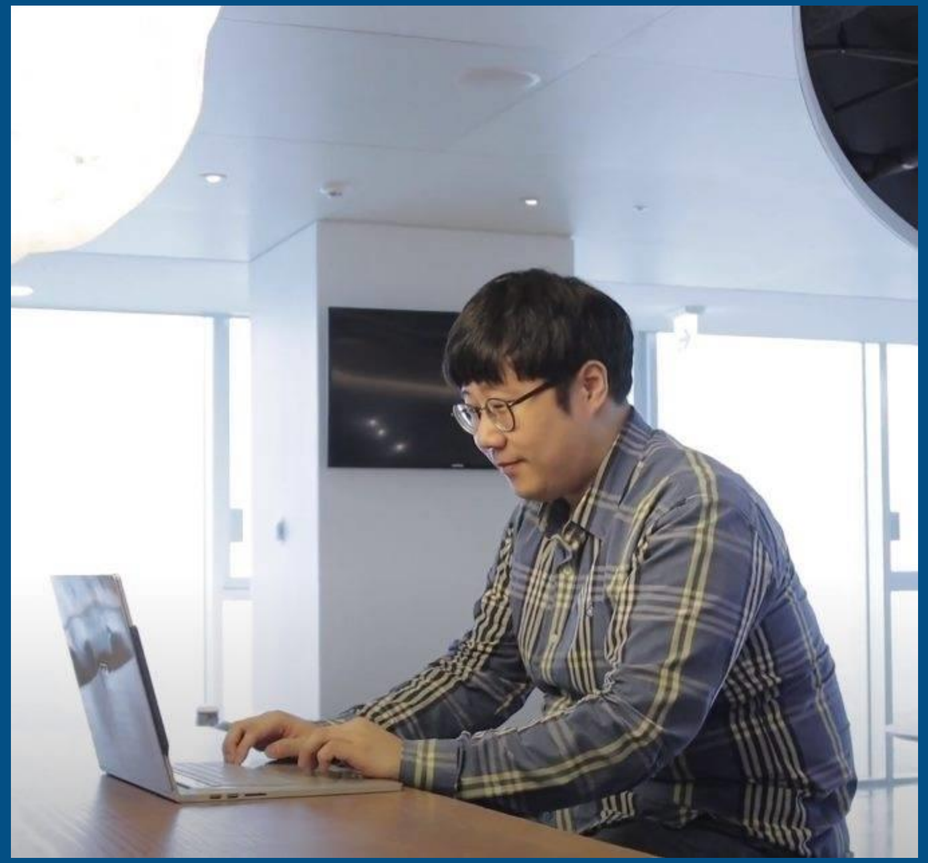
KAIST Include 동아리 스터디

AlphaGo와 AlphaGo Zero를 만들며 익히는 딥러닝 및 강화학습

Chris Ohk
utilForever@gmail.com

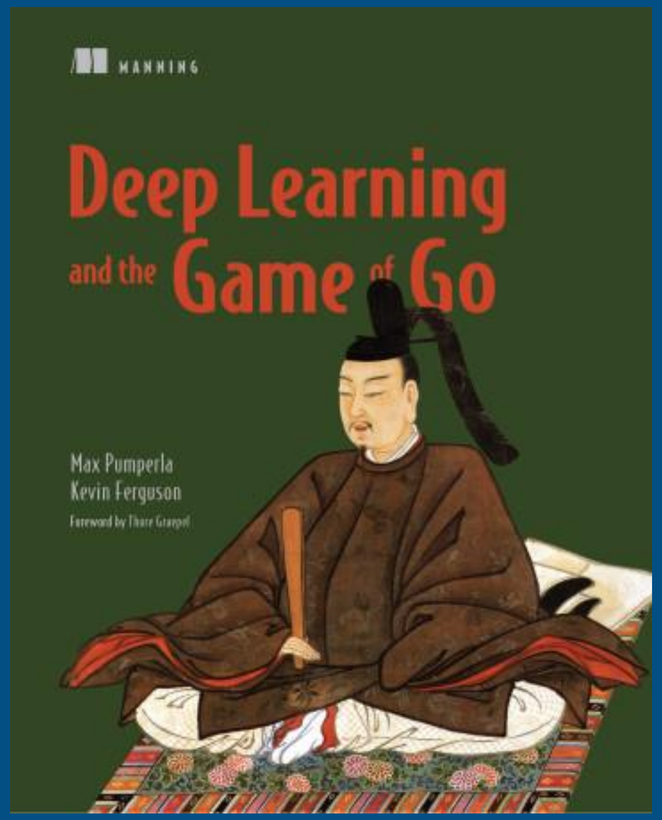
강사 소개

- 옥찬호 (Chris Ohk)
  - 경북대학교 컴퓨터공학과 학사 (08)
  - KAIST 전산학부 석사 (13)
  - 넥슨 코리아 게임 프로그래머
  - Microsoft Developer Technologies MVP
  - C++ Korea, Reinforcement Learning KR 관리자
  - IT 전문서 집필 및 번역 다수
    - 게임샐러드로 코드 한 줄 없이 게임 만들기 (2013)
    - 유니티 Shader와 Effect 제작 (2014)
    - 2D 게임 프로그래밍 (2014), 러스트 핵심 노트 (2017)
    - 모던 C++ 입문 (2017), C++ 최적화 (2019)



