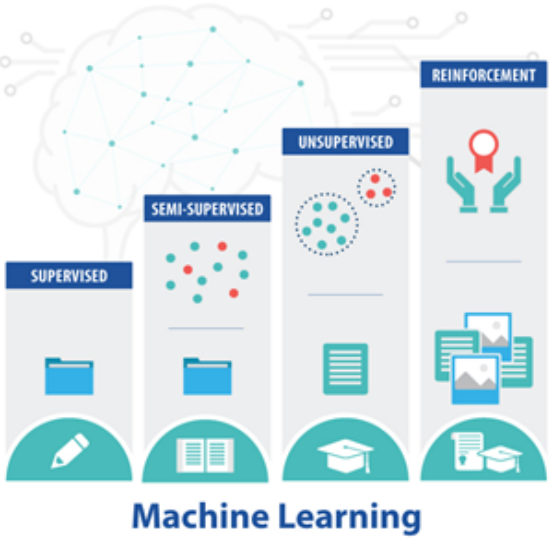
머신러닝(Machine Learning) 개요

[학습목표] 머신러닝의 개념, 머신러닝 역사, 머신러닝 종류 등을 살펴본다

1. 머신러닝이란?

Machine Learning, 기계 학습, 機械學習

- 컴퓨터 프로그램이 데이터와 처리 경험을 이용한 학습을 통해 정보 처리 능력을 향상시키는 것 또는 이와 관련된 연구 분야
- 기계 학습은 자율 주행 자동차, 필기체 문자 인식 등과 같이 알고리즘 개발이 어려운 문제의 해결에 유용



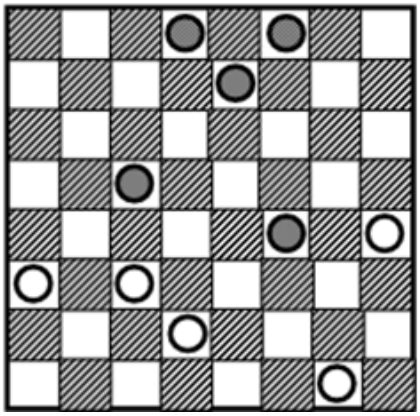
http://word.tta.or.kr/dictionary/dictionaryView.do?subject=%EA%B8%B0%EA%B3%84+%ED%95%99%EC%8A%B5

Arthur Samuel (1901 ~ 1990)

- 1959년 Machine Learning 이라는 용어를 대중화한 개척자
- 체커 프로그램으로 세계 최초의 성공적인 자기 학습 프로그램 중 하나로 인공 지능의 기본 개념에 대한 시연

“ 명시적인 프로그래밍 없이

컴퓨터가 학습하는 능력을 갖추게 하는 연구 분야 ”



https://artsandculture.google.com/entity/arthur-samuel/m02qqp3s?categoryid=historical-figure
http://webdocs.cs.ualberta.ca/~chinook/project/legacy.html

일반적인 프로그램과 머신러닝 프로그램

- 일반 프로그램 : A 입력에 B 조건이 성립되면 X를 동작시킴
- 머신러닝 프로그램 : A 정보를 입력할 때 정답이 X가 되는 조건 B를 찾도록 기계를 학습함

일반 프로그램

다음 수식을 계산한 결과는?

3 × 2 = 4 × -3 = 5 × 8 = 7 × -5 =	✓ 입력 A : 숫자 ✓ 조건 B : 곱하기 × ✓ 동작 X : 두 숫자를 곱해서 결과를 출력

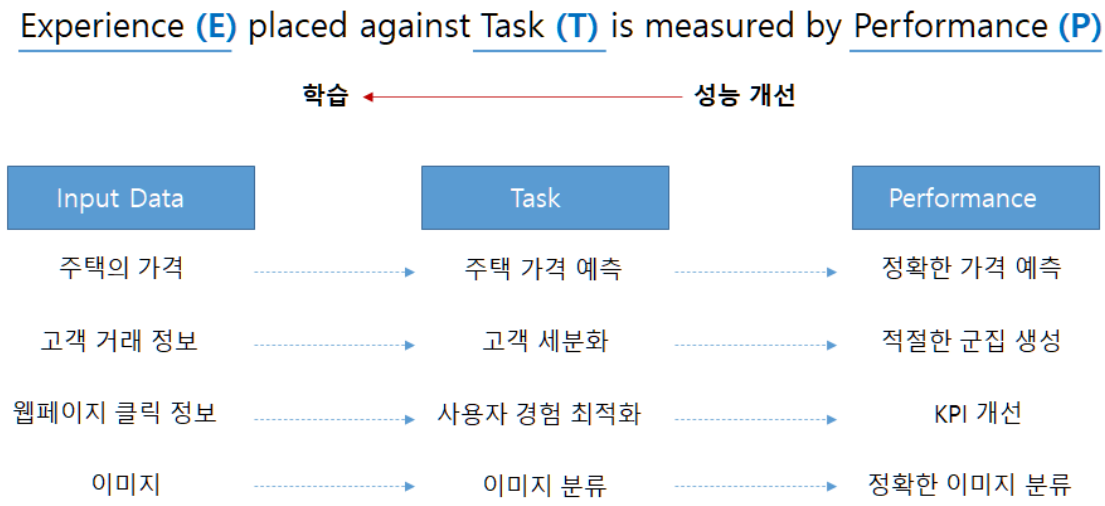
머신러닝 프로그램

□ 와 △에 들어갈 정수는?

입력 A						정답 X
5	× □ +	5	× △ =	0		조건 B
3	× □ +	2	× △ =	1		□ = 1
8	× □ +	3	× △ =	5		△ = -1
1	× □ +	4	× △ =	-3		

Tom Mitchell (1951 ~, 카네기 멜론 대학 컴퓨터 과학 교수)

“어떤 작업 T에 대한 컴퓨터 프로그램의 성능을 P로 측정했을 때 경험 E로 인해 성능이 향상됐다면, 이 컴퓨터 프로그램은 작업 T와 성능 측정 P에 대해 경험 E로 학습한 것이다.”



