The Machine Learning Landscape

Samkeun Kim <skim@hknu.ac.kr>

http://cyber.hankyong.ac.kr

What is Machine Learning?

Why Use Machine Learning?

Examples of Applications

- Types of Machine Learning Systems
  - ✓ Supervised/Unsupervised Learning
  - ✓ Batch and Online Learning
  - ✓ Instance-Based Versus Model-Based Learning

What Is Machine Learning?

- Explicit programming의 제한 사항
  - ✓ Spam filter: many rules
  - ✓ Automatic driving: too many rules

- Machine learning:

  "Field of study that gives computers the ability to learn without being explicitly programmed"

  (Arthur Samuel, 1959)

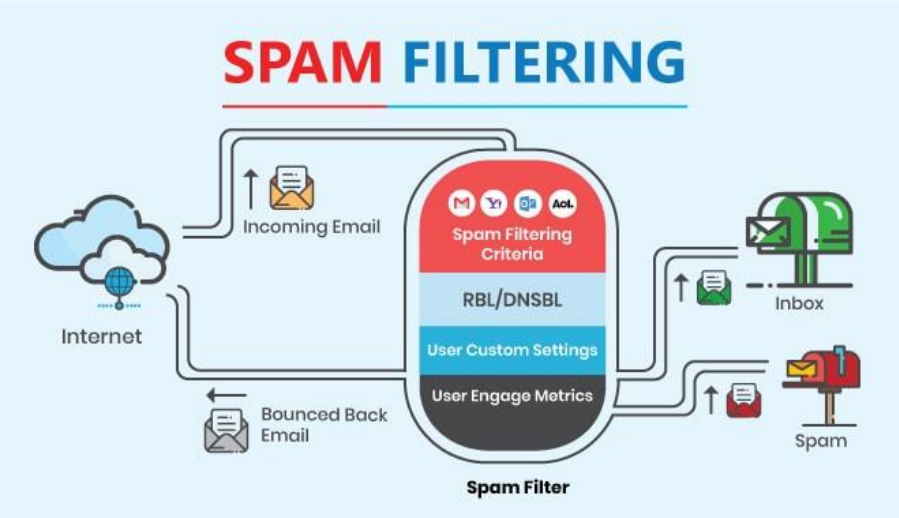
첫 번째 Machine Learning (ML) 애플리케이션 => "Spam filter"

- 수억 명의 생활을 개선시켜 줌
- 기술적으로 (**학습**을 매우 잘해서) ML 애플리케이션으로서 자격이 있다고 판단됨
- 그 후, 다양한 분야에서 수백 개의 제품과 기능을 조용히 지원하는 성공적인 ML 애플리케이션들이 쏟아져 나옴

Machine Learning : "데이터로부터 학습할 수 있는 프로그래밍 기법"

A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E. — Tom Mitchell, 1997

스팸 필터: "스팸 이메일 예제들(사용자가 플래그 지정함)과 정상 이메일 예제들을 통해 스팸 이메일에 플래그를 지정하는 방법을 학습할 수 있는 Machine Learning 프로그램"



시스템이 학습에 사용하는 예제 세트 : training set

각각의 학습 예제 : training instance

태스크 T : 새 이메일에 대해 스팸 플래그를 지정하는 일

경험 E : training data(set)

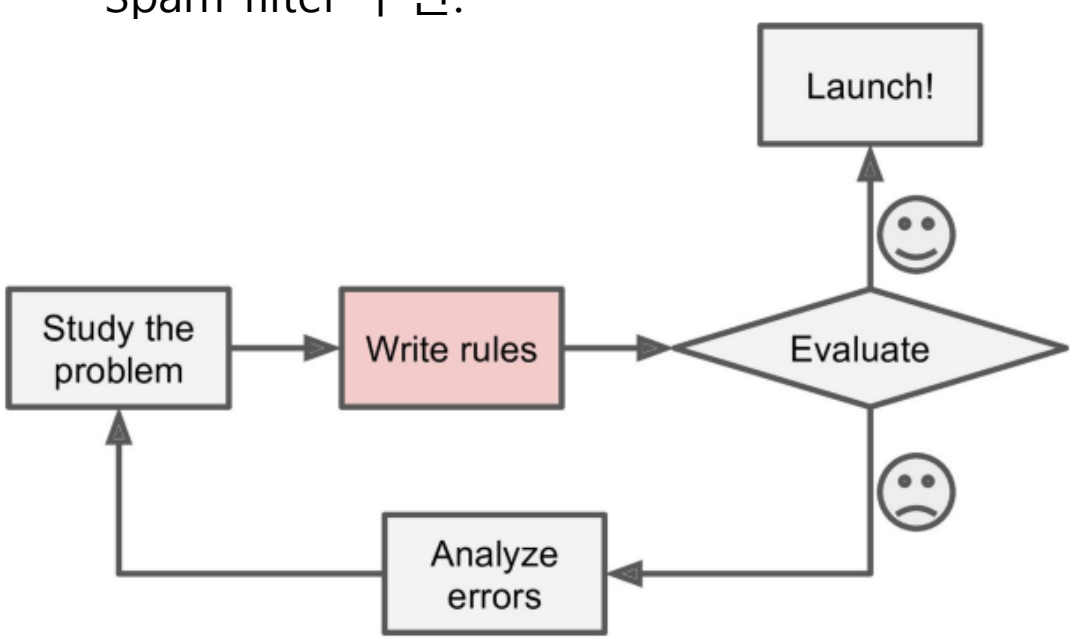
성능 측정 P : 옳게 분류된 이메일의 비율 ("accuracy")

그러나, Wikipedia의 어떤 내용을 다운로드하면 컴퓨터에 더 많은 데이터가 쌓이겠지만 어떤 태스크에서도 갑자기 성능이 좋아지는 것은 아니다.