교재

- 주 교재
  - 영어 : Deep Learning and the Game of Go, by Max Pumperla and Kevin Ferguson (Manning, 2019)
  - 한국어 : 딥러닝과 바둑 (한빛미디어, 2020)
- 부 교재
  - 알파고를 분석하며 배우는 인공지능 (제이펍, 2019)
  - 알파제로를 분석하며 배우는 인공지능 (제이펍, 2019)



스터디 진행 계획

- Week 1 (3/20)
  - Toward deep learning: a machine-learning introduction
  - Go as a machine-learning problem
- Week 2 (3/27)
  - Implementing your first Go bot
- Week 3 (4/3)
  - Playing games with tree search
- Week 4 (4/10) & Week 5 (4/17)
  - No lecture due to mid-term exam

스터디 진행 계획

- Week 6 (4/24)
  - Getting started with neural networks
- Week 7 (5/1)
  - Designing a neural network for Go data
- Week 8 (5/8)
  - Learning from data: a deep learning bot
- Week 9 (5/15)
  - Learning by practice: reinforcement learning

스터디 진행 계획

- Week 10 (5/22)
  - Reinforcement learning with policy gradients
  - Reinforcement learning with value methods
- Week 11 (5/29)
  - Reinforcement learning with actor-critic methods
- Week 12 (6/5) & Week 13 (6/12)
  - No lecture due to final-term exam

스터디 진행 계획

- Week 14 (6/19)
  - AlphaGo: Bringing it all together
- Week 15 (6/26)
  - AlphaGo Zero: Integrating tree search with reinforcement learning

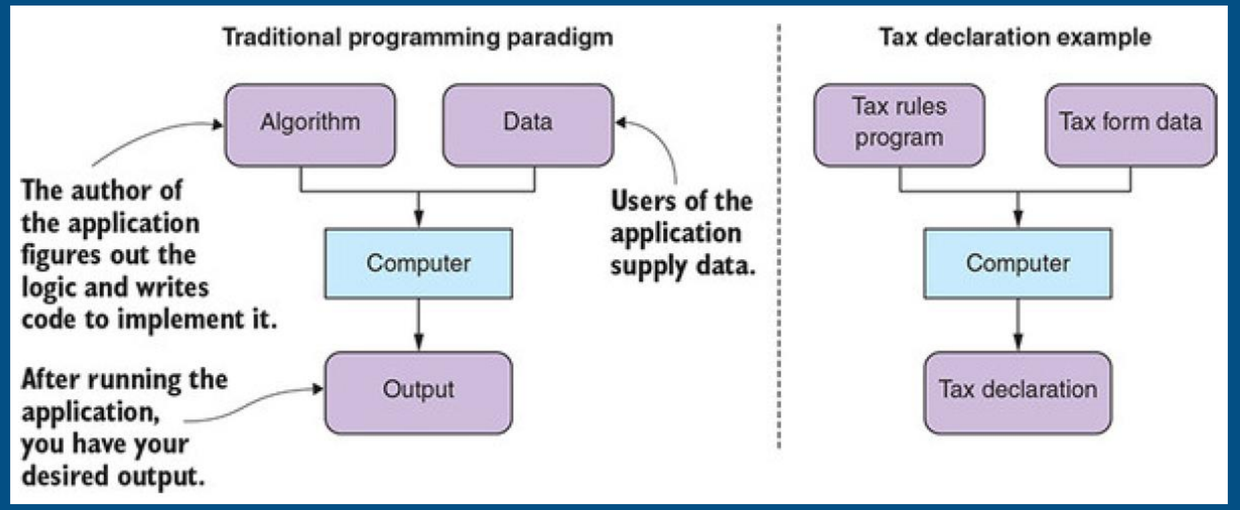
머신러닝 기초

- 머신러닝이란 무엇인가
  - 컴퓨터 프로그래밍 : 구조화된 데이터에 명확한 규칙을 적용
  - 머신러닝 : 규칙을 직접 구현하는 대신 예제 데이터를 사용해서 프로그램이나 알고리즘이 추론하는 방식의 기술

→ 머신러닝에서도 여전히 컴퓨터에 데이터를 입력해야 하지만 규칙을 부여하고 결과를 예상하는 대신 **예상 결과를 넣고 컴퓨터가 자체적으로 구체적인 알고리즘을 찾아내도록 한다**.

