머신러닝
(Machine Learning)

LECTURE V: 머신러닝의 개요 1
(Brief Overview of Machine Learning)

Dai-Gyoung Kim

Department of Applied Mathematics

Hanyang University ERICA

머신러닝의 개요
(Brief Overview of Machine Learning)

Contents

- 머신러닝을 위한 파이썬 라이브러리
- 왜 머신러닝을 사용하는가?
- 머신러닝 시스템의 종류
- 머신러닝의 주요 도전과제
- 테스트와 검증
- 첫 번째 어플리케이션

■ 머신러닝을 위한 파이썬 라이브러리

- 과학계산과 수학, 데이터 분석을 위한 파이썬 라이브러리는 머신러닝의 구현을 위한 필수 도구임.
  1) scikit-learn
  2) NumPy
  3) SciPy
  4) pandas
  5) matplotlib
  6) mglearn

❖ scikit-learn (SciPy Toolkit)

- scikit-learn은 Numpy와 SciPy를 기반으로 작성된 과학계산 패키지 그룹임.
- scikit-learn의 특징은 머신러닝 전용으로 다양한 파이썬 머신러닝 라이브러리를 제공함.

❖ NumPy (Numerical Python)

- NumPy는 파이썬으로 과학 계산을 수행하기 위한 필수적인 패키지임.
- NumPy의 주요 기능과 객체는 다음과 같음.
  - ✓ 효율적인 다차원 배열 객체 ndarray 포함
  - ✓ 배열 원소를 다루거나 배열 간의 수학 계산을 빠르게 수행하는 함수
  - ✓ 디스크로부터 배열 기반의 데이터를 읽거나 쓸 수 있는 도구
  - ✓ 선형대수 계산, 푸리에 변환, 난수 생성기 포함
  - ✓ 파이썬과 C, C++, Fortran 코드와 통합하는 도구

- NumPy의 핵심 기능은 다차원 배열인 ndarrary임.
  - ✓ ndarray는 표준 파이썬 배열(list)과 다름.

```
import numpy as np

x = np.array([[1, 2, 3], [4, 5, 6]])
print("x:\n{}".format(x))
```

```
x:
[[1 2 3]
[4 5 6]]
```

```
x
```

```
array([[1 2 3],
       [4 5 6]])
```

❖ SciPy (Scientific Python)

- SciPy는 과학, 분석, 그리고 엔지니어링을 위한 과학(계산)적 컴퓨팅 영역의 여러 기본적인 작업을 위한 라이브러리(패키지 모음)임.
- SciPy는 기본적으로 Numpy, matplotlib, pandas, SymPy등 과 함께 동작함.
- SciPy는 수치적분 루틴과 미분방정식 해석기, 방정식의 근을 구하는 알고리즘, 표준 연속/이산 확률분포와 다양한 통계관련 도구 등을 제공함.
  - ✓ NumPy와 SciPy를 함께 사용하면 MATLAB을 완벽하게 대체할 수 있음.

- SciPy의 주요 패키지는 다음과 같음.
  - ✓ scipy.integrate : 수치적분 루틴과 미분방정식 해법기
  - ✓ scipy.linalg : numpy.linalg에서 제공하는 것 보다 다 확장된 선형대수 루틴과 행렬분해를 제공
  - ✓ scipy.optimize : 함수 최적화와 방정식의 근을 구하는 알고리즘
  - ✓ scipy.signal : 신호 처리 도구

- ✓ scipy.sparse : 희소 행렬과 선형 시스템 풀이법 제공
- ✓ scipy.special : 감마 함수처럼 흔히 사용되는 수학 함수를 구현한 포트란 라이브러리인 SPECFUN 확장
- ✓ scipy.stats : 표준 연속/이산 확률 분포와 다양한 통계 테스트 도구
- ✓ scipy.weave : 배열 계산을 빠르게 하기 위한 인라인 C++코드를 사용하는 도구

