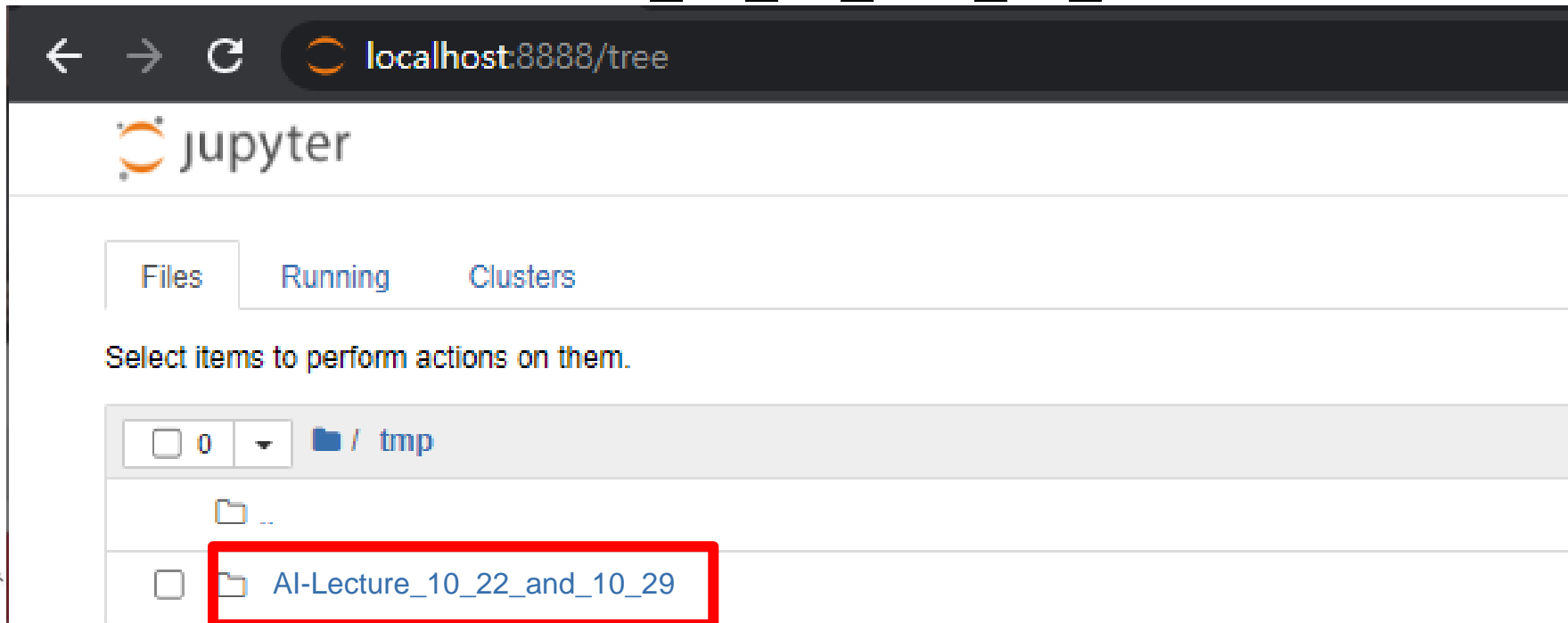
The Jupyter Notebook

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

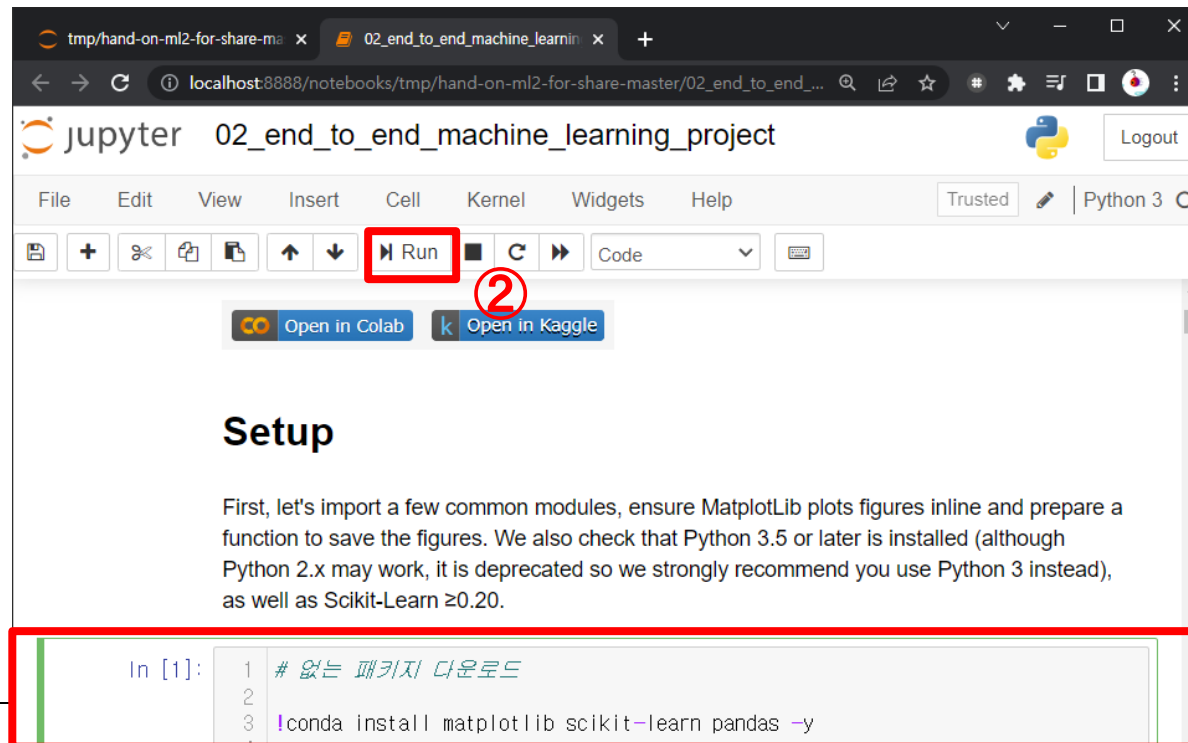
- 바탕화면의 “Jupyter Notebook (AI)” 실행



- 크롬 브라우저에 표시되는 Jupyter notebook
- 실수로 크롬탭을 꺾을 경우 크롬 주소창에 “localhost:8888” 입력
- 압축 푼 “AI-Lecture_10_22_and_10_29” 폴더 클릭



- “02_end_to_end_machine_learning_project.ipynb” 클릭
- 실행하고픈 코드블럭(Cell) 클릭-> 상단 바의 “Run” 클릭



① 클릭

■ 실행중 (별표시)

```
In [*]: 1 # 없는  
2  
3 !conda  
4  
5 # Pytho  
6 import  
7 assert  
8  
9 # Sciki  
10 import  
11 assert  
12  
13 # Commo  
14 import  
15 import  
16
```

■ 실행완료 (숫자표시)

```
In [1]: 1 # 없는  
2  
3 !conda  
4  
5 # Pytho  
6 import  
7 assert  
8  
9 # Sciki  
10 import  
11 assert  
12  
13 # Commo  
14 import  
15 import  
16
```

- 코드블럭(Cell)의 출력 결과는 블록 하단에 나타남
- (코드에 따라서 실행이 끝나고 출력이 없을 수도 있음)

```
33 print("saving figure", fig_id)
34 if tight_layout:
35     plt.tight_layout()
36 plt.savefig(path, format=fig_extension, dpi=resolution)
```

```
==> WARNING: A newer version of conda exists. <==
current version: 4.9.1
latest version: 4.14.0
```

Please update conda by running

```
$ conda update -n base -c defaults conda
```

```
Collecting package metadata (repodata.json): ...working... done
Solving environment: ...working... done
```

```
# All requested packages already installed.
```

-
- https://github.com/hunni10/AI_Lecture_10_22_and_10_29/blob/master/02_end_to_end_machine_learning_project.ipynb
 - 교재 깃허브의 chapter 2
 - 설치되어있는 Jupyter Notebook로 실행

-
- 제일 먼저 ‘Setup’ 부분 실행 확인
 - 에러가 뜨지는 않는지?
 - Python 3과 Scikit-Learn 0.20이상 버전 설치되어있어야함
 - ‘Get the Data’ 확인
 - ‘Take a Quick Look at the Data Structure’ 확인

- **Scikit Learn**

- Machine learning 분야에서 가장 폭넓게 사용되는 tool 중의 하나
- Classification, Regression, Clustering 등
- scikit-learn: machine learning in Python
— scikit-learn 1.1.2 documentation

Take a Quick Look at the Data Structure

```
In [5]: housing = load_housing_data()  
housing.head()
```

```
Out[5]:
```

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value	ocean_proximity
0	-122.23	37.88	41.0	880.0	129.0	322.0	126.0	8.3252	452600.0	NEAR BAY
1	-122.22	37.86	21.0	7099.0	1106.0	2401.0	1138.0	8.3014	358500.0	NEAR BAY
2	-122.24	37.85	52.0	1467.0	190.0	496.0	177.0	7.2574	352100.0	NEAR BAY
3	-122.25	37.85	52.0	1274.0	235.0	558.0	219.0	5.6431	341300.0	NEAR BAY
4	-122.25	37.85	52.0	1627.0	280.0	565.0	259.0	3.8462	342200.0	NEAR BAY

-
- **Python의 pandas의 dataframe 활용**
 - **head(): 데이터의 상위 N행 봄. 기본 N=5**
 - **info(): 데이터에 대한 전반적인 정보**
 - **describe(): 열별 요약 통계량 (수치형만)**
 - **hist(): histogram 보기**

- **특이점: 다른 열 (수치형)과 다르게 ocean_proximity 열은 자료형이 다름 (object형)**

In [6]:

```
housing.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 20640 entries, 0 to 20639
```

```
Data columns (total 10 columns):
```

#	Column	Non-Null Count	Dtype
0	longitude	20640 non-null	float64
1	latitude	20640 non-null	float64
2	housing_median_age	20640 non-null	float64
3	total_rooms	20640 non-null	float64
4	total_bedrooms	20433 non-null	float64
5	population	20640 non-null	float64
6	households	20640 non-null	float64
7	median_income	20640 non-null	float64
8	median house value	20640 non-null	float64
9	ocean_proximity	20640 non-null	object

```
dtypes: float64(9), object(1)
```

```
memory usage: 1.6+ MB
```