https://www.zldoty.com/what-is-machine-learning/

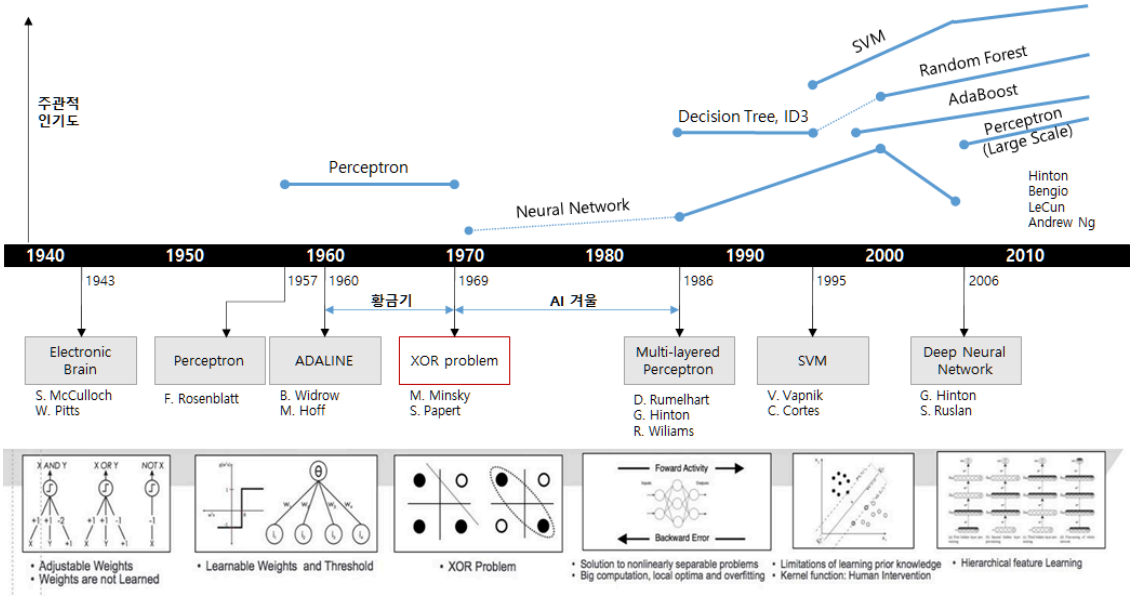
Machine Learning defined in Wikipedia

- **경험**을 통하여 **자동으로** 개선되는 컴퓨터 알고리즘에 대한 학문
- 샘플 데이터인 **Training Data**를 사용하여 **수학적 모델**을 생성
- 생성된 모델은 **예측 또는 의사결정**에 활용
- **통계학, 최적화, Data Mining, AI** 등과 밀접한 학문이고 자주 함께 언급됨
- 비즈니스 환경에서 활용하면서 **Predictive Analytics** 라고 표현하기도 함

https://ko.wikipedia.org/wiki/%EA%B8%B0%EA%B3%84_%ED%95%99%EC%8A%B5

---

2. 머신러닝의 역사



https://www.packtpub.com/product/deep-learning-essentials/9781785880360/

[참고]

1940년대

: 인공지능 태동기

- 1943년 Warren McCulloch(맥컬러) & Walter Pits(월터 피츠) 최초로 뇌의 뉴런 개념 발표
- 1949년 Hebb의 Hebbian Learning Thoery 발표

1950년대

: 논리 연산기로 기계가 논리적 추론을 할 수 있음을 보여줌

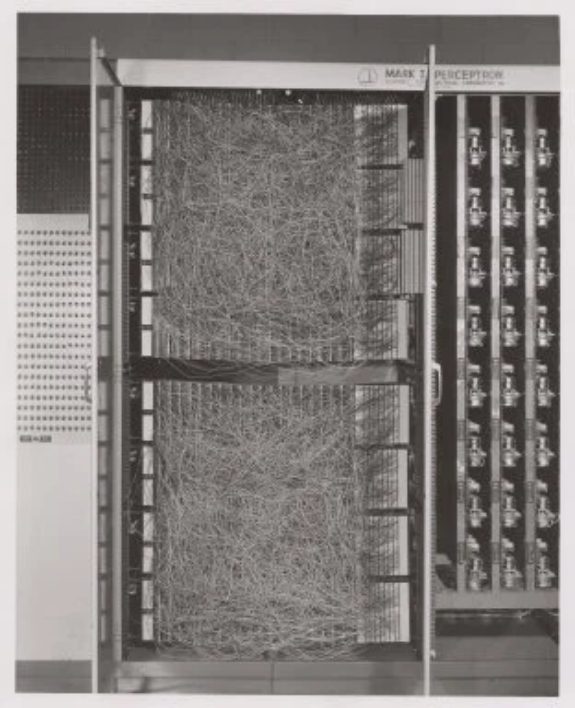
- 베이즈 정리를 기초로 확률 기반의 분류기 연구 시작(확률적 모델링)
  - 모든 특성값은 서로 독립임을 가정
  - 나이브 베이즈(Naive Bayes)
- 1950년 Alan Turing(앨런 튜링)의 Turing Test
  - 사람과 같은 지능을 가졌는지 테스트
- 1956년 Dartmouth AI Conference
  - Marvin Minsky, John McCarthy, Claude Shannon 및 Nathan Rochester는 "Artificial Intelligence (인공지능)"라는 용어를 제안

인공지능 황금기

- **1957년 : Frank Rosenblatt : Perceptron 개념 발표**
  - 최고의 인공신경망(다수의 입력으로부터 하나의 결과를 내는 알고리즘)
  - 뇌의 신경세포의 뉴런과 동작 방식이 비슷

마크I 퍼셉트론

- 뉴욕 타임스는 퍼셉트론 (Perceptron)이 "걸고, 말하고,보고, 쓰고, 스스로 번식하여 그 존재를 인식할 수 있는 전자 컴퓨터의 배아체"라고 보도



1960년대

- 1960년
  - Widrow, Hoff가 **Delta Learning Rule** 발표
  - Perceptron 훈련을 위한 실용적인 방법으로 사용. 더 나은 분류기 개발
- 1967년
  - Cover & Hart 최근접 이웃 알고리즘(Nearest Neighbor algorithm)
  - 거대한 이정표와 컴퓨터 패턴 인식의 탄생을 기록
- 1969년
  - Minsky Marvin은 **XOR 문제에 Perceptron이 적합하지 않다**고 주장

1974~1980 AI 첫 번째 겨울

1970년대

- 1970년
  - S. Linnainmaa가 **Back Propagation** 발표

1980년대

: Rule-based 알고리즘을 통한 탐색, 추론을 통한 지능 향상(전문가 시스템 발전)

- 1981년
  - P. J. Werbos가 **Multilayer Perceptron을 Back Propagation**으로 계산하는 방법을 발표
- 1986년
  - Smolensky이 **Boltzmann Machine** 발표
  - R. Quinlan이 머신러닝의 주류가 된 **Decision Tree 발표**
    - ID3 알고리즘
    - Decision Tree 변종인 ID4, CART 등의 알고리즘 발표(C4.5, C5.0, CART, CHAID 등)
- 1989년
  - Bell 연구소 Yann LeCun은 **LeNet 신경망** 발표
    - 초창기 합성곱 신경망(Convolution Neural Network:CNN)과 역전파를 연결
    - 손글씨 숫자 이미지를 분류하는 문제에 적용함
    - 우편 봉투의 우편 번호 코드를 자동으로 읽기 위해 1990년대 미국 우편 서비스에 사용됨