- SciPy의 함수 중에서 중요한 기능 중에 하나는 scipy.sparse임.
  - ✓ scipy.sparse는 희소 행렬 기능을 제공함.
  - ✓ 희소행렬(sparse matrix)은 0을 많이 포함한 다차원 배열을 저장할 때 사용함.

```
from scipy import sparse


# Create a 2D NumPy array with a diagonal of ones, and zeros everywhere else
eye = np.eye(4)
print("NumPy array:\n{}".format(eye))
```

```
NumPy array:
[[1. 0. 0. 0.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]
 [0. 0. 0. 1.]]
```

- 다음은 위의 단위행렬을 희소 행렬 포맷으로 변환하는 코드임.

```
from scipy import sparse
# Convert the NumPy array to a SciPy sparse matrix in CSR format
# Only the nonzero entries are stored
sparse_matrix = sparse.csr_matrix(eye)
print("SciPy sparse CSR matrix:\n{}".format(sparse_matrix))
```

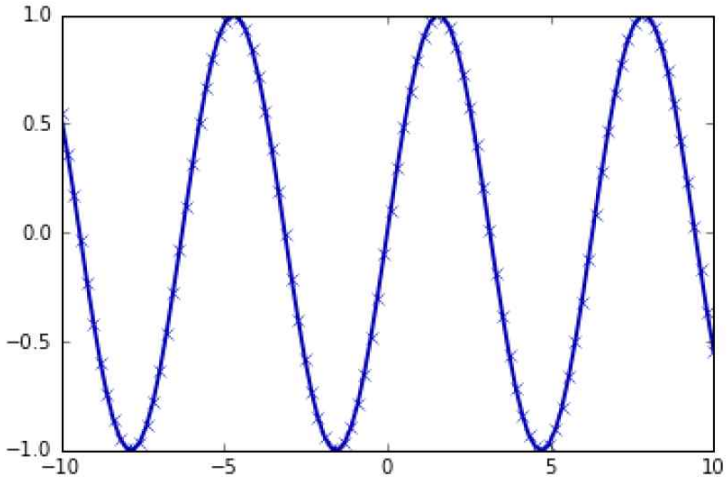
```
SciPy sparse CSR matrix:
(0, 0)    1.0
(1, 1)    1.0
(2, 2)    1.0
(3, 3)    1.0
```

❖ matplotlib

- matplotlib은 2차원 데이터 시각화를 생성하는 파이썬의 대표적인 과학 계산용 그래프 라이브러리임.
- 다음 코드는 사인 함수의 그래프를 그리는 간단한 예제임.

```
%matplotlib inline
import matplotlib.pyplot as plt

# Generate a sequence of numbers from -10 to 10 with 100 subintervals in between
x = np.linspace(-10, 10, 100)
# Create a second array using sine
y = np.sin(x)
# The plot function makes a line chart of one array against another
plt.plot(x, y, marker="x")
```



❖ pandas

- pandas는 구조화된 데이터를 빠르고 쉽게 다양한 형식으로 가공할 수 있는 풍부한 자료 구조와 함수를 제공하는 라이브러리임.
- pandas의 주요 객체인 DataFrame이라는 데이터 구조를 기반으로 함.

- DataFrame은 R의 data.frame을 모방하여 설계된 것이며 엑셀의 스프레드시트와 비슷한 테이블 형태라 할 수 있음.

- 다음 코드는 딕셔너리를 사용하여 DataFrame을 만드는 간단한 예제임.

```
import pandas as pd

# create a simple dataset of people
data = {'Name': ["John", "Anna", "Peter", "Linda"],
        'Location' : ["New York", "Paris", "Berlin", "London"],
        'Age' : [24, 13, 53, 33]
       }

data_pandas = pd.DataFrame(data)
# IPython.display allows "pretty printing" of dataframes in the Jupyter notebook
display(data_pandas)
```

	Age	Location	Name
0	24	New York	John
1	13	Paris	Anna
2	53	Berlin	Peter
3	33	London	Linda

- 다음은 위 테이블에 질의하는 간단한 예제임.

```
# Select all rows that have an age column greater than 30
display(data_pandas[data_pandas.Age > 30])
```

	Age	Location	Name
2	53	Berlin	Peter
3	33	London	Linda

❖ mglearn

- mglearn 라이브러리는 그래프나 데이터 적재와 관련한 세세한 코드를 일일이 쓰지 않아도 되게끔 이 책을 위해 만든 유틸리티 함수들을 모아놓은 것임.