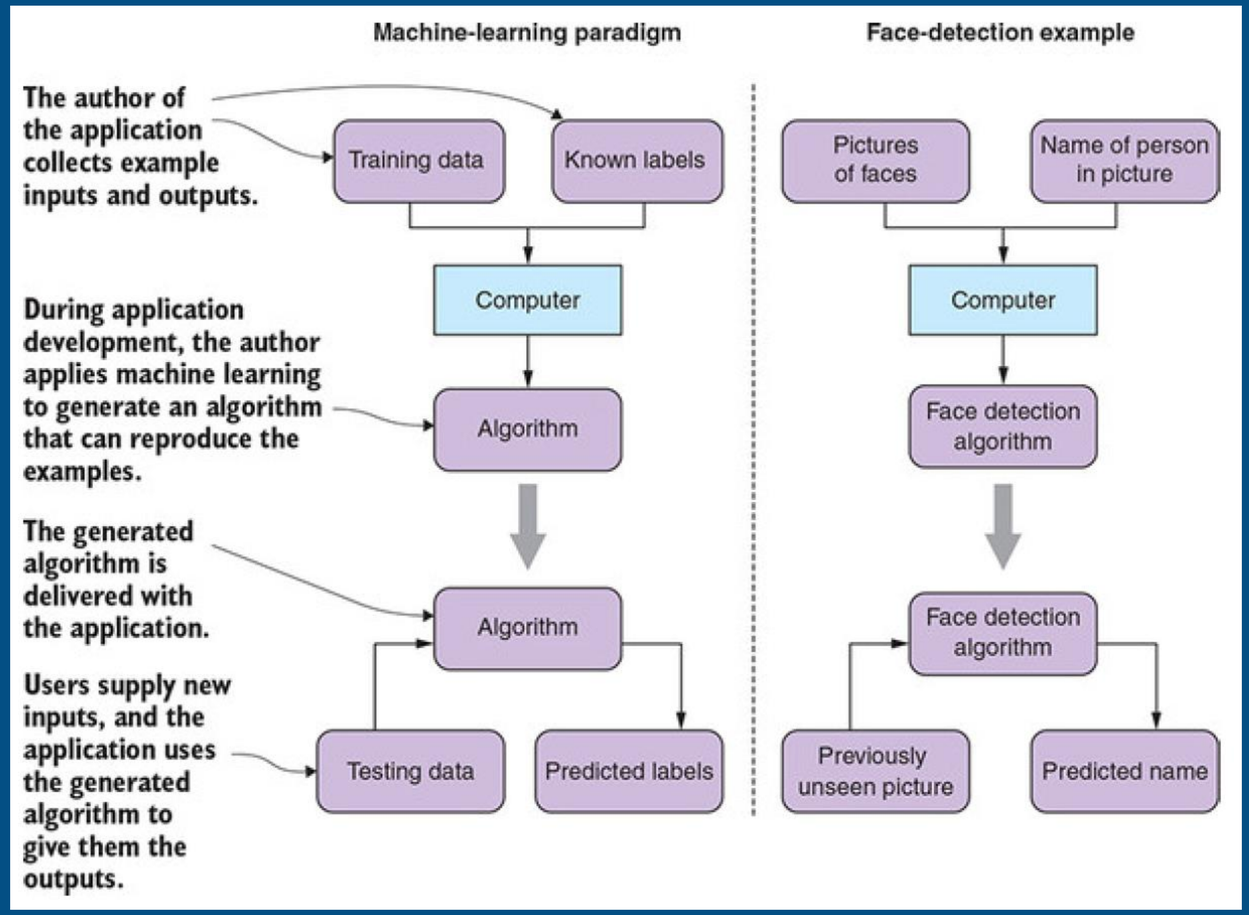
머신러닝 기초

- 머신러닝이란 무엇인가



머신러닝 기초

- 머신러닝이란 무엇인가



머신러닝 기초

- 머신러닝은 AI와 어떤 연관성이 있는가
  - 인공지능이란 컴퓨터가 사람의 행동을 모방하는 모든 기술을 의미
    - **논리 생성 시스템** : 정형적인 논리를 사용해서 명령에 대해 평가
    - **전문가 시스템** : 개발자가 사람의 지식을 소프트웨어로 직접 인코딩할 수 있도록 함
    - **퍼지 논리** : 컴퓨터가 모호한 명령을 처리할 수 있는 알고리즘을 정의
  - 위와 같은 규칙 기반 기술을 **고전 AI**라고 한다.