1987 ~ 1993 : AI 두 번째 겨울

1990년대

- 1995년
  - Vladimir Vapnik과 Corinna Cortes는 **현대적 Support Vector Machine** 발표
    - 2개 범주를 분류하는 이진 분류기
    - 고차원 평명인 초평면으로 분류
    - (Vladimir Vapnik과 Alexey Chervonekis가 만든 선형공식은 1963년 공개)
  - IBM 근무하던 Tin Kam Ho가 **Random Decision Forests 개념** 최초 발표
- 1997년
  - Freund와 Schapire는 부스팅, 앙상블 기법을 사용한 **AdaBoost 알고리즘**을 발표
    - 단일 분류기 성능을 높이는 방법
  - Leo Breiman은 **Gradient Boosting** 개념 제안

        - Cost 함수에 최적화 알고리즘을 적용한 것을 부스팅으로 해석
    - Jerome H. Friedman에 의해 Gradient Boosting 알고리즘 발전
    - Sepp Hochreiter; Jürgen Schmidhuber, **LSTM**(Long Short-Term Memory) 발표
- 1998년
    - LeNet5 발표

2000년대

: 사전에 정의된 feature와 weight로 스스로 학습하는 머신러닝

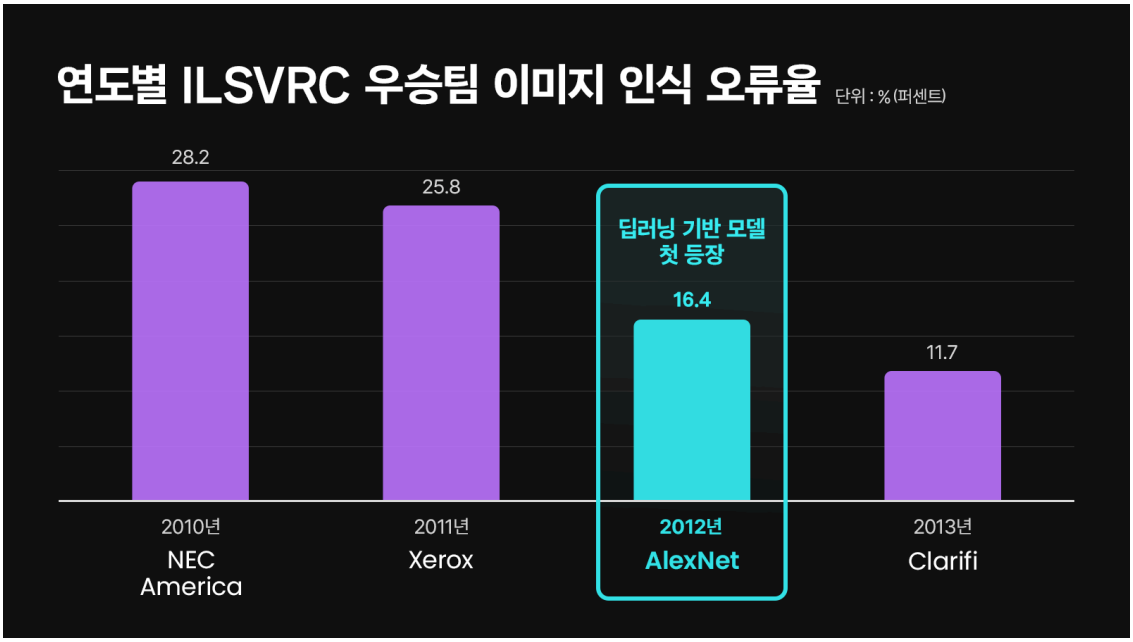
- 2000년
    - 커널함수를 사용한 **SVM** 발표
- 2001년
    - Leo Breiman이 Random Decision Forests 발전
    - 앙상블 기법을 이용한 여러 개 결정트리를 이용한 알고리즘으로 각각의 결정트리 생성 시 특성과 테스트 데이터를 무작위로 뽑는 기법
    - AdaBoost가 과적합과 이상치에 약점을 보였지만, Random Forest는 이에 뛰어난 성능을 보임
- 2005년
    - Hinton, LeCun, Bengio, Andrew Ng, Deep Learning
- 2006년
    - Hinton과 Salakhutdinov이 Boltzmann Machine 재조명
    - Hinton, **심층신뢰신경망**(DBN, Deep Belief Network) 발표
- 2009년
    - ImageNet dataset

2010년대

: 인간 뇌의 정보처리 방식을 흉내 낸 Deep Learning

- 2010년
    - Nair와 Hinton의 **ReLU**(Rectified linear unit) 함수 발표
- 2011년
    - IDSIA의 Dan Ciresan이 GPU로 훈련된 **심층신경망(Deep Neural Network)** 으로 학술 이미지 분류대회에서 우승(인식률 74.2%, 오류율 25.8%)
        - 현대적 딥러닝의 첫번째 성공
- 2012년
    - Google's Large Scale Deep Neural Networks Project
        - **고양이넷**
        - 스탠포드대학의 Andrew Ng과 구글이 함께한 딥 러닝 프로젝트
        - 16,000개의 컴퓨터 프로세서와 10억 개 이상의 neural networks, DNN을 이용하여 유튜브에 업로드 되어 있는 천만 개 넘는 비디오 중 고양이 인식에 성공
    - 이미지 분류 대회 ImageNet에서 Geoffrey Hinton 팀이 심층 합성곱 신경망(ConvNet)으로 우승
        - 모델 이름 : **AlexNet**
        - 1400만개 이미지를 훈련하고 고해상도 컬러이미지를 1000개 범주로 분류
        - 전통적 방식 정확도 74.3%, 제프리 힌트 팀 83.6%
- 2014년
    - Ian Goodfellow **GAN**(Generative Adversarial Nets)
- 2015년

    - 이미지 분류 대회 ImageNet에서 ConvNet으로 96.4% 정확도 달성
- 2016년
    - 알파고1.0 : 16만 기보 지도학습기반 심층강화학급과 확률적 샘플링 기반 의사결정
- 2018년
    - **GTP** 출시



이미지출처: Kien Nguyen, Arun Ross. "Iris Recognition with Off-the-Shelf CNN Features: A Deep Learning Perspective", IEEE ACCESS SEPT(2017), p.3

2020년대

생성형 AI의 혁명

- 2022년 : 생성형 AI 붐의 시작, **GPT-3.5** 출시
- 2023년 : **GPT-4** 출시
    - GPT-3.5보다 약 500배 더 큰 데이터셋을 활용한 모델
    - 이미지, 오디오, 비디오 등 다양한 입력 데이터 동시 처리
    - 데이터 포맷도 다양하게 생성하는 멀티모달모델(LMM)로 진화