■ “ocean_proximity”: 해안 근접도

```
In [7]: housing["ocean_proximity"].value_counts()
```

```
out[7]: <1H OCEAN      9136  
        INLAND    6551  
        NEAR OCEAN 2658  
        NEAR BAY   2290  
        ISLAND      5  
        Name: ocean_proximity, dtype: int64
```

■ 각 열별 통계정보 보여줌

In [8]: `housing.describe()`

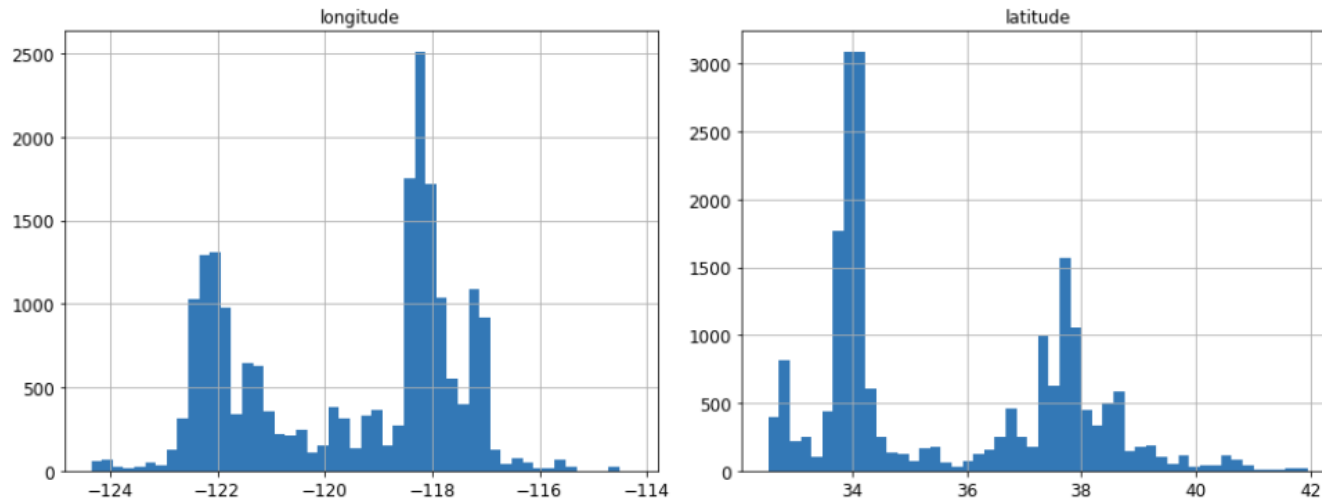
Out[8]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value
count	20640.000000	20640.000000	20640.000000	20640.000000	20433.000000	20640.000000	20640.000000	20640.000000	20640.000000
mean	-119.569704	35.631861	28.639486	2635.763081	537.870553	1425.476744	499.539680	3.870671	206855.816909
std	2.003532	2.135952	12.585558	2181.615252	421.385070	1132.462122	382.329753	1.899822	115395.615874
min	-124.350000	32.540000	1.000000	2.000000	1.000000	3.000000	1.000000	0.499900	14999.000000
25%	-121.800000	33.930000	18.000000	1447.750000	296.000000	787.000000	280.000000	2.563400	119600.000000
50%	-118.490000	34.260000	29.000000	2127.000000	435.000000	1166.000000	409.000000	3.534800	179700.000000
75%	-118.010000	37.710000	37.000000	3148.000000	647.000000	1725.000000	605.000000	4.743250	264725.000000
max	-114.310000	41.950000	52.000000	39320.000000	6445.000000	35682.000000	6082.000000	15.000100	500001.000000

■ 히스토그램 시각화 (각 열별)

```
In [9]: %matplotlib inline
import matplotlib.pyplot as plt
housing.hist(bins=50, figsize=(20,15))
save_fig("attribute_histogram_plots")
plt.show()
```

Saving figure attribute_histogram_plots

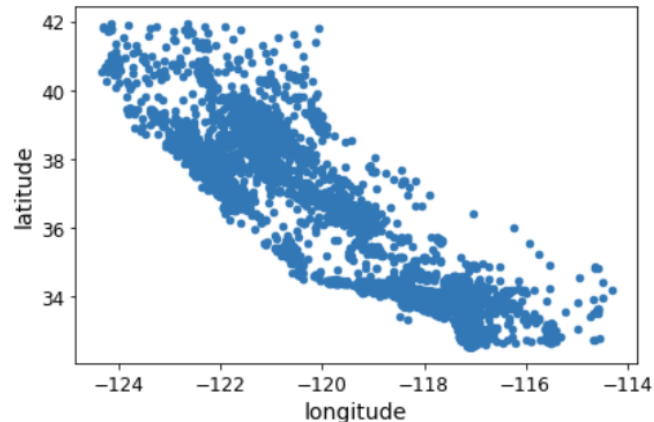


- **Training data 시각화:** 지리적 데이터 시각화 (scatter plot)
- 구역이 집결된 지역과 그렇지 않은 지역 구분 가능
- 샌프란시스코의 베이 에어리어, LA, 샌디에고 등 밀집된 지역 확인 가능
- 아래: bad visualization 예

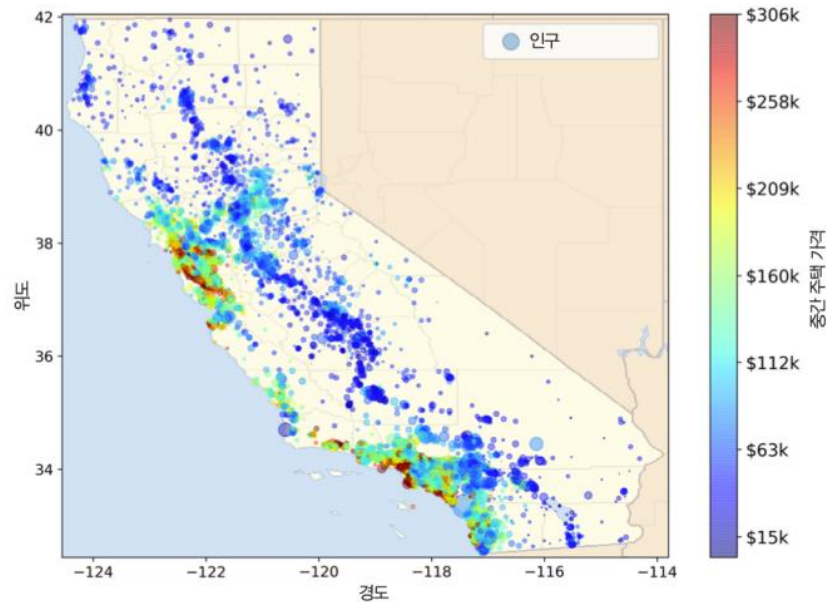
In [33]:

```
housing.plot(kind="scatter", x="longitude", y="latitude")  
save_fig("bad_visualization_plot")
```

Saving figure bad_visualization_plot



- 주택 가격이 해안 근접도, 인구 밀도와 관련이 큼
- 해안 근접도: 위치에 따라 다르게 작용 **대도시 근처: 해안 근처 주택 가격이 상대적 높음**
- 북부 캘리포니아 지역: 높지 않음

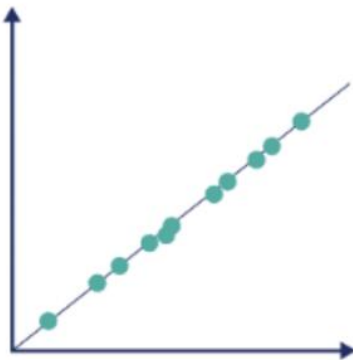


- 상관계수 조사
- 중간 주택 가격 특성과 다른 특성 사이의 상관계수 활용

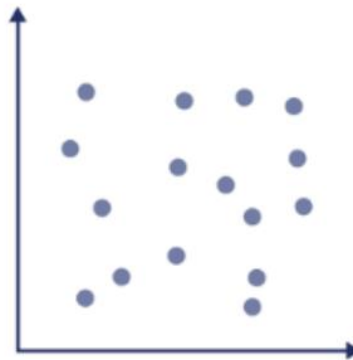
```
In [39]: corr_matrix["median_house_value"].sort_values(ascending=False)
```

```
Out[39]: median_house_value    1.000000  
         median_income        0.687160  
         total_rooms          0.135097  
         housing_median_age    0.114110  
         households           0.064506  
         total_bedrooms        0.047689  
         population           -0.026920  
         longitude            -0.047432  
         latitude             -0.142724  
         Name: median_house_value, dtype: float64
```

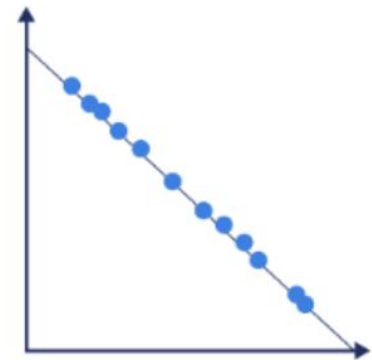

Perfect positive correlation



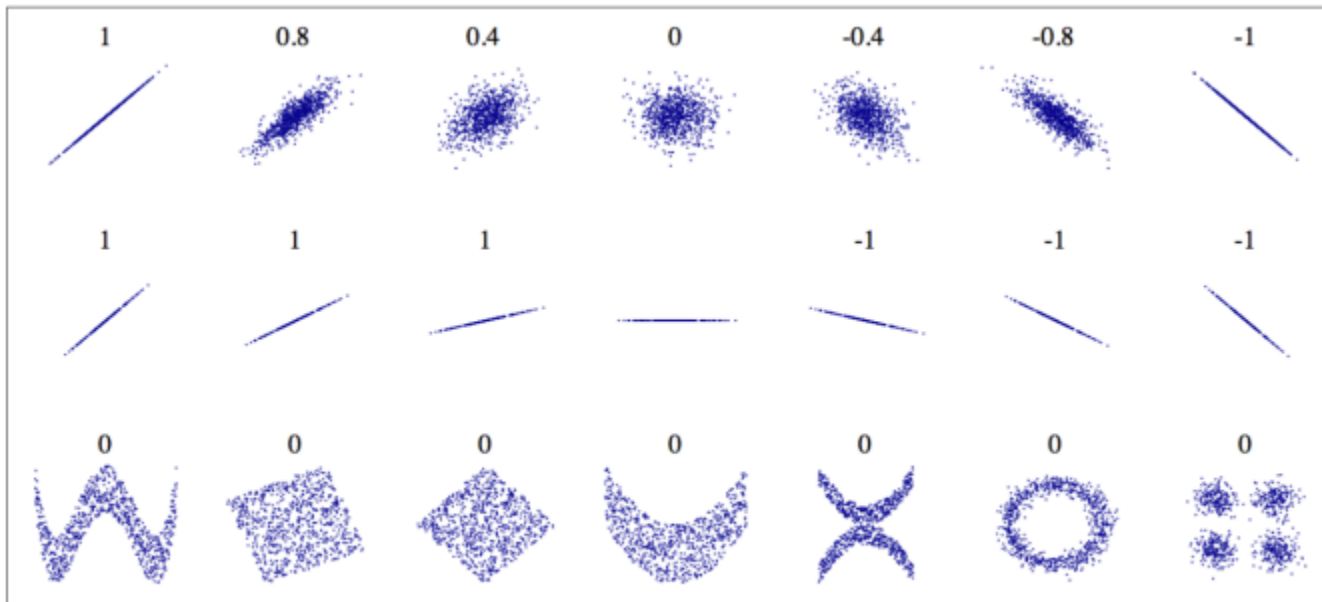
Zero correlation



Perfect negative correlation



- 상관계수 (standard correlation coefficient)의 특징
- 상관계수: $[-1, 1]$ 구간의 값
- 1에 가까울 수록: 강한 양의 선형 상관관계
- -1에 가까울 수록: 강한 음의 선형 상관관계
- 0에 가까울 수록: 매우 약한 선형 상관관계