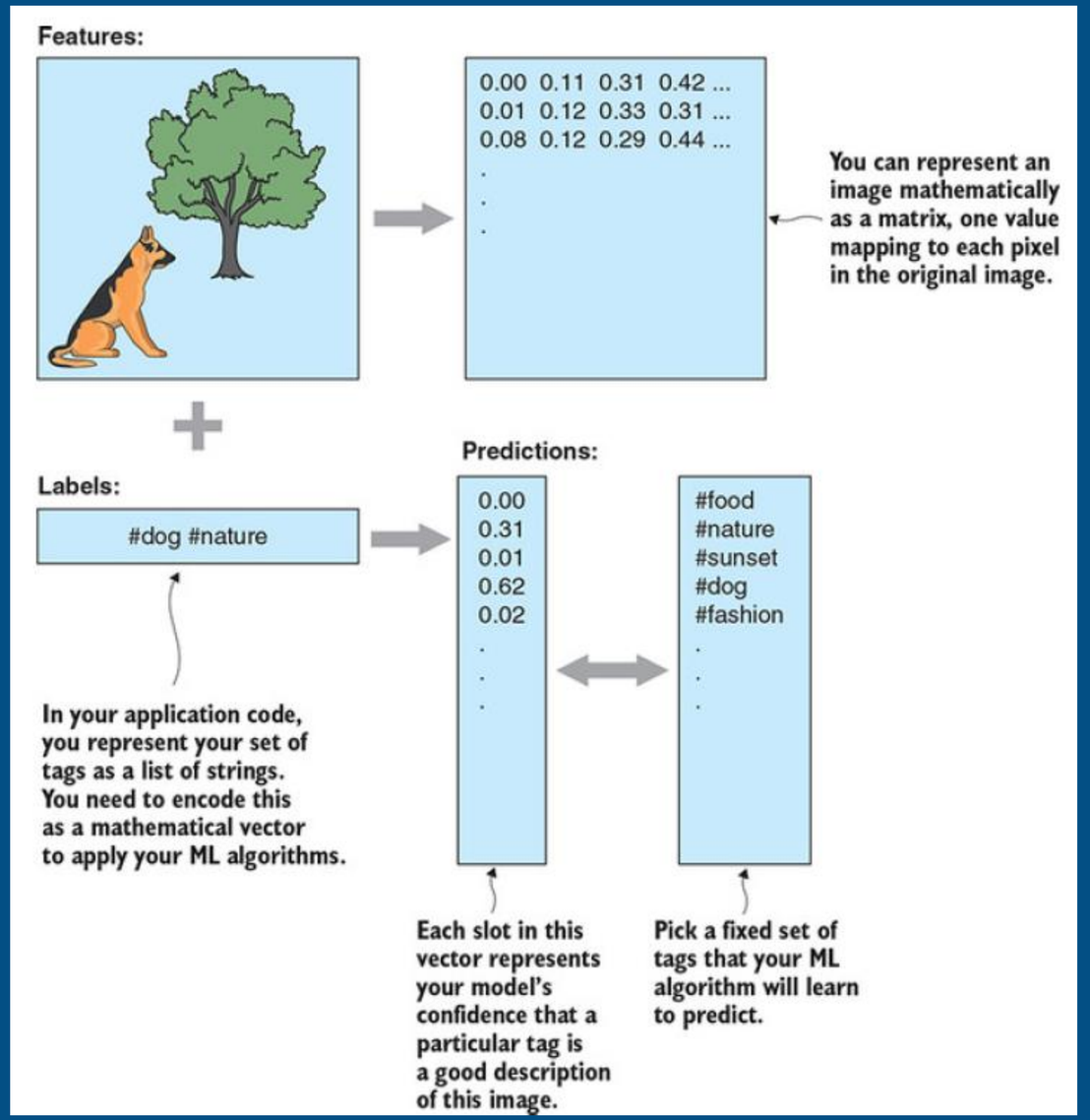
→ AI는 컴퓨터로 인간의 행동을 흉내 내는 일반적인 문제를 말하고, 머신러닝이나 딥러닝은 사례를 통해 알고리즘을 추출하는 수학적 기법을 말한다.

머신러닝 기초

- 머신러닝으로 할 수 있는 것과 할 수 없는 것
  - 전통적인 프로그래밍 방식을 선호하는 경우
    - 전통적인 알고리즘은 모든 문제를 직접적으로 해결한다.
    - 정확도가 완벽해야 한다.
    - 경험적으로 만들어진 간단한 규칙을 직접 넣는 건 유용하다.

머신러닝 기초

- 사례로 보는 머신러닝
  - 사용자들이 수백만 개의 태그가 달린 사진을 올리는 사진 공유 애플리케이션을 만들고 있다고 가정하자. 여기에 새 사진에 자동으로 태그를 추천해주는 기능을 추가하려 한다.
  - 데이터 전처리 단계는 모든 머신러닝 시스템에서 필수적인 부분이다. 보통 원래의 데이터를 가져와서 **특징** 생성(머신러닝 알고리즘에 사용할 입력 데이터)을 위한 전처리 단계를 거친다.