출처: https://news.skhynix.co.kr/post/all-around-ai-1

머신러닝의 주요 알고리즘

확률적 모델링

- 통계학 이론을 데이터 분석에 응용
- 초창기 머신러닝 형태 중 하나
- 나이브 베이즈(Naive Bayes)
- 로지스틱 회귀

- 참고 : 베이즈정리와 통계학 : 18세기

신경망

- 1세대 : Perceptron
- 2세대 : Multilayer Perceptron, Back Propagation
- 3세대 : Boltzmann Machine, ReLU, Dropout, Local Minima

커널(Kernel) 방법

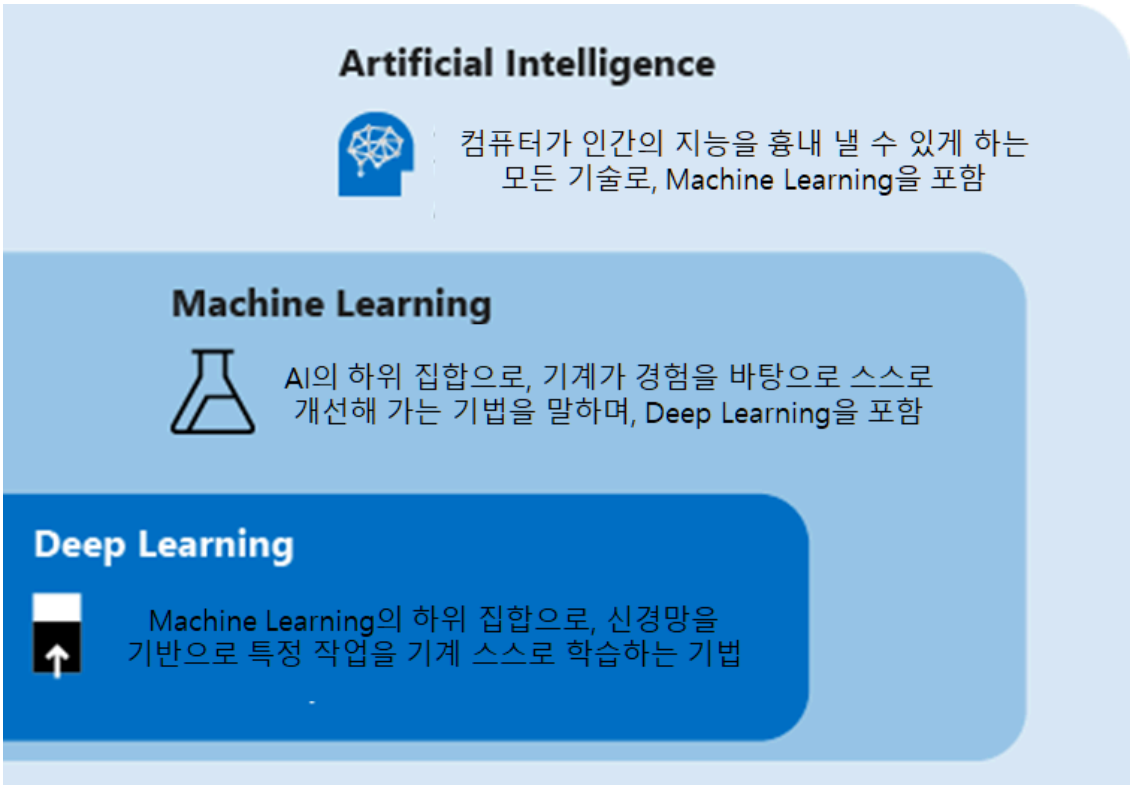
- 분류 알고리즘의 한 종류
- 서포트 벡터 머신(support vector machine)

결정 트리

- 랜덤 포레스트
- 그래디언트 부스팅
- 머신러닝 경연대회 사이트인 캐글(Kaggle)
  - 2010년 시작되었을 때 랜덤포레스트가 가장 선호하는 알고리즘
  - 2014년 그래디언트 부스팅이 선호

---

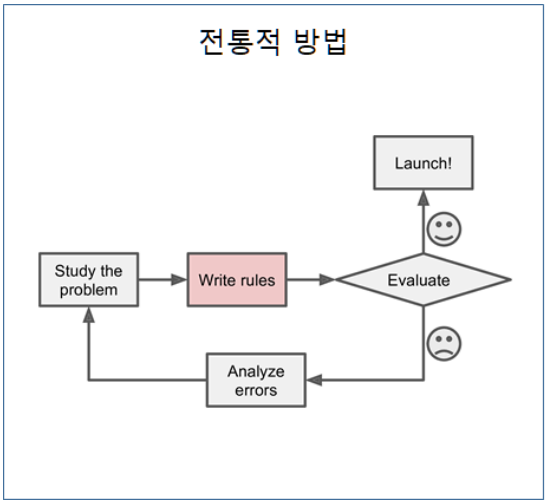
3. 머신러닝과 AI

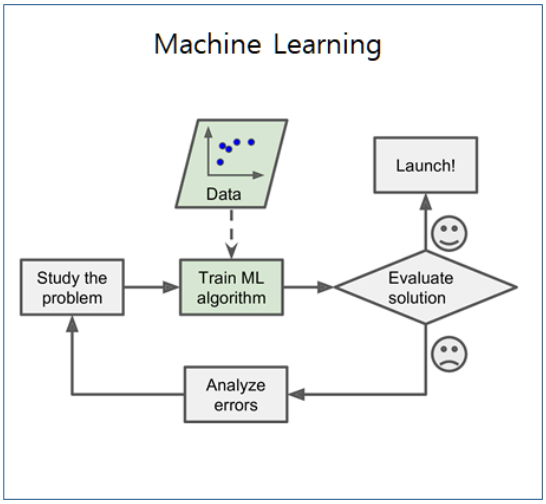


https://docs.microsoft.com/ko-kr/azure/machine-learning/concept-deep-learning-vs-machine-learning

머신러닝과 전통적 방식과 차이

- 전통적 방법의 경우 사소한 문제가 아닌 경우 규칙이 점점 길고 복잡해져 유지보수가 매우 어려움
- 기존 솔루션으로는 많은 수동 조정과 규칙이 필요한 경우, 머신러닝 모델 코드를 간단하게 만들 수 있음
- 전통적인 방식으로 해결할 수 없는 복잡한 문제를 머신러닝으로 해결





https://www.oreilly.com/library/view/hands-on-machine-learning/9781491962282/ch01.html