- 상관계수를 통해 확인할 수 있는 정보 중간 주택 가격과 중간 소득의 상관계수가 0.68로 가장 높음
중간 소득이 올라가면 중간 주택 가격도 상승하는 경향이 있음

```
In [39]: corr_matrix["median_house_value"].sort_values(ascending=False)
```

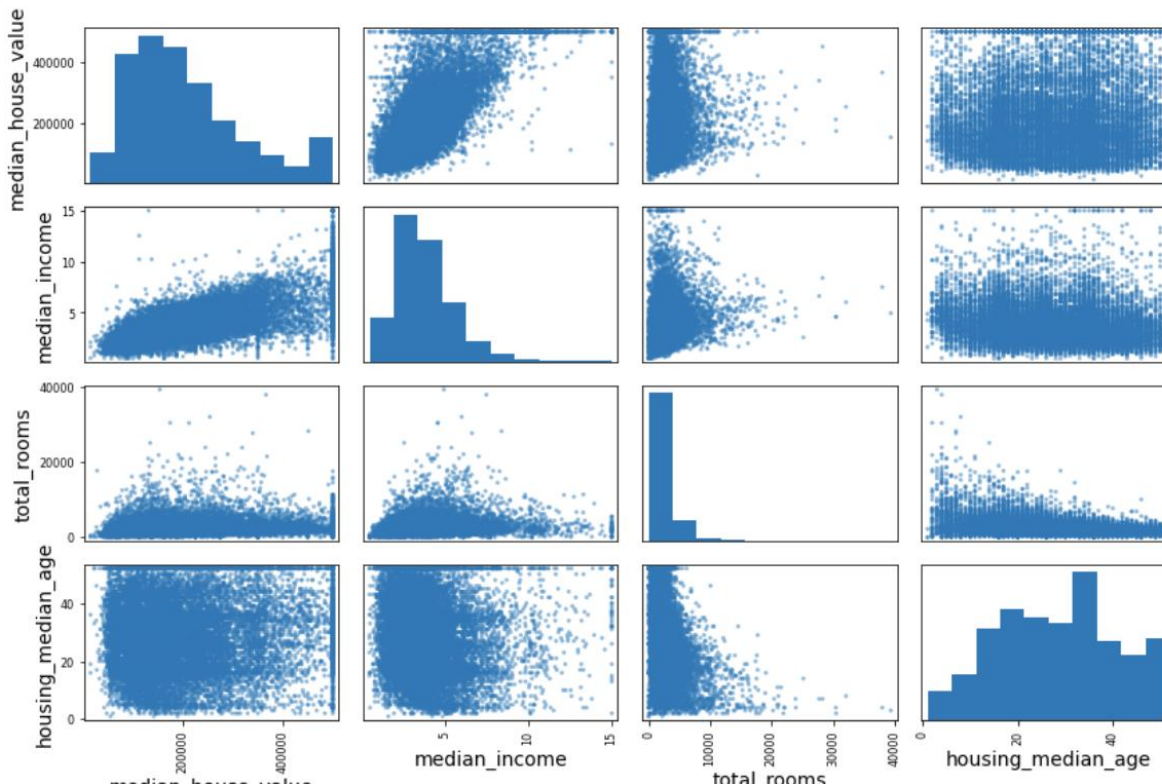
```
Out[39]: median_house_value    1.000000  
         median income        0.687160  
         total_rooms          0.135097  
         housing_median_age    0.114110  
         households            0.064506  
         total_bedrooms        0.047689  
         population            -0.026920  
         longitude             -0.047432  
         latitude              -0.142724  
         Name: median_house_value, dtype: float64
```

■ 각 특징 (열)끼리의 상관관계 시각화

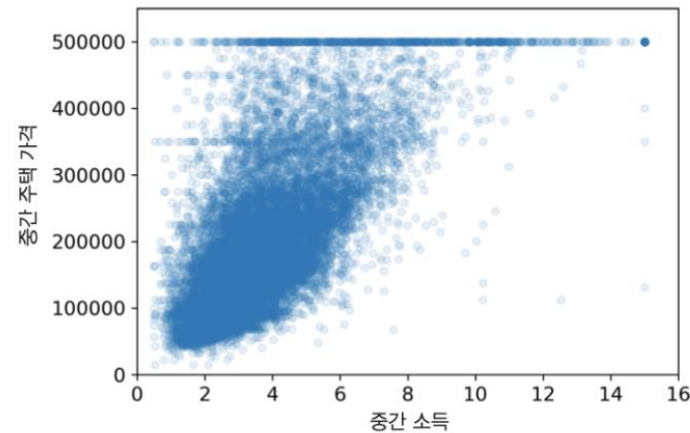
```
In [40]: # from pandas.tools.plotting import scatter_matrix # For older versions of Pandas
from pandas.plotting import scatter_matrix

attributes = ["median_house_value", "median_income", "total_rooms",
             "housing_median_age"]
scatter_matrix(housing[attributes], figsize=(12, 8))
save_fig("scatter_matrix_plot")
```

Saving figure scatter_matrix_plot



- 중간 주택 가격과 중간 소득의 관계: 산점도 활용
- 점들이 너무 넓게 퍼져 있음. 완벽한 선형관계와 거리 멀.
- 50만 달러 수평선: 가격 제한
- 35만, 28만, 그 아래 정도에서도 수평선 존재



- 4. (실습) Scikit-learn과 MNIST dataset을 이용한 간단한 이진 분류기 (핸즈온 챕터 3)

-
- Most common supervised learning tasks are **classification (predicting class)** and regression (predicting values)
 - MNIST dataset을 이용한 classification (분류기) 실습

-
- **MNIST dataset**
 - 미국 고등학생과 인구조사국 직원들이 손으로 쓴 70,000개의 숫자 이미지로 구성된 데이터셋
 - 사용된 0부터 9까지의 숫자는 각각 $28 \times 28 = 784$ 크기의 픽셀로 구성된 이미지 데이터
 - 2차원 어레이가 아닌 길이가 784인 1차원 어레이로 제공
 - Label: 총 70,000개의 사진 샘플이 표현하는 값

5	0	4	1	9	2	1	3	1	4
3	5	3	6	1	7	2	8	6	9
4	0	9	1	1	2	4	3	2	7
3	8	6	9	0	5	6	0	7	6
1	8	1	9	3	9	8	5	9	3
3	0	7	4	9	8	0	9	4	1
4	4	6	0	4	5	6	1	0	0
1	7	1	6	3	0	2	1	1	7
9	0	2	6	7	8	3	9	0	4
6	7	4	6	8	0	7	8	3	1

-
- 문제 정의
 - 지도학습: 각 이미지가 담고 있는 숫자가 레이블 (label)로 지정됨
 - 분류: 이미지 데이터를 분석하여 0부터 9까지의 숫자로 분류
 - 이미지 그림을 총 10개의 클래스로 (class)로 분류하는 multiclass classification

-
- Training data와 test data 나누기
 - MNIST dataset은 이미 6:1로 분류되어 있음
 - 훈련세트: 앞쪽 60,000개 이미지
 - 테스트세트: 나머지 10,000개 이미지

-
- https://github.com/hunni10/hand-on-ml2-for-share/blob/master/03_classification.ipynb
 - 교재 깃허브의 chapter 3

-
- **Setup**
 - **Matplotlib 설치: Python에서의 시각화 라이브러리**
 - **Python: 3.5이상 설치**
 - **Scikit-Learn: 0.20이상 버전**

-
- handson-ml2/03_classification.ipynb at master · ageron/handson-ml2 · GitHub
 - 최초 setup 실행후 오류가 뜨는지 확인

■ 1. Loading the dataset

```
In [2]: from sklearn.datasets import fetch_openml
mnist = fetch_openml('mnist_784', version=1, as_frame=False)
mnist.keys()
```

```
Out[2]: dict_keys(['data', 'target', 'frame', 'categories', 'feature_names', 'target_names', 'DESCR', 'details', 'url'])
```

- Scikit-learn을 사용해 MNIST dataset 가져옴
- ‘DESCR’ key: dataset 설명
- ‘target’ key: label이 들어있음

■ 2. Exploring data

```
In [3]: X, y = mnist["data"], mnist["target"]  
        X.shape
```

```
Out[3]: (70000, 784)
```

```
In [4]: y.shape
```

```
Out[4]: (70000,)
```

- 행렬의 차원을 shape으로 표현

- 차원수 return: 70,000행, 784열

즉, 70,000개의 data, 784 특징 (28x28 pixel)

- y 는 label을 의미
- $x[0]$ 는 5처럼 보임 실제 $x[0]$ 의 label 확인



`y[0]`

'5'

-
- Image의 픽셀값은 0-255사이의 정수로 표현. 이러한 자료형 (uint8)으로 변환

```
In [8]: y = y.astype(np.uint8)
```

■ 최초 100개의 image 시각화

In [11]:

```
plt.figure(figsize=(9,9))  
example_images = X[:100]  
plot_digits(example_images, images_per_row=10)  
save_fig("more_digits_plot")  
plt.show()
```



5	0	4	1	9	2	1	3	1	4
3	5	3	6	1	7	2	8	6	9
4	0	9	1	1	2	4	3	2	7
3	8	6	9	0	5	6	0	7	6
1	8	7	9	3	9	8	5	9	3
3	0	7	4	9	8	0	9	4	1
4	4	6	0	4	5	6	1	0	0
1	7	1	6	3	0	2	1	1	7
9	0	2	6	7	8	3	9	0	4
6	7	4	6	8	0	7	8	3	1

-
- **3. Splitting dataset into training and test dataset**
 - **MNIST dataset은 이미 training set (60,000개)과 test data (10,000개)로 구분되어 있음**

```
In [13]: X_train, X_test, y_train, y_test = X[:60000], X[60000:], y[:60000], y[60000:]
```

■ 4. Binary classification

■ 이진 분류기: “5” 인가? “5”가 아닌가?