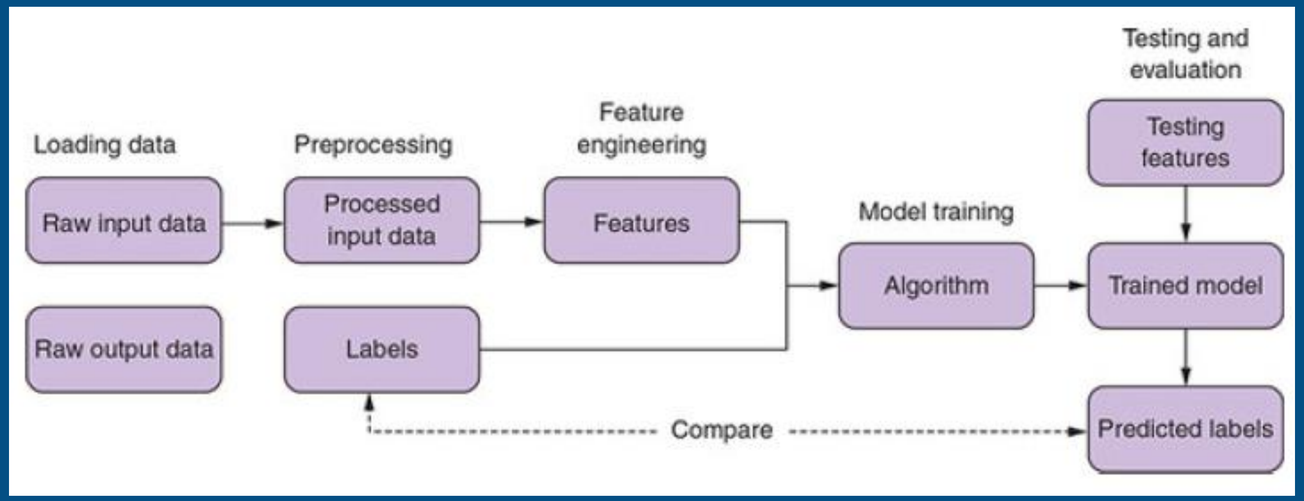


머신러닝 기초

- 지도학습
  - 이제 모델을 훈련할 알고리즘이 필요하다. 예제에서는 이미 정확하게 분류된 사진이 많다
  - 사용자들이 이미 애플리케이션에 수동으로 태그를 달고 사진을 올려뒀다. 이 예제로부터 가능한 한 태그와 이미지를 맞추는 함수를 학습시키고, 이 함수에 새 사진을 넣었을 때 합리적인 태그가 생성되도록 일반화된 형태를 만들어 보자. → **지도 학습 (Supervised Learning)**
  - 지도 학습이라고 부르는 이유는 사람이 사전에 생성한 예제의 **라벨**이 훈련 과정에 대한 지침을 제공하기 때문.