머신러닝의 장점

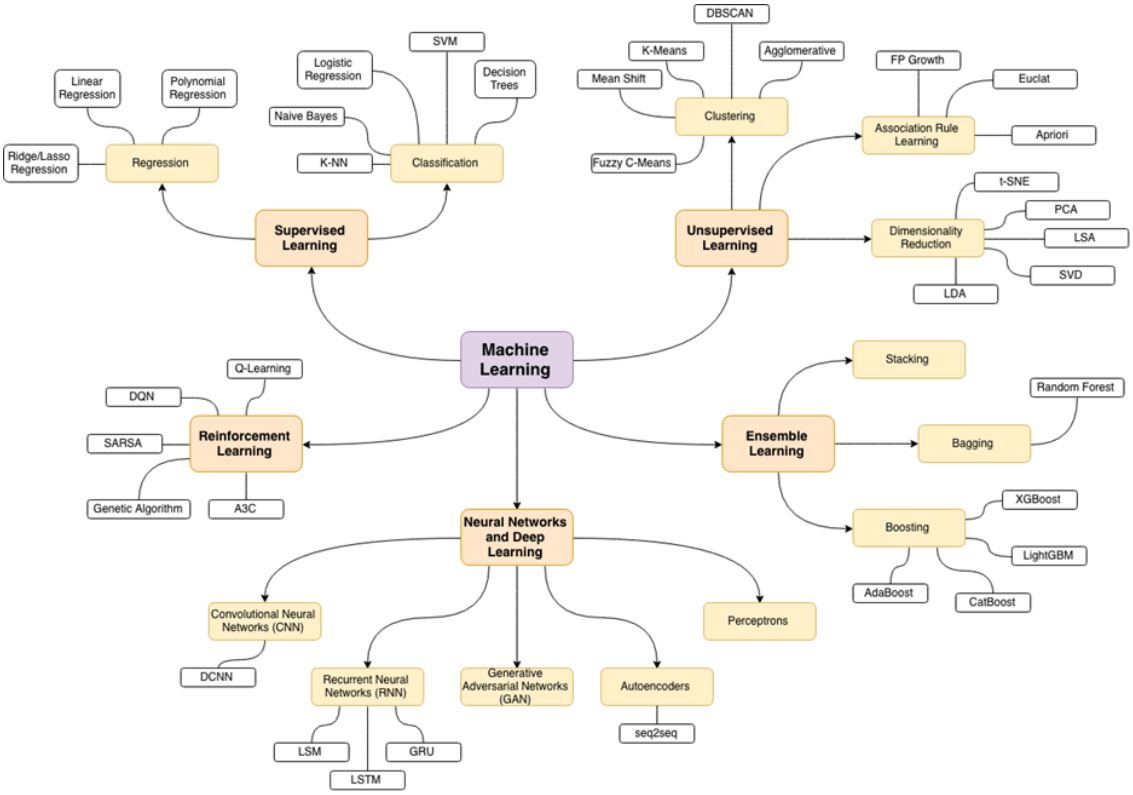
- 기존 솔루션으로는 많은 수동 조정과 규칙이 필요한 문제
    - 하나의 머신러닝 모델이 코드를 간단하게 만들고 전통적인 방법보다 더 잘 수행되도록 할 수 있음
- 전통적인 방식으로는 해결 방법이 없는 복잡한 문제
    - 가장 뛰어난 머신러닝 기법으로 해결 방법을 찾을 수 있음
- 유동적인 환경: 머신러닝 시스템은 새로운 데이터에 적응 가능
- 복잡한 문제와 대량의 데이터에서 통찰 얻기

어플리케이션 사례

- 이미지 분류 작업: 생산 라인에서 제품 이미지를 분석해 자동으로 분류
- 시맨틱 분할 작업: 뇌를 스캔하여 종양 진단
- 텍스트 분류(자연어 처리)
    - 자동으로 뉴스 기사 분류
    - 토론 포럼에서 부정적인 코멘트를 자동으로 구분
- 텍스트 요약
    - 긴 문서를 자동으로 요약
- 자연어 이해 : 챗봇(chatbot) 또는 개인 비서 만들기
- 회사의 내년도 수익을 예측하기 : 회귀분석
- 음성 인식 : 음성 명령에 반응하는 앱
- 이상치 탐지: 신용 카드 부정 거래 감지
- 군집 작업 : 구매 이력을 기반으로 고객을 나누고 각 집합마다 다른 마케팅 전략을 계획
- 데이터 시각화 : 고차원의 복잡한 데이터셋을 명확하고 의미 있는 그래프로 표현하기
- 추천 시스템 : 과거 구매 이력을 기반으로 고객이 관심을 가질 수 있는 상품 추천하기
- 강화 학습 : 지능형 게임 봇(bot) 만들기