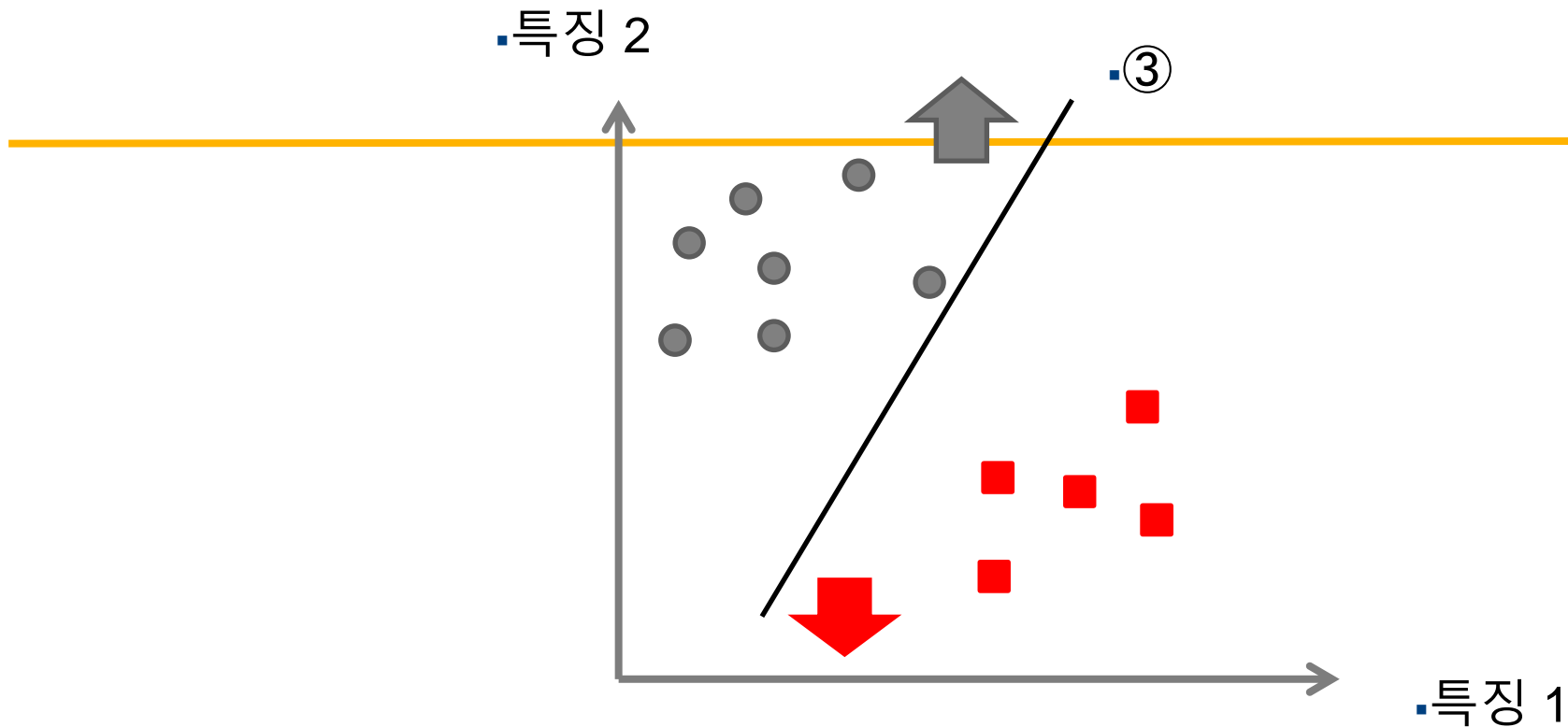
```
In [14]: y_train_5 = (y_train == 5)
          y_test_5 = (y_test == 5)
```

■ 분류기: stochastic gradient descent (SGD) classifier 사용 (Scikit-learn 제공)

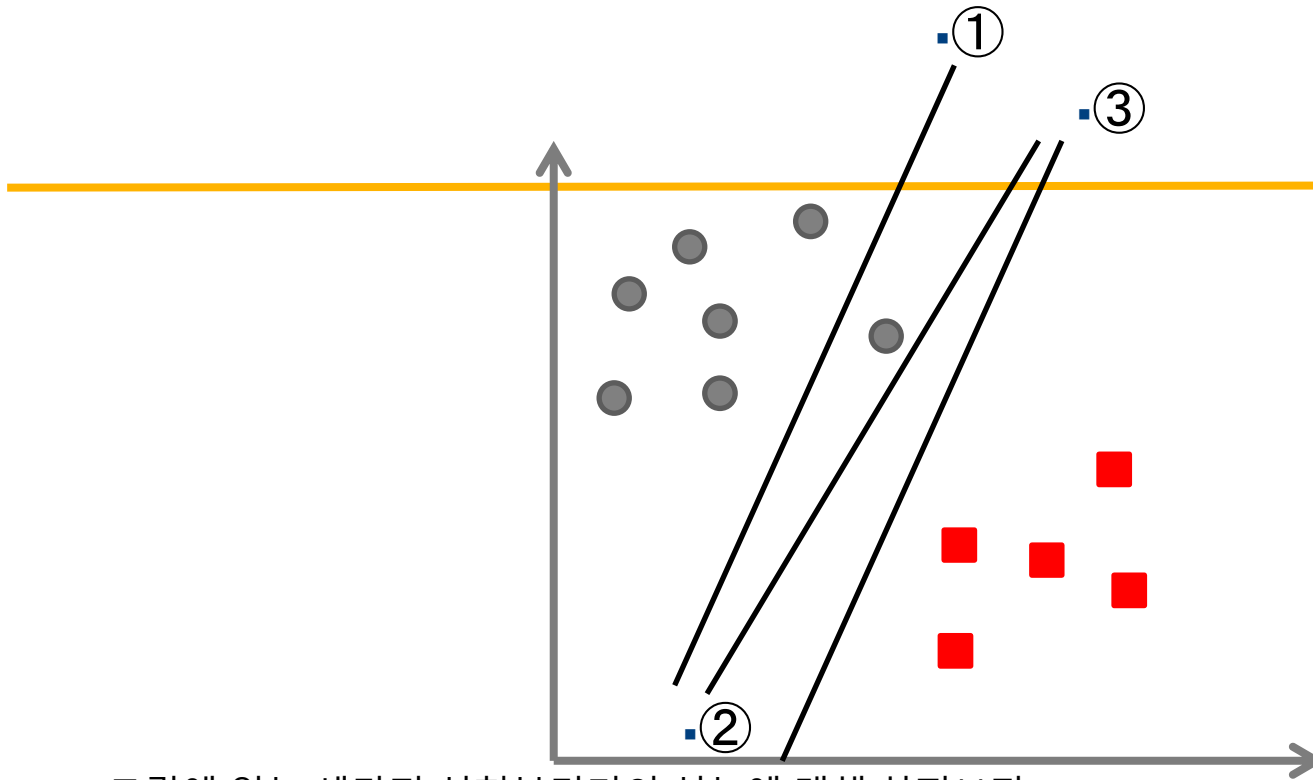
```
In [15]: from sklearn.linear_model import SGDClassifier

          sgd_clf = SGDClassifier(max_iter=1000, tol=1e-3, random_state=42)
          sgd_clf.fit(X_train, y_train_5)
```

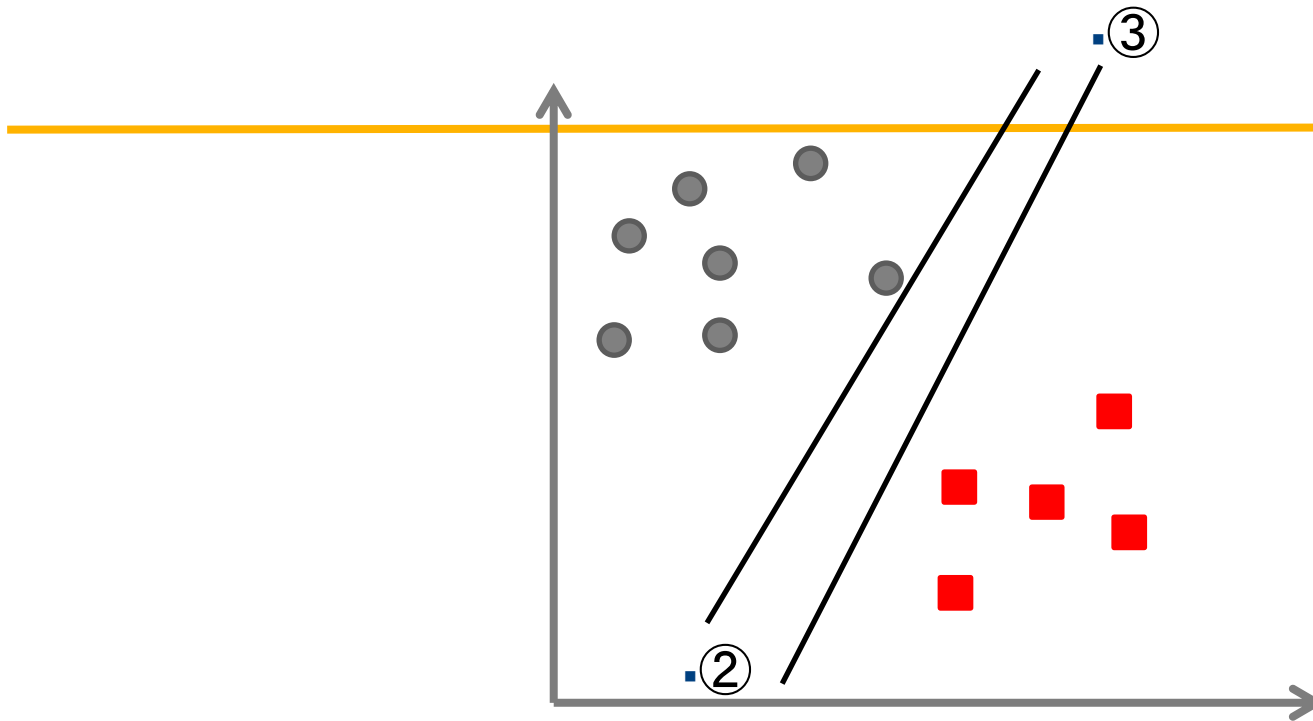
-
- 분류기 성능 평가: **K-fold Cross validation (CV)**
 - 왜 필요한가? 학습한 dataset에 평가까지 하면 왜곡된 결과가 나올 수 있음 (학습하지 않은 데이터에 test를 해보야함). **Overfitting 문제**
 - **한쪽 데이터에 치우친 결과를 방지하기 위하여**
 - **Splitting the training set into K-folds (in this case 3), then making predictions and evaluating them on each fold using a model trained on the remaining folds**