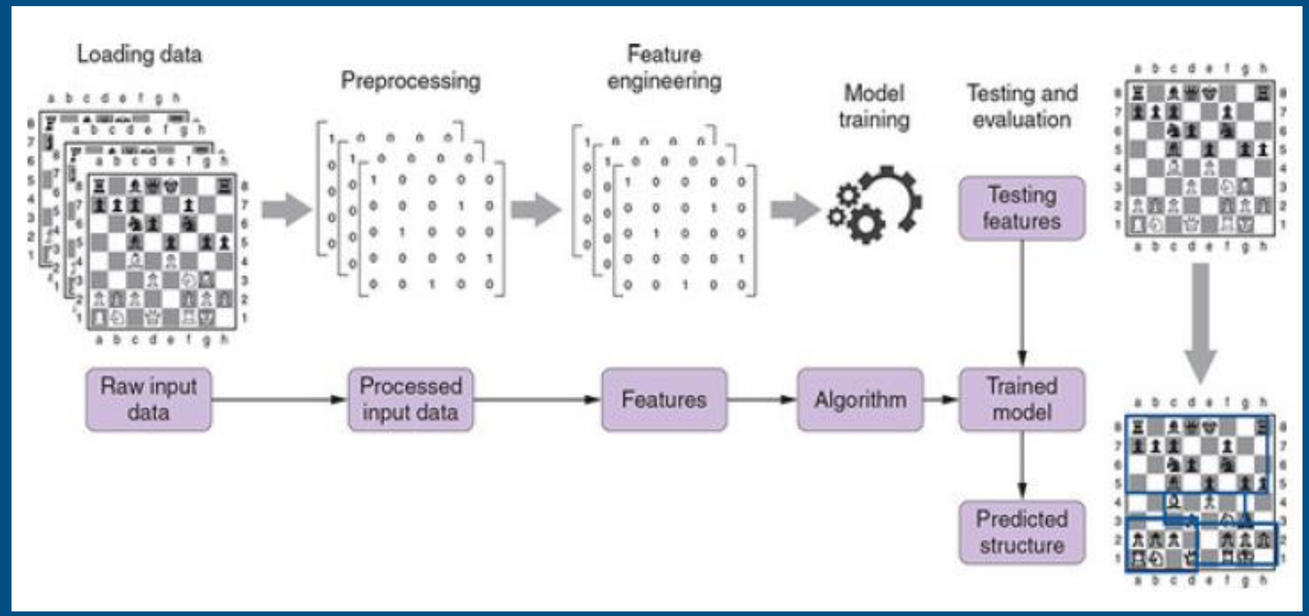
머신러닝 기초

- 지도학습을 위한 머신러닝 파이프라인



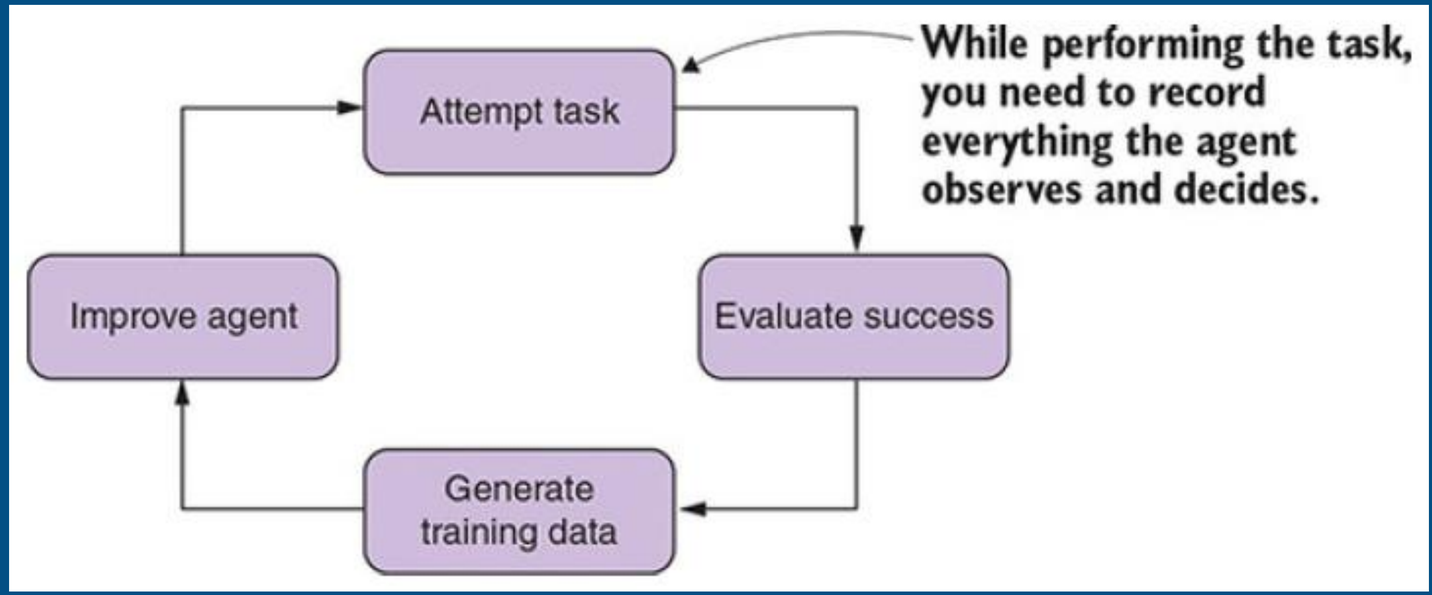
머신러닝 기초

- 비지도학습
  - **비지도학습 (Unsupervised Learning)**은 지도학습과 반대로 학습 과정에서 어떤 라벨도 사용하지 않는다. 비지도학습에서 알고리즘은 입력된 데이터만으로 그 데이터가 갖는 패턴을 찾는 학습을 한다.



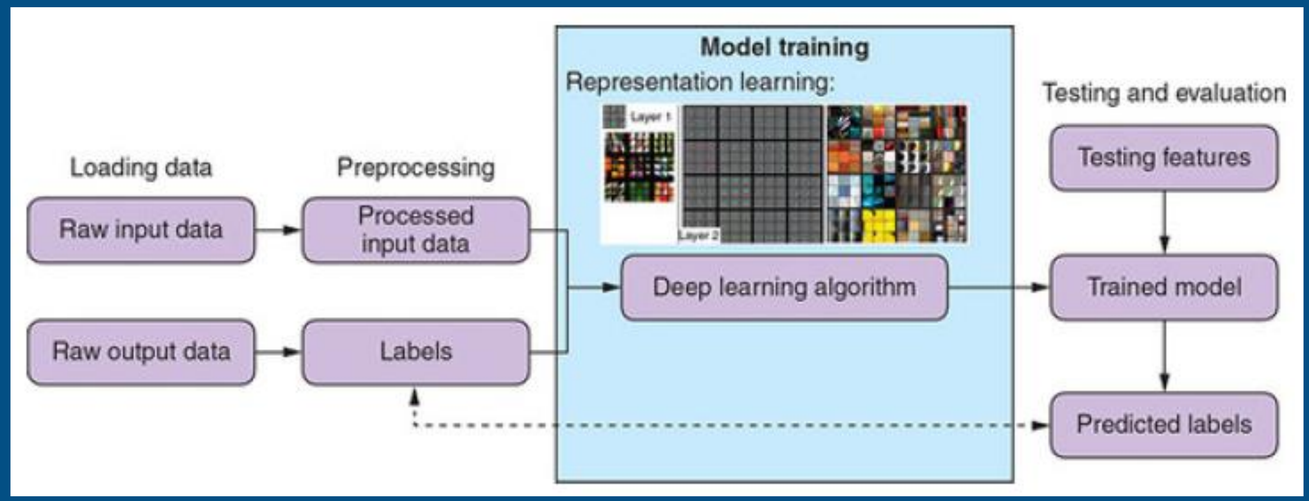
머신러닝 기초

- 강화학습
  - 지도학습은 매우 강력한 방식이지만 이를 위해 좋은 훈련 데이터를 찾아야 한다는 넘어서기 힘든 장벽이 있다. → 이 때 시행착오 접근 방식 중 하나인 **강화학습 (Reinforcement Learning)**을 적용할 수 있다.



머신러닝 기초

- 딥러닝
  - 딥러닝은 특정 모델 군을 사용하는 머신러닝의 하위 분야로, 간단한 기능들이 서로 연결되어 있는 구조다. → **신경망 (Neural Network)**
  - 딥러닝 모델의 첫번째 층은 원시 데이터로부터 학습한 후 이를 기본적으로 구조화한다. 이후 각 연속된 층에서 앞 층의 결과를 더 고도화하고 더 추상된 개념으로 구조화한다.
    → **표현형 학습 (Representation Learning)**