■ 왜 머신러닝을 사용하는가?

머신러닝으로 풀 수 있는 문제들의 예

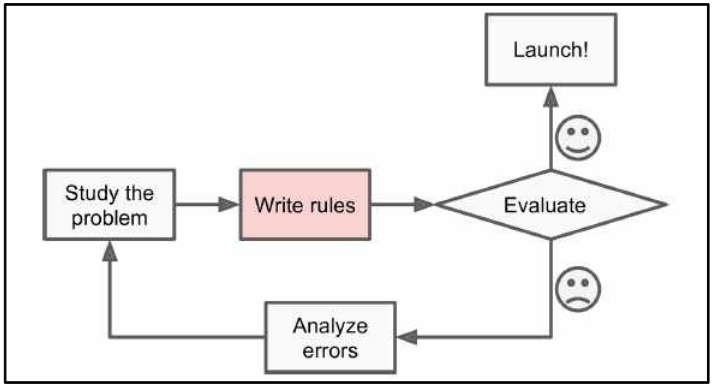
- 지도학습(supervised learning): 이미 알려진 사례를 바탕으로 일반화된 모델을 만들어 의사 결정 프로세스를 자동화함
  - ✓ 스팸 메일의 분류
  - ✓ 편지 봉투에 손으로 쓴 우편번호 숫자판별
  - ✓ 의료영상 이미지에 기반을 둔 종양판단
  - ✓ 의심되는 신용카드 거래 감지

- 비지도학습(unsupervised learning): 데이터는 주어지지만 출력은 제공되지 않고 데이터로부터 직접 학습함
  - ✓ 블로그 글의 주제 구분
  - ✓ 고객들을 취향이 비슷한 그룹으로 묶기
  - ✓ 비정상적인 웹사이트 접근 탐지

❖ 스팸 메일을 분류하는 스팸 필터 디자인 문제

전통적인 접근방법

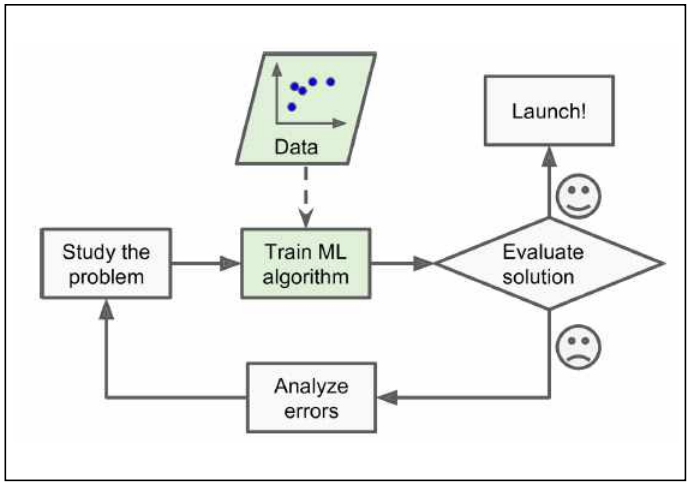
- 전통적인 스팸 필터 디자인
  1) 먼저 스팸에 어떤 단어들이 주로 나타나는 지 분석
     - ✓ 발송자 이름, 메일 주소, 본문 등에서 패턴 감지 (예를 들어, '4U', '신용카드', '무료', '굉장한' 등등)
  2) 발견한 각 패턴을 감지하는 알고리즘을 작성하여 프로그램이 이런 패턴을 발견했을 때 그 메일을 스팸으로 분류
  3) 프로그램을 테스트하고 충분한 성능이 나올 때까지 1단계와 2단계를 반복