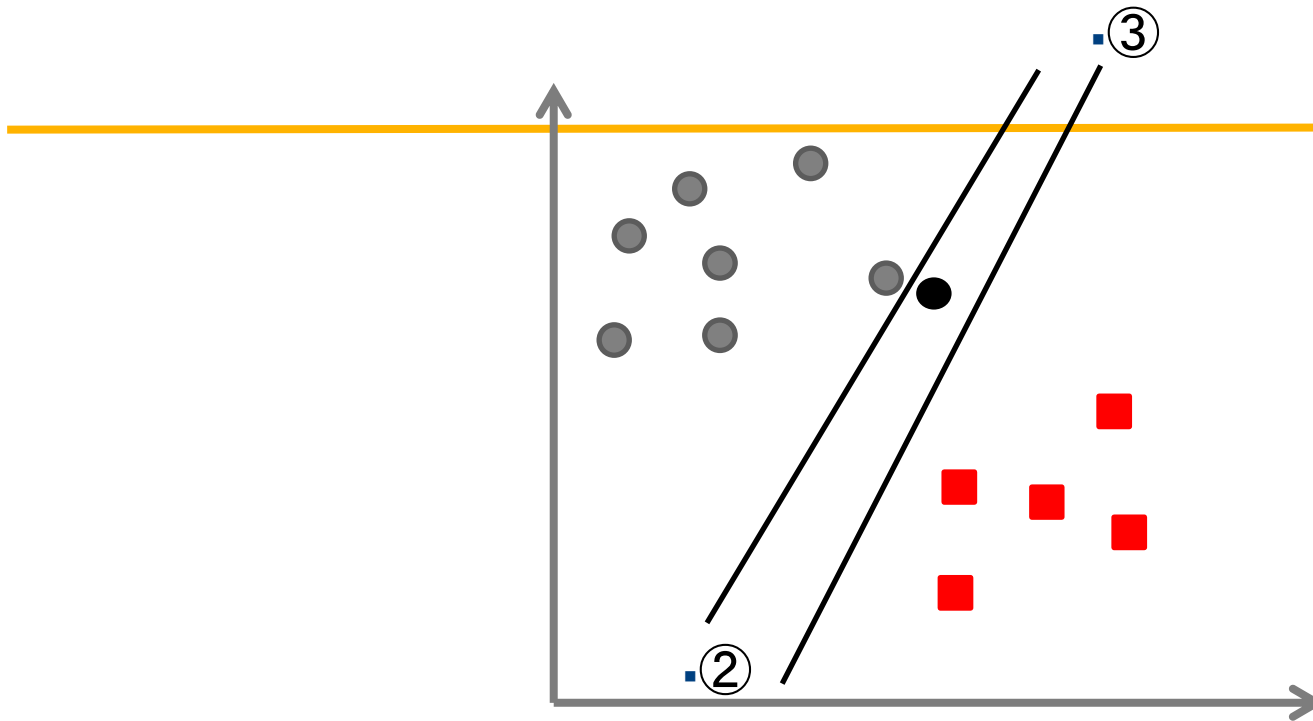
- 선형 분류기란 간단히 말하면 데이터를 직선으로 분류한다는 의미이다 (2D의 경우)
- 즉 이 직선을 기점으로 위쪽은 파란색원으로 분류
아래쪽은 빨간색 사각형으로 분류 한다
(영역을 두 부분으로 나눈다고 생각하면 된다)



- 그림에 있는 세가지 선형분리기의 성능에 대해 살펴보자
- 직선의 위쪽은 파란색원, 아래쪽은 빨간색 사각형으로 분류하고자 한다
- 여기서 샘플 데이터 들은 training data이다.
- 선형 분류기 1: 샘플 하나를 틀리게 분류하므로 좋지 않은 분류기이다
- 선형 분류기 2: 모든 샘플을 제대로 분리하였고 오류가 없다
- 선형 분류기 3: 모든 샘플을 제대로 분리하였고 오류가 없다



Q) 선형 분류기 2와 3은 성능이 같을까?



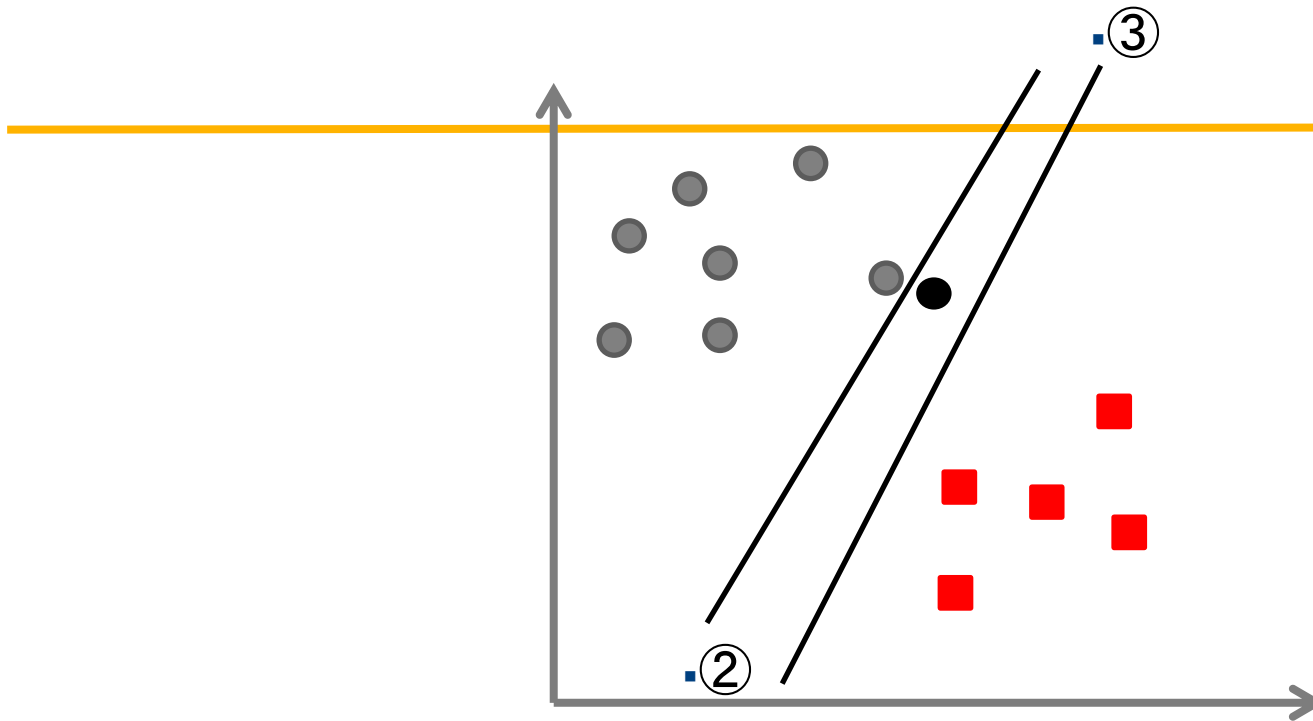
선형분류기 2와 3은 성능이 같지 않다.

machine learning에서 분류기의 목적은 미래의 데이터가 들어왔을때 이 데이터가 어떤 분류에 속하는지 예측하는 것이기 때문이다.

예를 들어 검정 데이터가 생성되었을때 선형분류기 2는 빨간색으로 분류 한다

선형분류기 3은 파란색으로 분류 한다

Q) 그렇다면 선형 분류기 2와 3중에 어떤것이 분류기로써 더 성능이 좋을까?

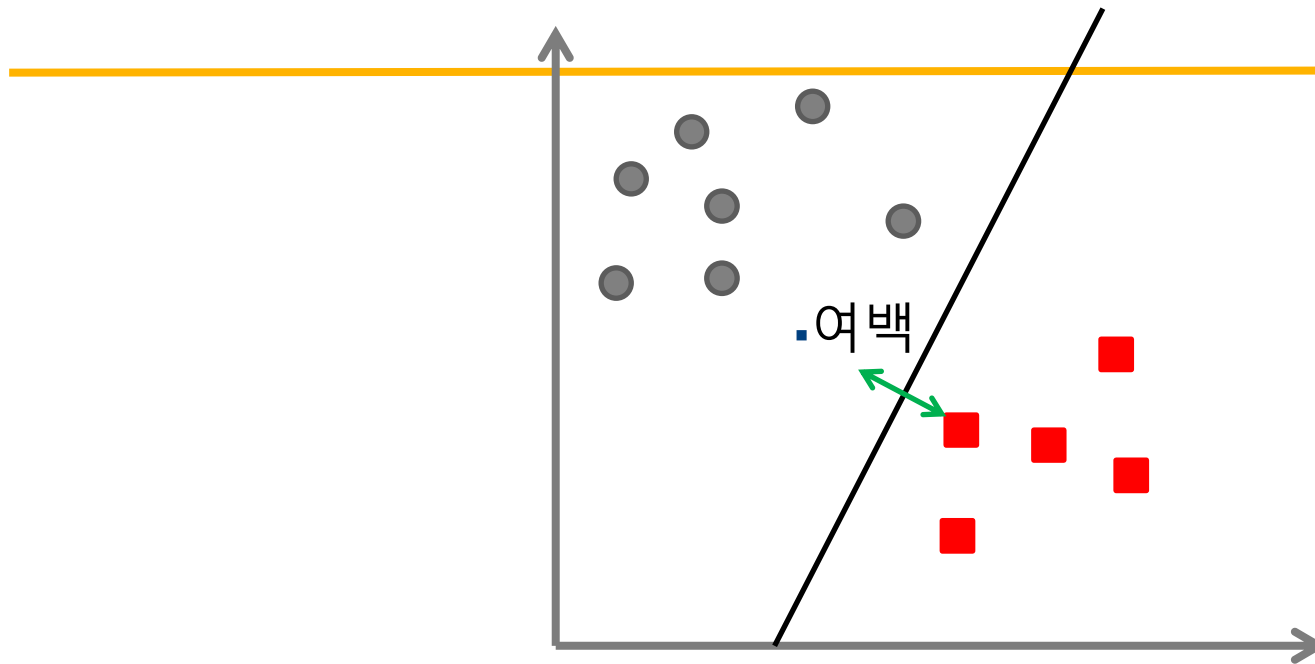


Q) 그렇다면 선형 분류기 2와 3중에 어떤것이 분류기로써 더 성능이 좋을까?