---

4. 머신러닝 종류

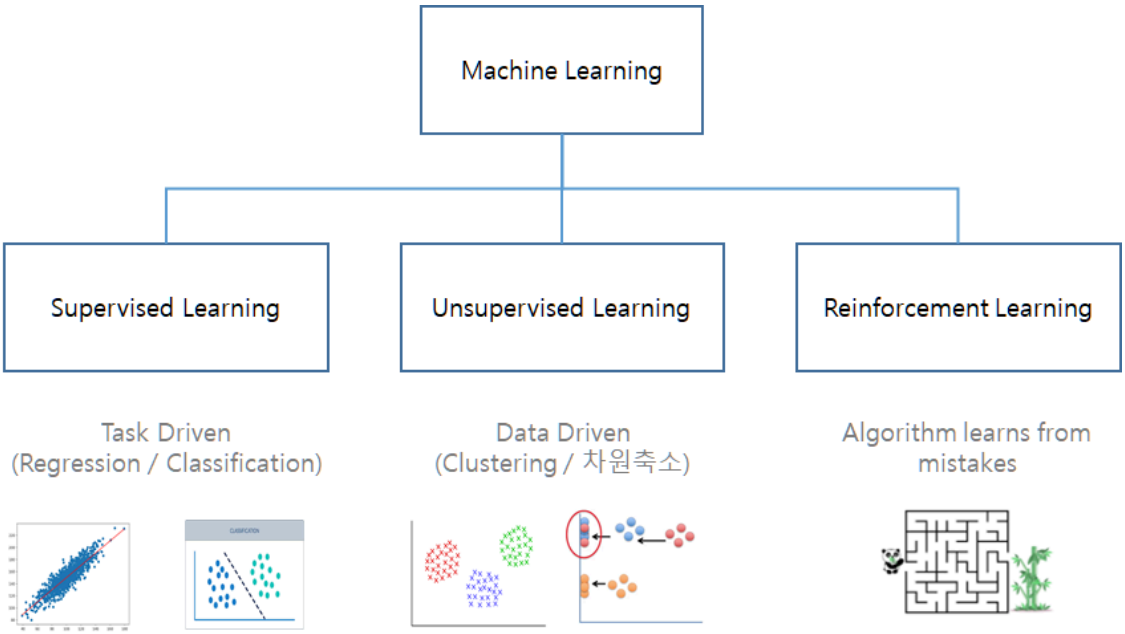


https://github.com/trekhleb/homemade-machine-learning

넓은 범주의 분류

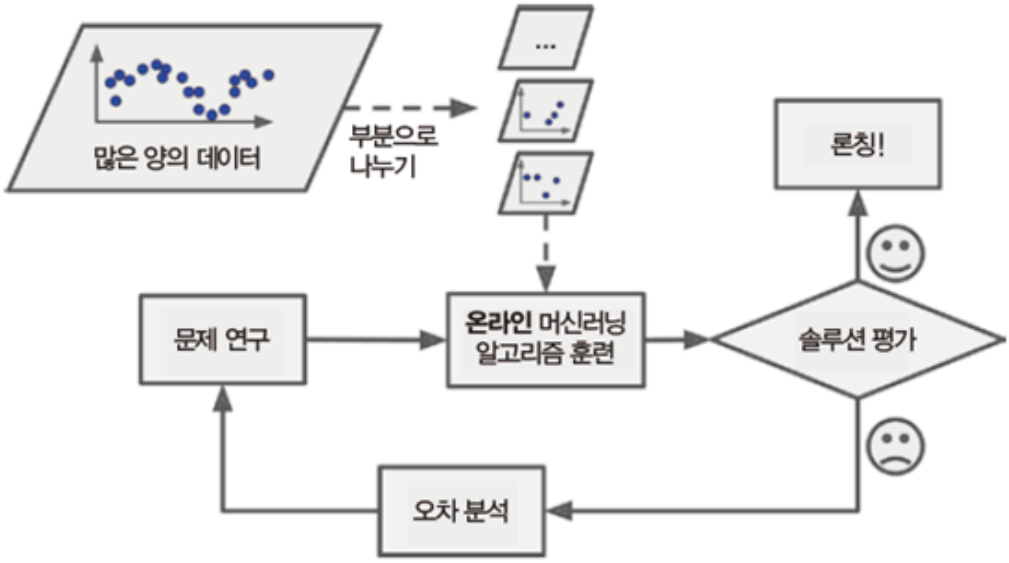
1) 감독 하에 훈련하는가?

- 지도(Supervised) 학습
- 비지도(Unsupervised) 학습
- 준지도(Semi-supervised) 학습
- 강화(Reinforcement) 학습
- 전이(Transfer) 학습



2) 실시간으로 점진적인 학습을 하는지?

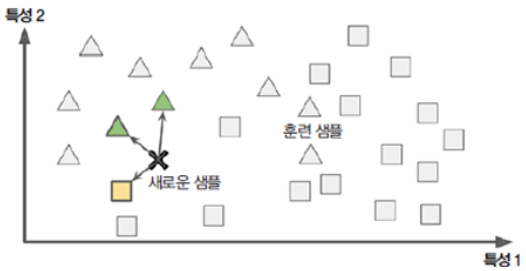
- 온라인 학습
  - 개별적 또는 소그룹(mini batch)으로 데이터를 순차적으로 공급하여 점진적으로 훈련
- 배치 학습
  - 시스템이 점진적으로 학습할 수 없음
  - 모든 데이터를 사용해 학습
  - 오프라인으로 수행 (많은 시간과 리소스 사용)



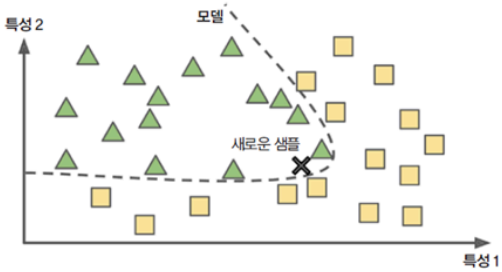
온라인 학습

3) 데이터 비교 vs 모델 기반인지?

- 인스턴스 기반 학습
- 모델 기반 학습



사례기반 학습



모델기반 학습

지도 학습(Supervised Learning)

- 학습 데이터에 입력값(특성)에 대한 출력값(레이블)이 함께 제시됨
  - 입력으로 훈련 데이터 세트가 있고 출력으로 각 훈련 세트에 대한 레이블 또는 "정답" 세트가 있음
- 알고리즘은 입력값과 출력값 사이의 관계를 가장 잘 설명할 수 있는 "모델"을 찾음
- "모델"을 사용하여 새로운 입력값에 대한 예측 수행
- 출력값이 수치형인 회귀와 범주형인 분류 문제로 나누어 짐

- 알고리즘
    - k-최근접이웃(K-Nearest Neightbor: KNN)
    - 선형 회귀(Linear Regression)
    - 로지스틱 회귀(Logistic Regression)
    - 서포트 벡터 머신(Support Vector Machine)
    - 의사결정트리(Decision Tree)
    - 랜덤 포레스트(Random Forest)
    - 신경망

비지도 학습(Unsupervised Learning)

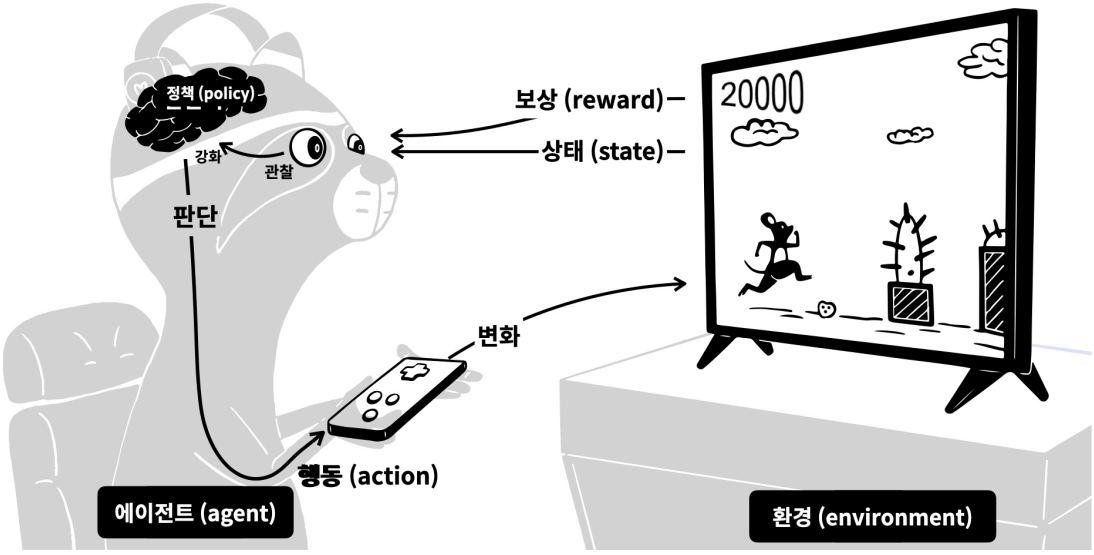
- 학습 데이터에 레이블이 지정되거나 분류되지 않은 테스트 데이터에서 학습
- 알고리즘은 학습 데이터의 특징만을 활용하여 목표한 결과를 산출
- 적절한 군집을 찾거나, 변수의 복잡성을 낮추기 위한 차원 축소 등
- GAN(Generative Adversarial Nets) 등과 같은 새로운 기법이 등장하고 있음
- 알고리즘
    - 군집분석(Clustering)
        - K-평균(K-Means)
        - 계층적 군집 분석(Hierarchical Cluster Analysis,HCA)
        - DBSCAN
    - 시각화와 차원축소
        - 주성분 분석(Principal Component Analysis, PCA)
        - 커널(Kernel) PCA
        - 지역적 선형 임베딩(Locally-Linear Embedding, LLE)
        - t-SNE(t-distributed Stochastic Neighbor Embedding)
    - 이상치 탐지
        - 가우스 분포를 이용한 이상치 탐지
    - 연관규칙(Association Rule)
        - Apriori
        - Eclat