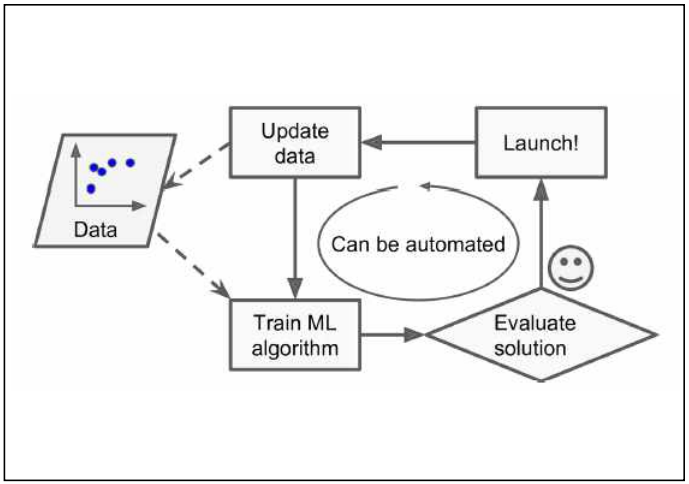


- 전통적인 접근 방법은 문제가 단순하지 않아 규칙이 점점 길고 복잡해지므로 유지보수가 매우 힘들어짐

머신러닝 접근방법

- 일반 메일에 비해 스팸에 자주 나타나는 패턴을 감지하여 어떤 단어와 구절이 스팸 메일을 판단하는 데 좋은 기준인지 자동으로 학습

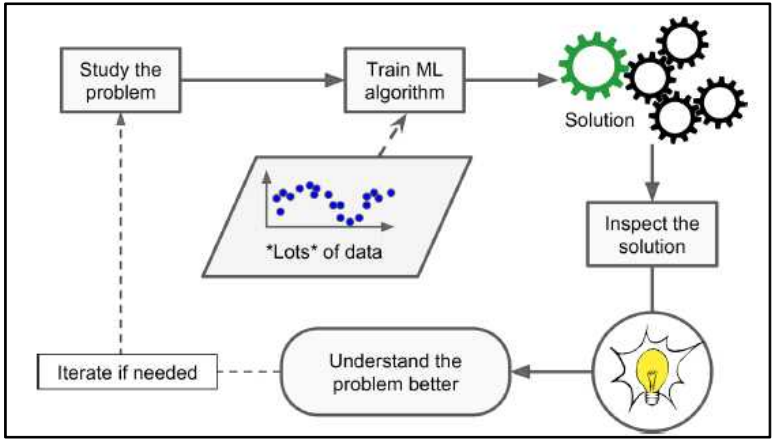




- 머신러닝이 유용한 또 다른 분야는 전통적인 방식으로는 너무 복잡하거나 알려진 알고리즘이 없는 문제를 다루는 분야임.
  - ✓ 예를 들어, 음성 인식, 얼굴 인식 문제 등

❖ 머신러닝을 통한 배움 (인간 학습: 통찰력)

- 머신러닝 기술을 적용해서 대용량의 데이터를 분석
  - ✓ 보이지 않은 패턴을 발견
  - ✓ 데이터 마이닝



❖ 머신러닝이 효과적인 분야 (요약)

- 기존 솔루션으로는 많은 수동 조정과 규칙이 필요한 문제
- 전통적인 방식으로는 전혀 해결 방법이 없는 복잡한 문제
- 유동적인 환경에 있는 문제
- 복잡한 문제와 대용량 데이터에서 통찰 얻기

■ 머신러닝 시스템의 종류

넓은 범주에서 분류

- 사람의 감독 하에 학습을 하는지 여부
  - ✓ 지도, 비지도, 준지도, 강화 학습
- 실시간으로 점진적인 학습을 하는지 여부
  - ✓ 온라인, 배치 학습
- 단순하게 알고 있는 데이터 포인트(샘플)와 새 데이터 포인트를 비교하는 것인지, 아니면 훈련 데이터셋에서 패턴을 발견하여 예측모델을 만드는지 여부
  - ✓ 사례 기반, 모델 기반 학습

▶ 지도 학습과 비지도 학습

- 머신러닝 시스템을 학습하는 동안 감독 형태나 정보량에 따라 분류 함

  - ❖ 지도 학습 (supervised learning)
  - ❖ 비지도 학습 (unsupervised learning)
  - ❖ 준지도 학습 (semisupervised learning)
  - ❖ 강화 학습 (reinforcement learning)