머신러닝 기초

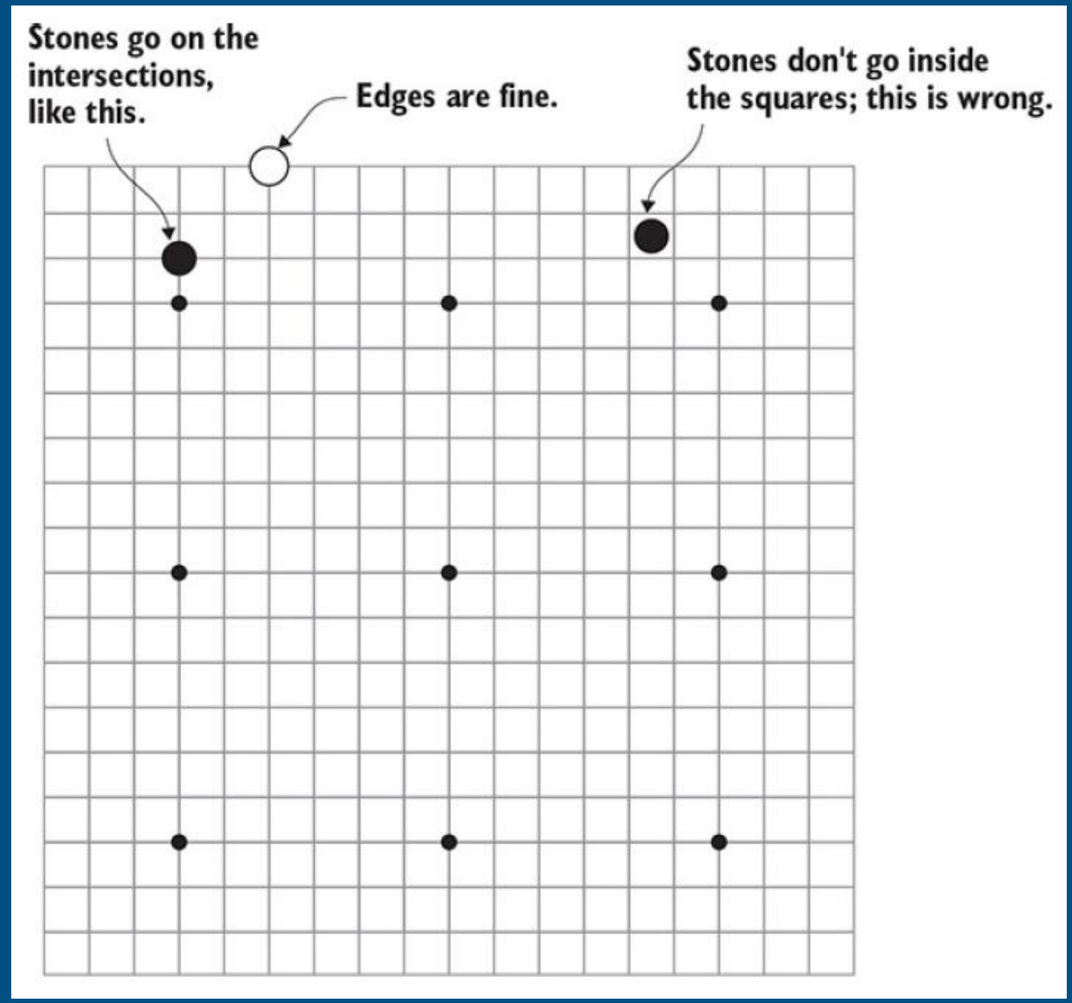
- 딥러닝을 사용하기 좋은 환경
  - 데이터가 구조화되어 있지 않은 경우
  - 많은 데이터를 사용할 수 있거나 데이터를 더 확보할 계획이 있는 경우
  - 충분한 컴퓨팅 자원과 충분한 시간이 있는 경우
- 전통적인 모델을 사용하기 좋은 환경
  - 구조화된 데이터를 가진 경우
  - 서술적 모델을 원하는 경우

머신러닝 문제로서의 바둑

- 왜 게임인가
  - 현실의 복잡성을 단순하게 만들어 놓아서 연구하고 있는 알고리즘에만 집중할 수 있음
  - 게임에서 여러분이 해야 할 일은 AI를 동작시키는 거 뿐이다.
  - 게임을 사용하면 실질적인 훈련 환경 생성이 필요 없고 강화학습에 대한 기술이나 이론에 더 집중할 수 있다. (과연 그럴까…?)

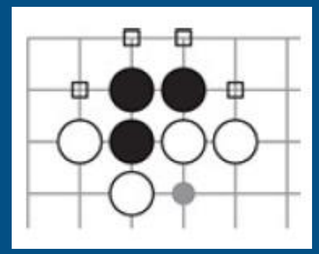
머신러닝 문제로서의 바둑

- 바둑판 이해하기

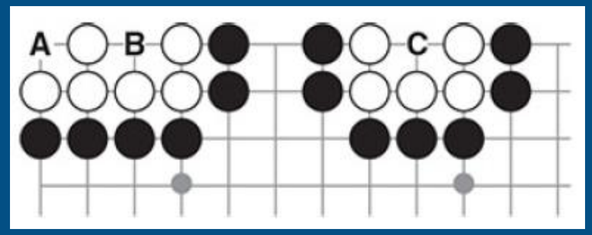


머신러닝 문제로서의 바둑

- 돌 놓기와 잡기
  - 흑돌/백돌을 잡을 기사를 정하고, 흑돌을 잡은 기사부터 번갈아가며 판 위에 돌을 놓는다.
  - 돌을 놓는 것을 착수라고 하며 착수한 돌 한 점을 착점이라고 한다.
  - 같은 동일한 색 돌이 맞닿아 있으면 연결되었다고 한다.
    - 연결되었다고 말하기 위해서는 위, 아래, 왼쪽, 오른쪽으로 맞닿아 있어야 한다.
    - 대각선 방향은 연결되었다고 하지 않는다.