준지도 학습(Semi-supervised Learning)

- 모든 데이터에 항상 레이블을 달아 줄 수 있는 것이 아닌 현실을 고려한 접근법
- 레이블이 달려있는 데이터와 레이블이 달려있지 않은 데이터를 동시에 사용하여 더 좋은 모델을 만들고자 함
    - 지도와 비지도 알고리즘 조합
- 항상 최선의 성능이 발휘되는 것은 아니나, 군집 형태에 가까운 경우 좋은 결과를 나타냄
- 알고리즘
    - 심층신뢰신경망(DBN)
    - 제한된 볼츠만 기계(RBM)

강화학습(Reinforcement Learning)

- 행동심리학에서 영감을 받은
- 구체적인 행동에 대한 지시가 없이 목표만 주어짐
- 현재 상태에 대한 최선의 액션을 보상에 의해 스스로 찾아 학습하게 하는 방법
- 학습한 내용은 최고의 결과를 얻기 위한 전략으로 활용됨
    - Agent인 환경(environment)을 관찰하고 행동(action)을 실행하고 그 결과 보상(reward)을 받음
    - 보상을 얻기 위한 정책이라 부르는 최상의 전략(policy)을 스스로 학습
- 데이터에서 보지 못한 내용에도 적응하여 반복하면서 더 좋은 결과를 얻을 수 있음
- 알고리즘
    - SARSA
    - Q-Learning



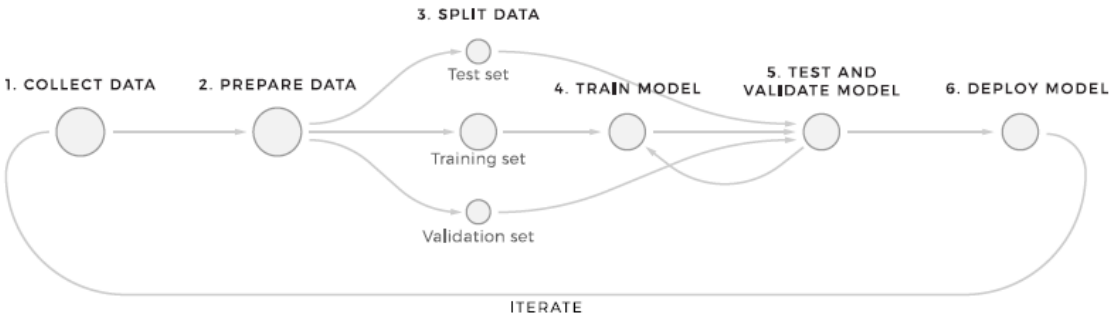
https://opentutorials.org/course/4548/28949

전이학습(Transfer Learning)

- 기존의 학습 방법들은 학습에 사용한 데이터와 이후 분석을 하려는 데이터가 같은 분포를 가지고 있다는 가정을 바탕으로 함
- 새로운 문제를 해결하고자 할 때 기존에 학습된 모델을 이용하여 새로운 모델을 만드는 방법
- 이미 잘 훈련된 모델이 있고, 해결하고자 하는 문제가 유사성이 있을 경우, 학습 데이터가 부족한 경우 등에 사용
- 기존의 pre-trained model을 미세 조정하여 사용하는 학습 방법이 대표적

---

5. 머신러닝 워크플로우(Workflow)



https://www.altexsoft.com/whitepapers/machine-learning-bridging-between-business-and-data-science/

1. Collect data : 유용한 데이터를 최대한 많이 확보하고 하나의 데이터 세트로 통합
2. Prepare data : 결측값, 이상값, 기타 데이터 문제를 적절하게 처리하여 사용 가능한 상태로 준비
3. Split data : 데이터 세트를 학습용과 평가용 세트로 분리
4. Train a model : 이력 데이터의 일부를 활용하여 알고리즘이 데이터 내의 패턴을 잘 찾아 주는지 확인
5. Test and validate a model : 학습 후 모델의 성능을 평가용 데이터 세트로 확인하여 예측 성능을 파악
6. Deploy a model : 모델을 의사결정 시스템에 탑재 / 적용
7. Iterate : 새로운 데이터를 확보하고 점진적으로 모델을 개선