❖ 지도 학습

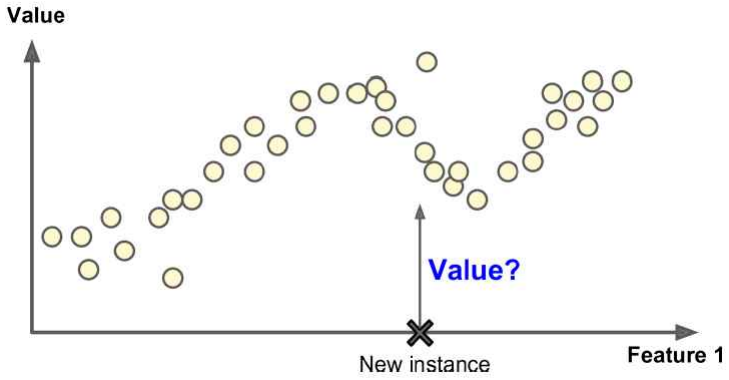
- 지도학습에는 알고리즘에 주입하는 훈련 데이터에 레이블(label)이 포함됨.
- 세부적으로 어떤 것을 예측하느냐에 따라 데이터의 레이블의 성질이 규명됨.
- 데이터를 통해 학습하는 레이블이 어떤 성질을 지니는지에 따라 크게 분류, 회귀, 랭킹으로 구분.

1. 분류 (classification)

- ✓ **분류는 클래스(class)를 판정을 산출하는 알고리즘**
  (클래스는 데이터의 항목의 집합)
- ✓ 분류는 입력 데이터의 항목을 나누는 것임.
- ✓ 이진 분류와 다중 분류가 나뉨.

2. 회귀 (regression)

- ✓ **회귀는 연속적인 수치를 예측하는 알고리즘**
- ✓ 어떤 함수가 내재되어 있다고 가정하여 예측 변수라고 하는 특성(feature)을 사용해 타깃(target) 수치를 예측함.



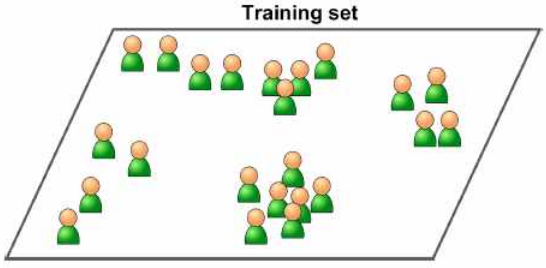
3. 랭킹 (ranking)

✓ 데이터의 순위 또는 순서를 예측하는 알고리즘

- **지도학습의 핵심 알고리즘**
  - ✓ k-최근접 이웃 (k-Nearest Neighbors)
  - ✓ 선형 모델
    - ◆ 선형 회귀 (Linear Regression)
    - ◆ 선형 분류 (Linear Classification)
  - ✓ 로지스틱 회귀 (Logistic Regression)
  - ✓ 나이브 베이즈 분류 (Naive Bayes Classification)
  - ✓ 결정 트리와 랜덤 포레스트 (Decision Tree and Random Forests)
  - ✓ 서포트 벡터 머신 (Support Vector Machine, SVM)
  - ✓ 신경망 (Neural Networks)

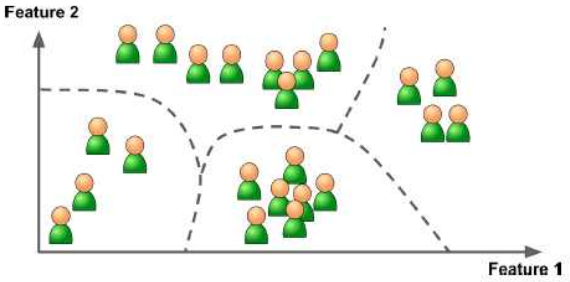
❖ 비지도 학습

- 비지도학습은 훈련 데이터의 레이블 정보 없이, 데이터를 직접 모델링하는 기법
- 대표적인 기법은 풀고자 하는 목표에 따라 군집, 시각화 및 차원 축소, 이상치 탐지, 연관규칙 학습으로 구분.