이 경우 검정색 데이터는 파란색에 가깝기 때문에 파란색으로 분류하는것이 더 타당하다고 추측할 수 있다 (빨간색 점들과는 멀다)

그러므로 분류기 2와 분류기 3중에 분류기 3이 더 좋다고 생각할 수 있다. 그 이유는 분류기 2는 파란색 패턴에서 조금만 변형이 있는 데이터가 들어와도 분류가 잘못 되기 때문이다.

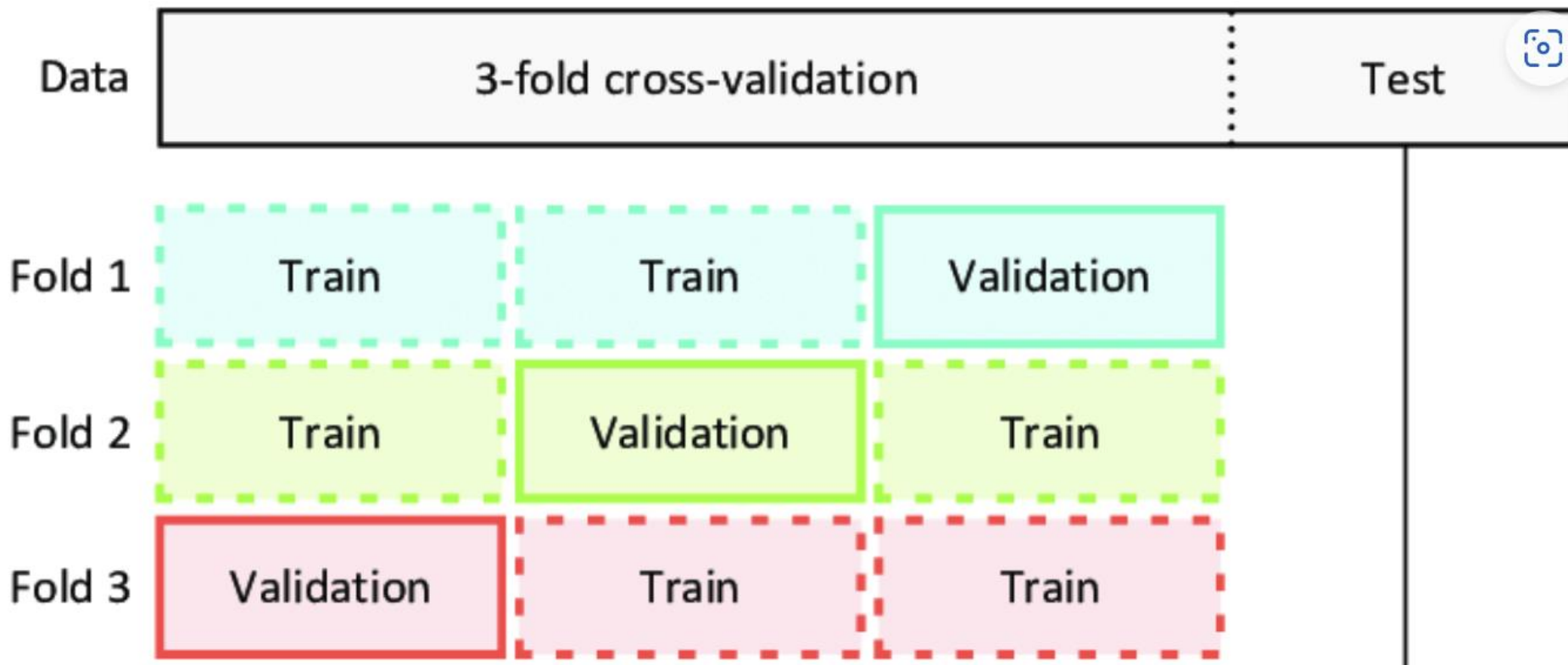
한편으로 분류기3이 좋은 이유는 이 선형 분류기 (3)에서 가장 가까운 점들 사이의 간격 (여백, margin) 이 크기 때문이다 라고 얘기할 수 있다.



- SVM에서 여백 (margin)이란 선형 분리기 (직선)으로부터 가장 가까운 샘플데이터까지의 거리의 2배 (위의 그림에서 초록색)라고 정의 된다
- 선형 SVM (SVC)의 목적은 여백을 가장 크게하는 직선을 찾는 것이다!

그렇다면 새로운 데이터가 들어왔을때 제대로 된 분류를 할 확률이 높다

- 모든 training data를 활용하여 검증
- 일부만 사용하여 왜곡된 결과를 방지하기 위함



- **Above 95% accuracy on all cross-validation folds**

```
In [17]: from sklearn.model_selection import cross_val_score  
cross_val_score(sgd_clf, X_train, y_train_5, cv=3, scoring="accuracy")
```

```
Out[17]: array([0.95035, 0.96035, 0.9604 ])
```

- **Good? Let's look at a very dumb classifier that just classifies every single image in the “not-5” class**

- It has over 90% accuracy? Why?
- Because only about 10% of the images are 5s, so if you always guess that an image is not a 5, you will be right about 90% of the time

In [19]:

```
from sklearn.base import BaseEstimator
class Never5Classifier(BaseEstimator):
    def fit(self, X, y=None):
        pass
    def predict(self, X):
        return np.zeros((len(X), 1), dtype=bool)
```

In [20]:

```
never_5_clf = Never5Classifier()
cross_val_score(never_5_clf, X_train, y_train_5, cv=3, scoring="accuracy")
```

Out[20]: array([0.91125, 0.90855, 0.90915])

-
- This demonstrates why accuracy is generally not the preferred performance measure for classifiers
 - 오차 행렬 (confusion matrix)
 - 클래스별 예측 결과를 정리한 행렬
 - 오차행렬의 행은 실제 class를 열린 예측된 class를 가리킴

- The first row of this matrix considers non-5 images: 53,892 of them were correctly classified as non-5s, while 687 were wrongly classified as 5s
- 1,891 were wrongly classified as non-5s, 3,530 were correctly classified as 5s

```
In [21]: from sklearn.model_selection import cross_val_predict  
  
y_train_pred = cross_val_predict(sgd_clf, X_train, y_train_5, cv=3)
```

```
In [22]: from sklearn.metrics import confusion_matrix  
  
confusion_matrix(y_train_5, y_train_pred)
```

```
Out[22]: array([[53892,   687],  
               [ 1891,  3530]])
```

- 완벽한 분류기
- 오차 행렬의 대각부분 (diagonal)에만 값이 있음

```
In [23]: y_train_perfect_predictions = y_train_5 # pretend we reached perfection  
confusion_matrix(y_train_5, y_train_perfect_predictions)
```

```
Out[23]: array([[54579,    0],  
               [    0, 5421]])
```

■ 5. 군집화

•Scikit-learn의 machine-learning 방법 분류

•군집화 (clustering)

•군집화는 매우 다양한 응용 분야를 가지고 있다

Classification

Identifying to which category an object belongs to.

Applications: Spam detection, Image recognition.

Algorithms: *SVM, nearest neighbors, random forest, ...* — Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.

Algorithms: *SVR, ridge regression, Lasso, ...* — Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes

Algorithms: *k-Means, spectral clustering, mean-shift, ...* — Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency

Algorithms: *PCA, feature selection, non-negative matrix factorization.* — Examples

Model selection

Comparing, validating and choosing parameters and models.

Goal: Improved accuracy via parameter tuning

Modules: *grid search, cross validation, metrics.* — Examples

Preprocessing

Feature extraction and normalization.

Application: Transforming input data such as text for use with machine learning algorithms.

Modules: *preprocessing, feature extraction.* — Examples

■ 군집화 사용 예시

· 어떤 사람이 몇년간 쇼핑몰을 성공적으로 운영해 왔다 (수백만명의 고객을 가지고 있다). 그 동안 한 종류의 홍보 판플렛을 만들어 발송했는데 이제부터는 고객의 취향을 분석하여 4-6종류의 판플렛을 만들어 맞춤 홍보를 하고자 한다. 일종의 개인화 (personalization) 홍보 전략이다.
· 고객에 대한 각종 정보는 DB에 저장되어 있어 이것을 기초 자료로 활용 가능하다

· 고객 정보는 월평균 구매액, 선호하는 물품의 종류와 수준, 결제 방법, 반품 성향, 직업, 성별, 나이, 거주 지역등 다양하다.

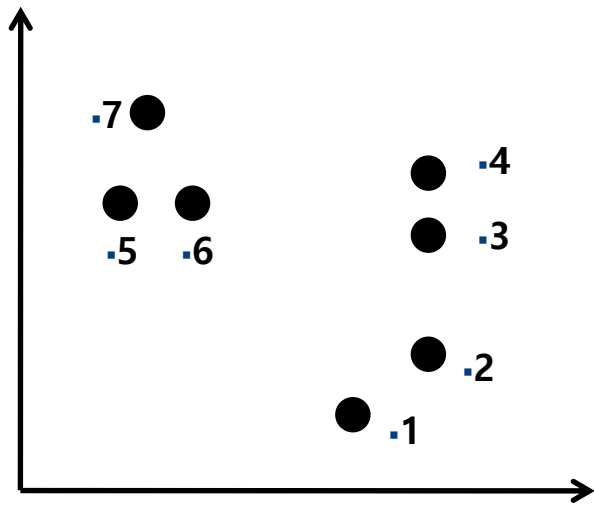
· Q) 수백만명의 고객이 있을때 이를 어떻게 4-6개의 그룹으로 나눌수 있을까?