  - 연결된 돌들과 맞닿은 빈 점을 **활로**라고 한다.
    - 모든 연결된 바둑돌이 판 위에 있으려면 1개 이상의 활로가 남아 있어야 한다.
    - 상대방의 활로를 모두 막으면, 즉 상대방의 연결된 돌의 마지막 활로를 막으면 그 돌 모두를 따낸 것이며 바둑판에서 들어낸다. 그러면 두 기사 모두에게 새 활로가 열린다.
    - 반대로 **상대 돌을 따내기 위해 마지막으로 두는 돌을 제외**하고 활로가 없는 곳에는 돌을 둘 수 없다.

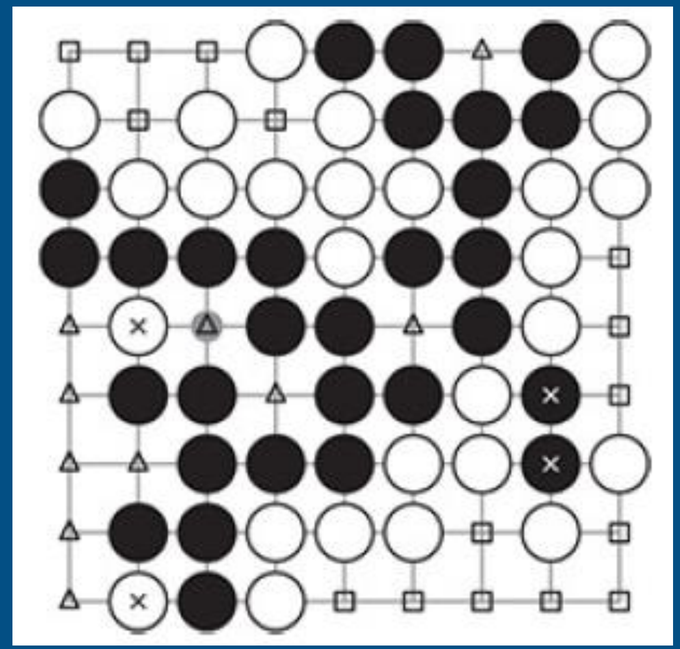


머신러닝 문제로서의 바둑

- 경기 종료 및 점수 계산
  - 양쪽 기사 모두 돌을 두는 대신 차례를 넘길 수 있다.
    만약 두 기사가 연속으로 차례를 넘기면 경기가 종료된다.
  - 점수 계산 전에 두 집을 만들지 못하거나 다른 돌과 연결되지 못하고 죽은 돌이 얼마인지 센다. 점수 계산 시점에서 죽은 돌은 따낸 돌과 동일하게 센다.
  - 경기의 최종 목표는 상대방보다 판에서 더 많은 영토를 차지하는 것이다.
    집을 세는 걸 계가라고 하는데, 두 가지 방법이 있다.
    - 영토가 얼마나 많은지 보는 방식 (한국/일본)
    - 구역을 세는 방식 (중국)

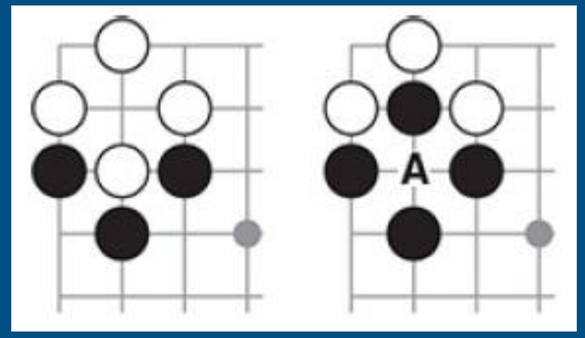
머신러닝 문제로서의 바둑

- 경기 종료 및 점수 계산
  - 또한 백은 돌을 나중에 두므로 추가점을 얻게 된다. 이를 **덤**이라고 한다.
    - 영토를 세는 방식의 경우(한국/일본)에는 6집 반
    - 구역을 세는 경우(중국)에는 7집 반
    - 반집을 두는 이유는 무승부를 피하기 위해서
  - 그림을 보며 계가하는 방법을 익히도록 하자.



머신러닝 문제로서의 바둑

- 패(Ko) 이해하기
  - 돌을 둘 때 한 가지 제약이 더 있다. 무한 동형반복이 일어나서 대국이 끝나지 않는 걸 방지하고자 이전 형태로 돌아가는 수를 금지한다.



머신러닝 문제로서의 바둑

- 머신에게 무엇을 가르칠 수 있을까
  - 포석 두기
  - 다음 수 찾기
  - 고려할 수 줄이기
  - 게임 현황 파악하기

머신러닝 문제로서의 바둑

- 바둑 AI가 얼마나 강력한 지 측정하는 방법
  - 일반 바둑 등급
  - 엘로(Elo) 평점 시스템

감사합니다!

스터디 듣느라 고생 많았습니다.