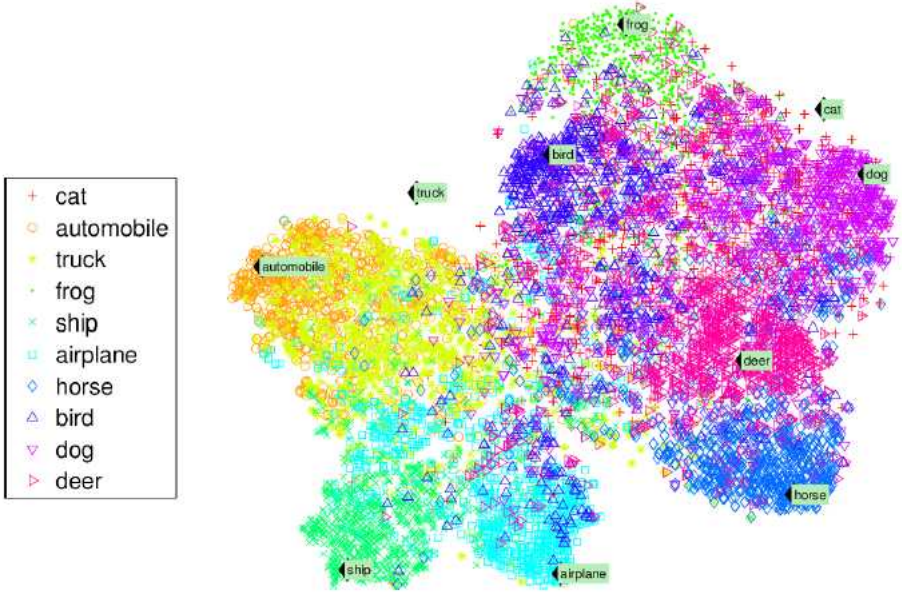
1. 군집 (clustering)

✓ 비슷한 데이터들을 묶어서 몇 개의 그룹을 만들어 데이터 패턴을 파악.



2. 시각화 및 차원 축소 (Visualization and Dimensionality Reduction)

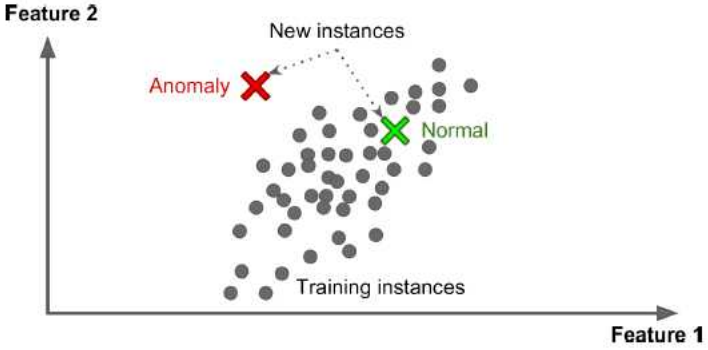
✓ **시각화 알고리즘**은 레이블이 없는 대규모의 고차원 데이터를 도식화가 가능한 2차원 또는 3차원 표현을 통해 데이터가 어떻게 조직되어 있는지 보여 줌으로써 예상치 못한 패턴을 감지할 수 있게 함.



✓ **차원 축소**는 데이터의 너무 많은 정보를 잃지 않으면서 데이터의 특성을 축소하는 기법

3. 이상치 탐지 (Anomaly Detection)

✓ 시스템이 정상 샘플로 훈련되고, 새로운 샘플이 정상 데이터인지 또는 이상치인지 판단함.



4. 연관 규칙 학습 (Association Rule Learning)

✓ 대량의 데이터에서 특성 간의 흥미로운(유의미한) 관계를 찾음.