-
- 해결 방법: 유사한 샘플(데이터)들끼리 모아
서 4-5개의 그룹으로 만든다
 - 여기서 유사하다는 것은 데이터와의 거리가
가깝다는 것을 의미 한다
 - 이 각각의 그룹을 군집 (cluster)라고 하고
이 경우에는 4-5개의 군집이 만들어 진다
-

.여태까지 배운 Supervised learning과 다르게 오늘 배울 군집화 (clustering)은 unsupervised learning으로써 데이터 정보만 있을 뿐 이 데이터의 분류는 주어지지 않는다

K-means 알고리즘

• k (군집개수)=3이고 7개의 데이터가 있는 경우를 생각해 보자



1. x_1, x_2, x_3 를 초기 군집 중심으로 삼고

• 이를 z_1, z_2, z_3 라고 놓는다

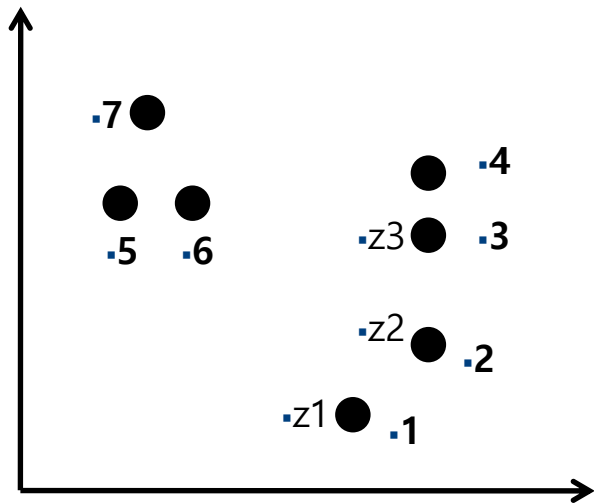
• $z_1 = (18, 5)$

• $z_2 = (20, 9)$

• $z_3 = (20, 14)$

• $x_1 = (18, 5), x_2 = (20, 9), x_3 = (20, 14), x_4 = (20, 17), x_5 = (5, 15), x_6 = (9, 15), x_7 = (6, 20)$

• k (군집개수)=3이고 7개의 데이터가 있는 경우를 생각해 보자



• 2. 각 데이터들은 $z1, z2, z3$ 중

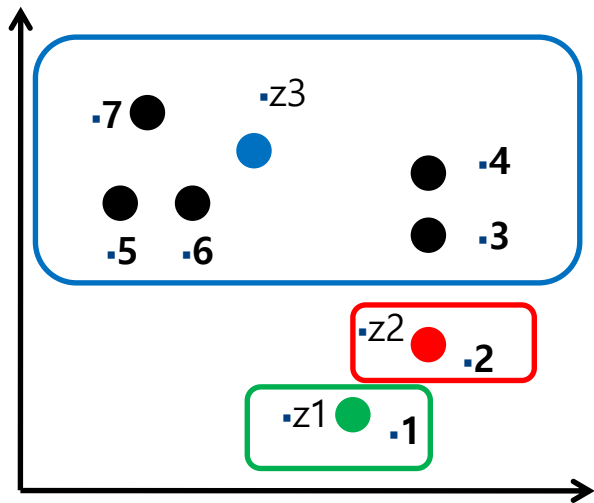
• 가장 가까운 군집 중심에 각각 배정된다

• $\{x1\}$ 은 $z1$

• $\{x2\}$ 은 $z2$

• $\{x3, x4, x5, x6, x7\}$ 은 $z3$ 에 배정된다

• k (군집개수)=3이고 7개의 데이터가 있는 경우를 생각해 보자



• 2. $\{x_1\}$ 은 z_1

• $\{x_2\}$ 은 z_2

• $\{x_3, x_4, x_5, x_6, x_7\}$ 은 z_3 에 배정된다

• 3. 2에서 배정된 데이터들을 이용해 새로운

• 군집 중심을 찾고 (위치 평균)

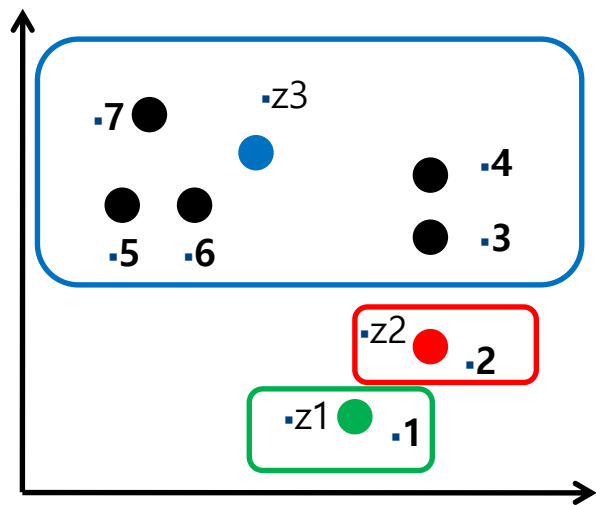
• 이를 새로운 z_1, z_2, z_3 라 놓는다

• $z_1 = (18, 5)$

• $z_2 = (20, 9)$

• $z_3 = (x_3 + x_4 + x_5 + x_6 + x_7) / 5 = (12, 16.2)$

• k (군집개수)=3이고 7개의 데이터가 있는 경우를 생각해 보자



• 앞의 step 2를 반복한다. 즉

• 각 데이터들은 가장 가까운 군집 중심에

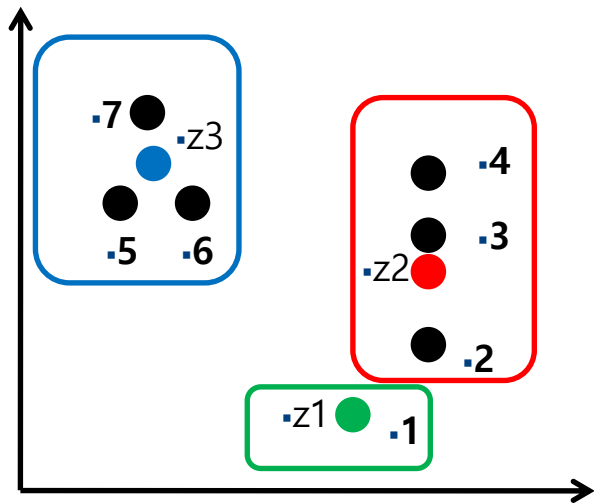
• 각각 배정된다

• $\{x_1\}$ 은 z_1

• $\{x_2, x_3, x_4\}$ 는 z_2

• $\{x_5, x_6, x_7\}$ 은 z_3 에 배정된다

• k (군집개수)=3이고 7개의 데이터가 있는 경우를 생각해 보자



• $\{x_1\}$ 은 z_1

• $\{x_2, x_3, x_4\}$ 는 z_2

• $\{x_5, x_6, x_7\}$ 은 z_3 에 배정된다

• 앞의 step 3를 반복한다. 즉

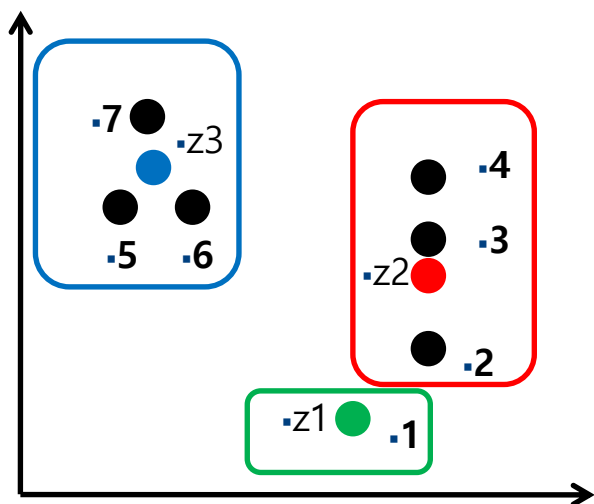
• 배정된 데이터들의 군집 중심을 찾고

• 이를 새로운 z_1, z_2, z_3 라 놓는다

• $z_1 = (18, 5)$, $z_2 = (x_2 + x_3 + x_4)/3 = (20, 13.333)$

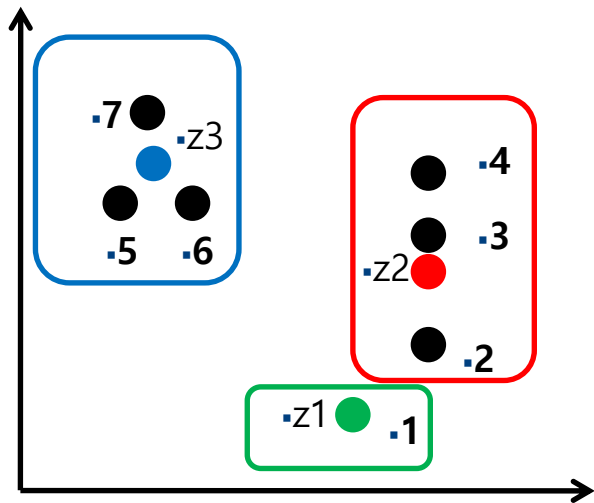
• $z_3 = (x_5 + x_6 + x_7)/3 = (6.667, 16.667)$

• k (군집개수)=3이고 7개의 데이터가 있는 경우를 생각해 보자



- 그 다음에 step 2와 3를 반복한다
- Step 2: 각 데이터들은 가장 가까운 군집 중심에 각각 배정된다
- 변화가 있는가? 없음
- Step 3: 배정된 데이터들의 각각의 군집 중심을 찾고 (위치 평균)
- 이를 새로운 z_1, z_2, z_3 라 놓는다
- 따라서 step3의 변화도 없음