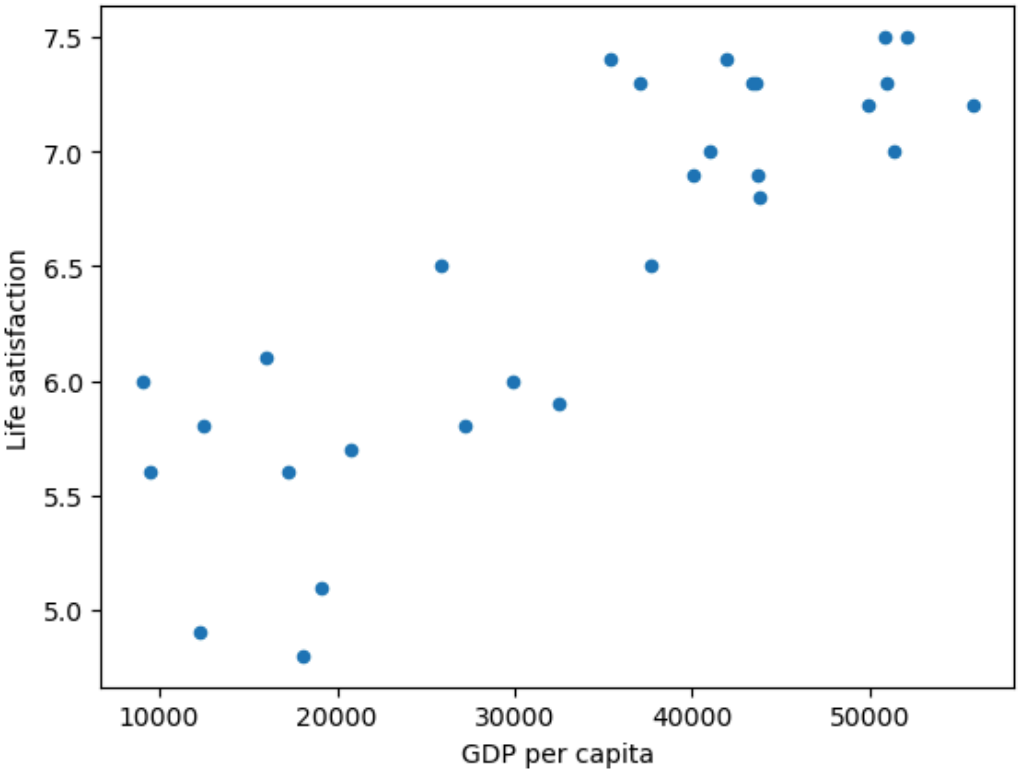
머신러닝 프로젝트 사례

예. 돈이 사람들을 행복하게 하는가?

- OECD 웹사이트 : Better Life Index 데이터셋
  - https://stats.oecd.org/index.aspx?DataSetCode=BLI
- IMF 웹사이트 : 1인당 GDP
  - https://www.imf.org/en/Publications/SPROLLs/world-economic-outlook-databases#sort=%40imfdate%20descending

In [2]:
```
import matplotlib
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import sklearn.linear_model

def prepare_country_stats(oecd_bli, gdp_per_capita):
    oecd_bli = oecd_bli[oecd_bli["INEQUALITY"]=="TOT"]
    oecd_bli = oecd_bli.pivot(index="Country", columns="Indicator",
                              values="Value")
    gdp_per_capita.rename(columns={"2015": "GDP per capita"},
                          inplace=True)
    gdp_per_capita.set_index("Country", inplace=True)
    full_country_stats = pd.merge(left=oecd_bli, right=gdp_per_capita,
                                  left_index=True, right_index=True)
    full_country_stats.sort_values(by="GDP per capita", inplace=True)
    remove_indices = [0, 1, 6, 8, 33, 34, 35]
    keep_indices = list(set(range(36)) - set(remove_indices))
    return full_country_stats[["GDP per capita", 'Life satisfaction']].iloc[keep_indices]
```

```
# Load the data (데이터 적재)
oecd_bli = pd.read_csv("data/lifesat/oecd_bli_2015.csv", thousands=',')
gdp_per_capita = pd.read_csv("data/lifesat/gdp_per_capita.csv",
                             thousands=',',delimiter='\t',
                             encoding='latin1', na_values="n/a")

# Prepare the data (데이터 전처리)
country_stats = prepare_country_stats(oecd_bli, gdp_per_capita)
X = np.c_[country_stats["GDP per capita"]]
y = np.c_[country_stats["Life satisfaction"]]

# Visualize the data (데이터 탐색)
country_stats.plot(kind='scatter', x="GDP per capita", y='Life satisfaction')
plt.show()

# Select a linear model (모델 선택)
model = sklearn.linear_model.LinearRegression()

# Train the model (모델 학습, 훈련)
model.fit(X, y)

# Make a prediction for Cyprus(슬로베니아) (예측, 추론)
X_new = [[22587]]  # Cyprus' GDP per capita
print(model.predict(X_new)) # outputs [[ 5.96242338]]
```




```
[[5.96242338]]
```

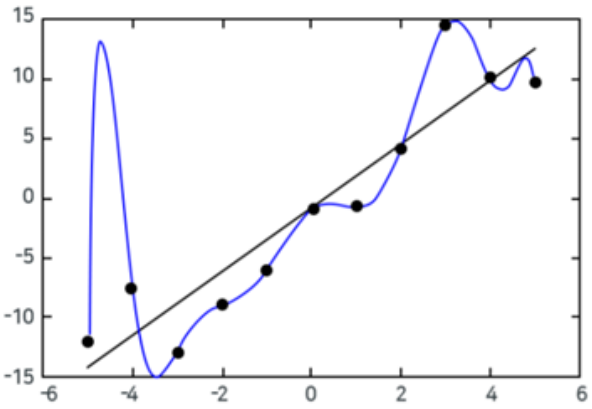
---

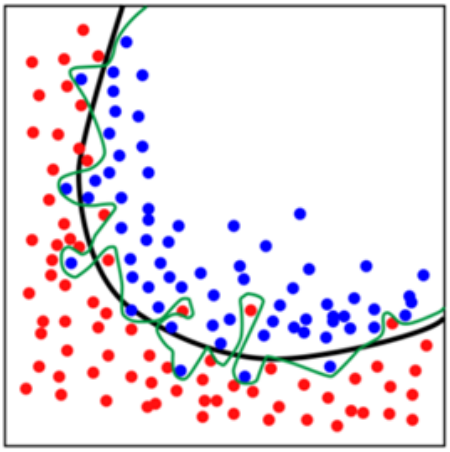
6. 머신러닝의 주요 이슈

머신러닝의 주요 작업 : 학습 알고리즘을 선택해서 데이터에 훈련시키는 것

- **나쁜 데이터**

    - 충분하지 않은 양의 데이터
    - 대표성 없는 데이터
    - 낮은 품질의 데이터 : 오류, 잡음, 이상치
    - 연관성이 적은 특성(변수)
- **나쁜 알고리즘**
    - 훈련 데이터 과대적합
    - 훈련 데이터 과소적합





테스트와 검증

- 모델이 새로운 사례에 잘 일반화 될수 있는가?

머신러닝에서 사용되는 주요 패키지

머신러닝 패키지

- 사이킷런(Scikit-Learn)

배열/선형대수/통계 패키지

- NumPy
- SciPy

데이터 핸들링

- Pandas

시각화

- Matplotlib
- Seaborn

딥러닝

- 텐서플로(Tensorflow)
- 케라스(Keras)
- 파이토치(PyTorch)

출처 :

-
https://projectresearch.co.kr/2017/06/14/%EB%A8%B8%EC%8B%A0%EB%9F%AC%EB%8B%9Dml%E
%EA%B0%84%EB%9E%B5%ED%95%9C-%EC%97%AD%EC%82%AC/

- https://ko.wikipedia.org/wiki/%EA%B8%B0%EA%B3%84_%ED%95%99%EC%8A%B5