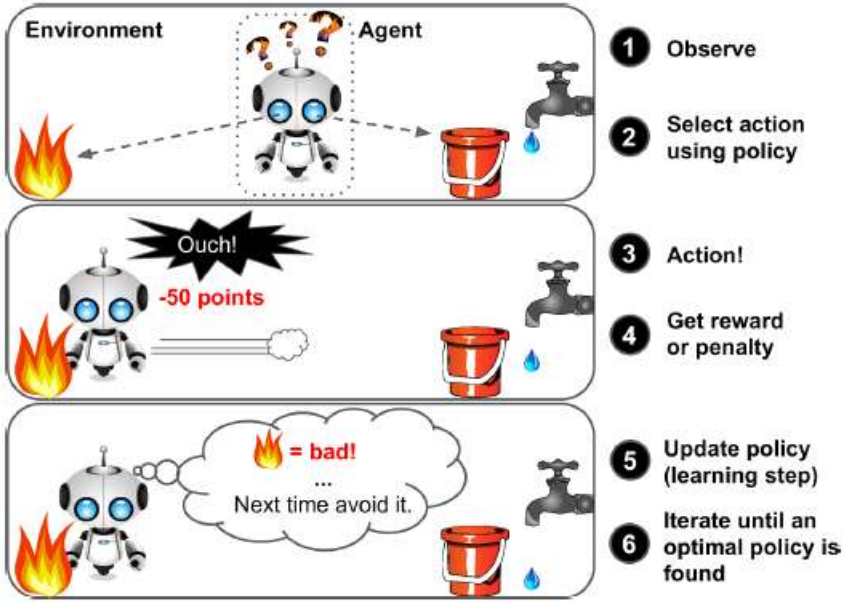
- **비지도학습의 핵심 알고리즘**
  - ✓ 군집
    - k-평균 (k-Means)
    - 계층 군집 분석 (Hierarchical Cluster Analysis, HCA)
    - 기댓값 최대화 (Expectation Maximization)
  - ✓ 시각화, 차원 축소, 이상치 탐지
    - 주성분 분석 (Principal Component Analysis, PCA)
    - 커널 PCA (Kernel PCA)
    - 지역적 선형 임베딩 (Locally-Linear Embedding)
    - t-SNE (t-distributed Stochastic Neighbor Embedding)
  - ✓ 연관 규칙 학습
    - 어프라이어리 (Apriori)
    - 이클렛 (Eclat)

❖ 준지도 학습

- 레이블 정보가 아주 적은 대용량 데이터를 모델링하는 학습 기법
  - ✓ 비지도 학습 방식으로 순차적으로 훈련된 다음 전체 시스템이 지도 학습 방식으로 세밀하게 조정되게 함. (제약 볼츠만 머신, restricted Boltzmann machine)

❖ 강화 학습

- 학습 시스템이 일반화 능력을 갖도록 하는 비지도 학습
  - ✓ 시스템이 예측한 결과를 스스로 평가하여 이를 기초로 더 좋은 평가를 받을 수 있도록 학습.
  - ✓ 학습하는 시스템을 **에이전트**(agent)라고 하며 환경을 관찰해서 행동을 실행하고, 그 결과로 보상 또는 벌점을 받아 시간이 지나면서 가장 큰 보상을 얻기 위해 **정책**(policy)라 부르는 최상의 전략을 스스로 학습.



✓ 강화 학습은 로봇의 자율 제어, 컴퓨터 게임의 인공지능, 마케팅 전략에 대한 최적화에 응용 (대표적인 예: 알파고 제로).

▶ 배치 학습과 온라인 학습

- 머신러닝 시스템을 분류하는 데 사용되는 또 다른 기준은 입력 **데이터의 스트림** (stream)으로 부터 **점진적**으로 학습할 수 있는 지 여부에 따라 구분.

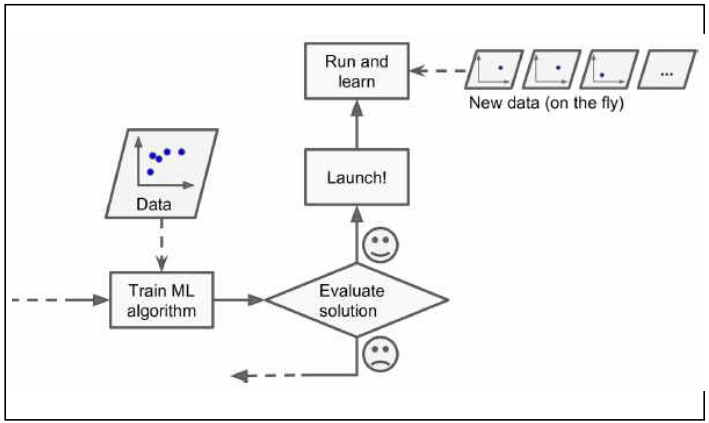
  ❖ 배치 학습 (batch learning)
  ❖ 온라인 학습 (online learning)