• k (군집개수)=3이고 7개의 데이터가 있는 경우를 생각해 보자



• 즉 최종적으로 왼쪽의 데이터들을

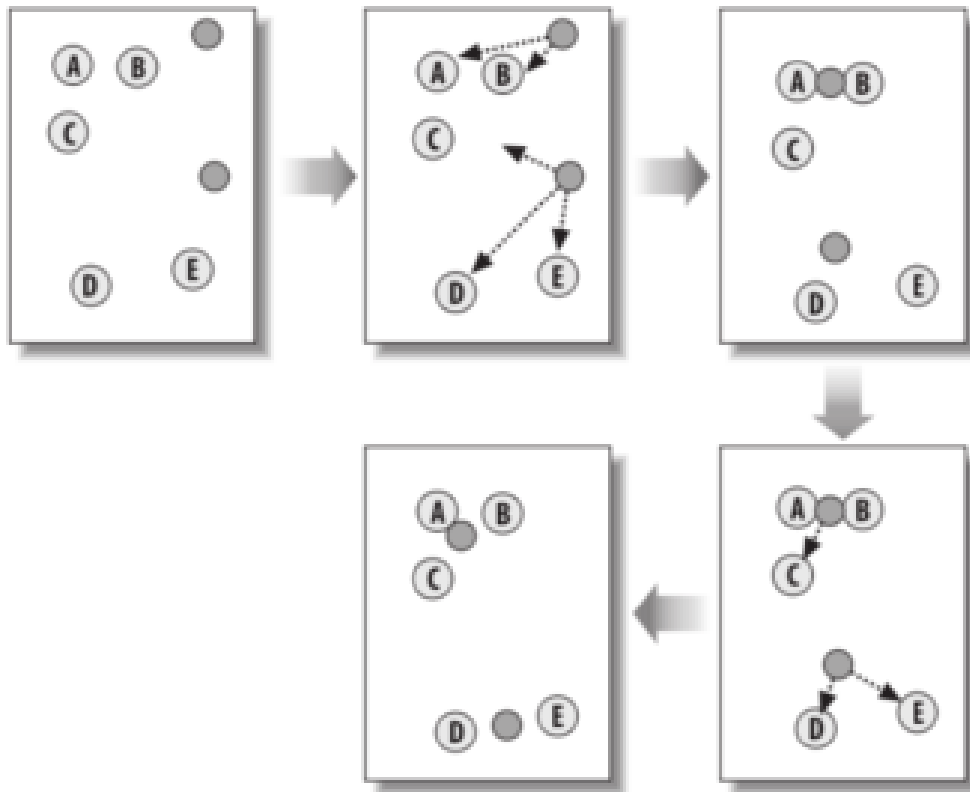
• $K=3$ (군집개수=3)으로 군집화 한

• 경우

• $\{x_1\}$, $\{x_2, x_3, x_4\}$, $\{x_5, x_6, x_7\}$ 과 같이

• 세 개의 군집으로 군집화 할 수 있다

K-means 알고리즘



•다음은 k-means 알고리즘의 Pseudo-code 이다

•입력: 데이터 $X=\{x_1, x_2, \dots, x_N\}$, 군집 개수= k

•출력: 군집화된 결과

•1. k 개의 군집 중심 $Z=\{z_1, z_2, \dots, z_k\}$ 를 초기화 한다

•2. while (1) {

• for ($i=1$ to N) x_i 를 가장 가까운 군집 중심에 배정

• if (이 배정이 이전 루프의 배정과 같음) break;

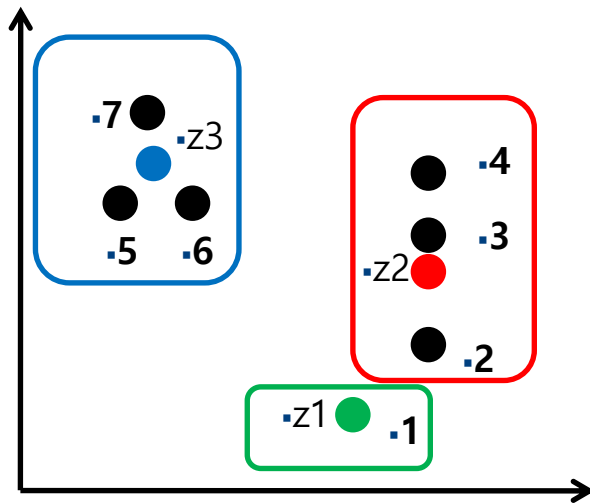
• for ($j=1$ to k) z_j 에 배정된 데이터의 평균으로 z_j 바꿈

- SSE와 Elbow method

-
- 군집화가 잘 되었다라는것은 어떻게 판단하나?
 - K-means 에서 군집화가 잘 되어 있다 라는건 데이터들이 군집 중심 근처에 모여 있는 것이다. 데이터 들이 군집내에서 퍼져 있는건 군집화가 잘되지 않았다고 생각할 수 있다
 - 군집화가 잘되었나 판단하는 척도 (metric) 중 하나는
 - 각각의 데이터가 그것에 가장 가까운 군집 중심 사이의 거리 (혹은 거리의 제곱)을 모두 더해 보는 것이다
 - 이를 Sum of squared error (SSE)라고도 한다
-

예를 들어 마지막 최종 군집화 결과의 경우

$\{x_1\} \Rightarrow z_1$, $\{x_2, x_3, x_4\} \Rightarrow z_2$, $\{x_5, x_6, x_7\} \Rightarrow z_3$



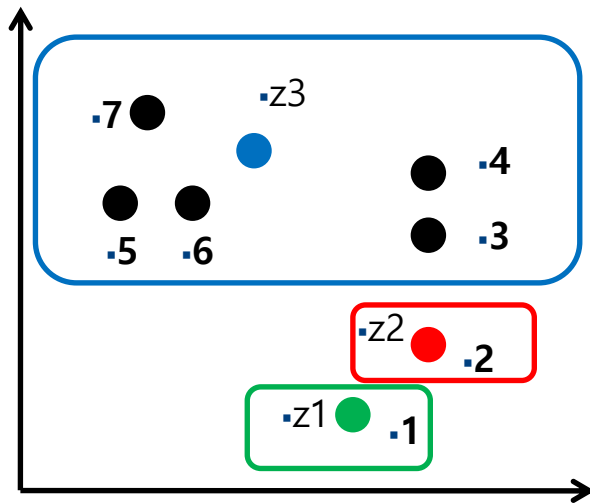
•예를 들어 마지막 최종 군집화 결과의 경우

• $\{x_1\} \Rightarrow z_1$, $\{x_2, x_3, x_4\} \Rightarrow z_2$, $\{x_5, x_6, x_7\} \Rightarrow z_3$

•SSE: 제곱 오류

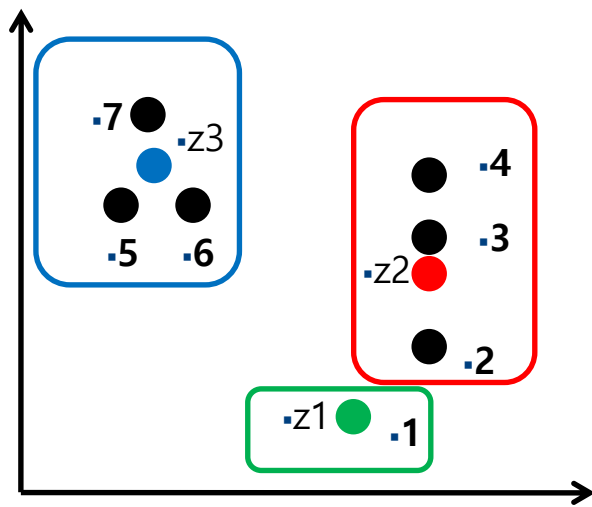
•SSE=58

• 최종 군집화 바로 이전의 경우

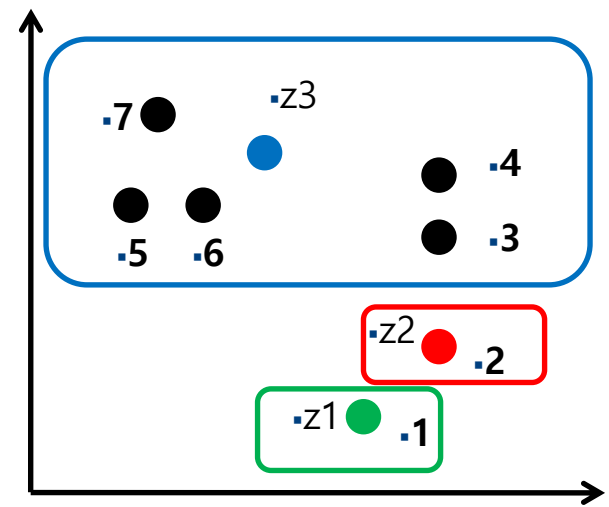


• $\{x_1\} \Rightarrow z_1, \{x_2\} \Rightarrow z_2, \{x_3, x_4, x_5, x_6, x_7\} \Rightarrow z_3$

• $SSE=244.8 \Rightarrow$ 즉 최종 군집화된 결과가 조금 오류가 더 적다!



·J=58.0



·J=244.8

·왼쪽 처럼 군집화한 것이 오류가 더 적다고 얘기할 수 있다.

-
- Iris dataset을 이용한 군집화 예제

■ Iris dataset

총 150개의 데이터

데이터설명 : 아이리스(붓꽃) 데이터에 대한 데이터이다. 꽃잎의 각 부분의 너비와 길이등을 측정한 데이터이며 150개의 레코드로 구성되어 있다. 아이리스 꽃은 아래의 그림과 같다. 프랑스의 국화라고 한다.



필드의 이해 :

데이터의 이해를 돕기 위해 포함된 6개의 변수에 대하여 간략하게 설명한다.

총 6개의 필드로 구성되어있다. caseno는 단지 순서를 표시하므로 분석에서 당연히 제외한다.

2번째부터 5번째의 4개의 필드는 입력 변수로 사용되고, 맨 아래의 Species 속성이 목표(종속) 변수로 사용된다.

caseno	일련번호이다. (1부터 150까지 입력된다.)
Sepal Length	꽃받침의 길이 정보이다.
Sepal Width	꽃받침의 너비 정보이다.
Petal Length	꽃잎의 길이 정보이다.
Petal Width	꽃잎의 너비 정보이다.
Species	꽃의 종류 정보이다. setosa / versicolor / virginica 의 3종류로 구분된다.

```
from sklearn.datasets import load_iris
```

```
iris = load_iris()  
iris
```

This code gives:

```
{'data': array([[5.1, 3.5, 1.4, 0.2],  
                [4.9, 3. , 1.4, 0.2],  
                [4.7, 3.2, 1.3, 0.2],  
                [4.6, 3.1, 1.5, 0.2],...  
'target': array([0, 0, 0, ... 1, 1, 1, ... 2, 2, 2, ...  
'target_names': array(['setosa', 'versicolor', 'virginica'], dtype='<U10'),  
...}]
```

실습