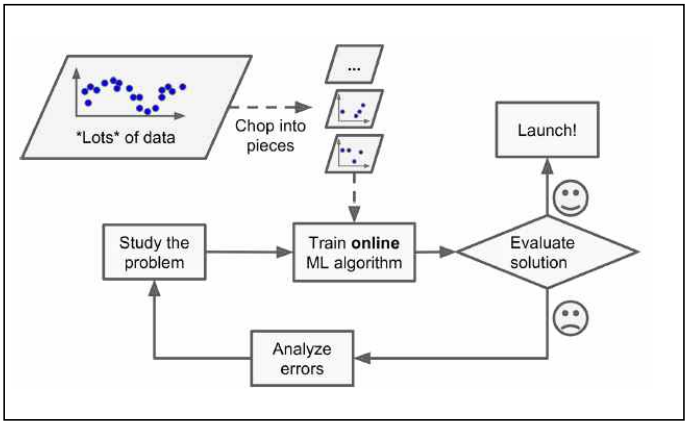
❖ 배치 학습

- 가용한 데이터를 모두 사용하여 훈련하는 학습
  - ✓ 보통 많은 시간과 컴퓨팅 자원을 사용하는 오프라인 학습
  - ✓ 배치 학습 시스템이 새로운 데이터에 대해 학습하려면 전체 데이터를 사용하여 새로운 시스템 버전을 처음부터 학습해야 함.
  - ✓ 자원이 제한된 시스템에는 부적합. (예, 스마폰, 탐사 로봇)

❖ 온라인 학습

- 데이터를 순차적으로 한 개씩 또는 **미니배치**(mini-bath)의 작은 묶음 단위로 주입하여 시스템을 훈련하는 학습
  - ✓ 매 학습 단계가 빠르고 비용이 적게 들어 데이터가 주어질 때마다 즉시 학습할 수 있음.
  - ✓ 연속적인 데이터를 받고 빠른 변화에 스스로 적응해야 하는 시스템에 적합.
  - ✓ 컴퓨팅 자원이 제한된 시스템에 적합.
  - ✓ 온라인 학습의 단점은 시스템에 나쁜 데이터가 주입되었을 때, 시스템 성능이 저하됨.





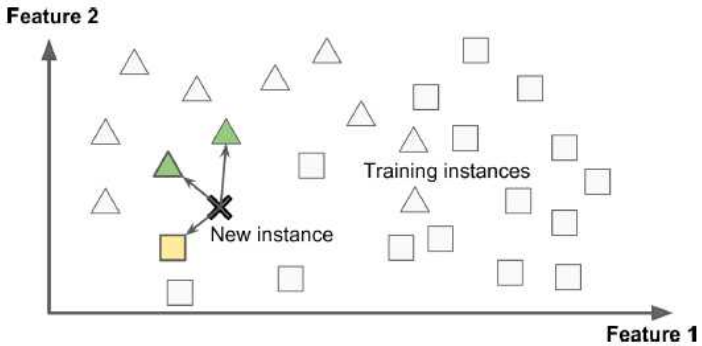
▶ 사례 기반 학습과 모델 기반 학습

- 머신러닝 시스템을 어떻게 **일반화**되는가에 따라 분류할 수 있음. 즉, 새로운 데이터셋에 어떻게 잘 작동하는 가에 따라 구분.

  ❖ 사례 기반 학습 (instance-based learning)
  ❖ 모델 기반 학습 (model-based learning)

❖ 사례 기반 학습

- 시스템이 사례를 기억함으로써 학습하고, 유사도(similarity) 측정을 통해 새로운 데이터에 일반화 함.



❖ 모델 기반 학습

- 데이터 샘플들의 모델(바라보는 관점, 가정)을 만들어 예측에 사용하는 학습 기법
  - ✓ 모델이 데이터를 어떻게 바라볼지에 대한 가정 또는 믿음
  - ✓ 모델이 머신러닝의 출발점
  - ✓ 모델기반의 작업
    1) 데이터 분석
    2) 모델과 모델 파라미터 선택 (예, 선형 모델)
    3) 훈련 데이터로 모델을 훈련
       (효용함수 또는 비용함수를 최적화)
    4) 새로운 데이터에 모델을 적용하여 평가하고, 예측 또는 추론하여 모델을 향상