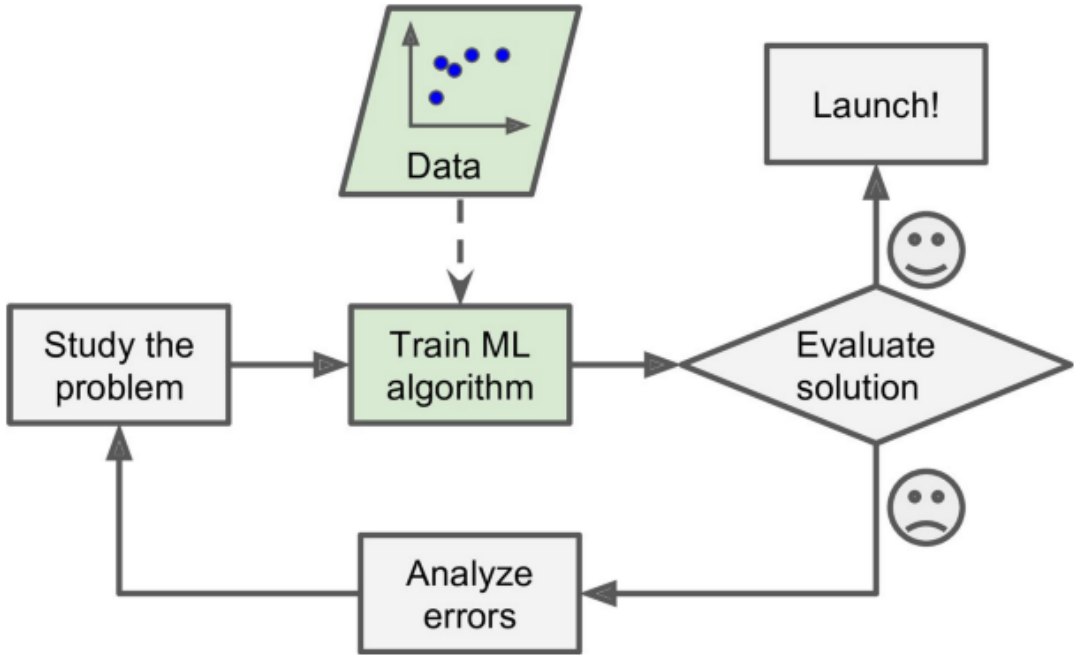
⇒ 따라서 이 경우는 Machine Learning이 아니다!!

Why Use Machine Learning?

Spam filter 구현:

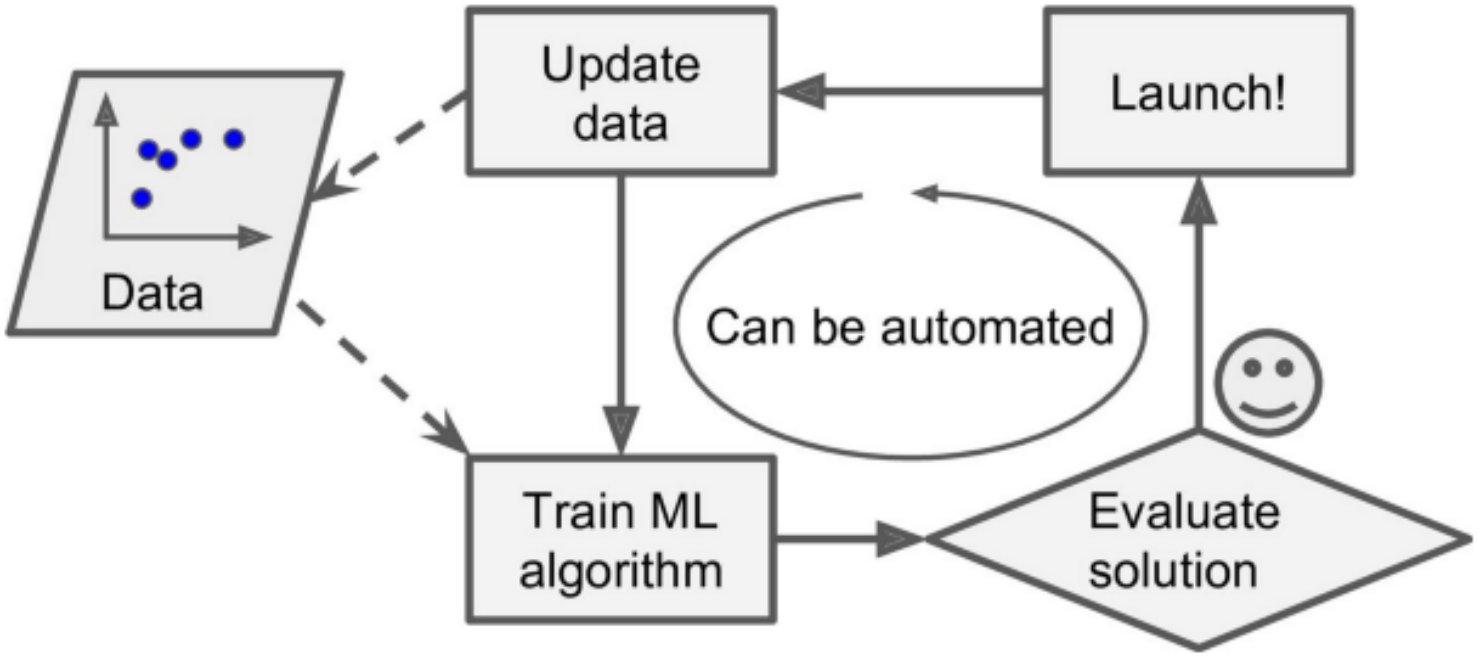


Traditional approach



Machine Learning approach

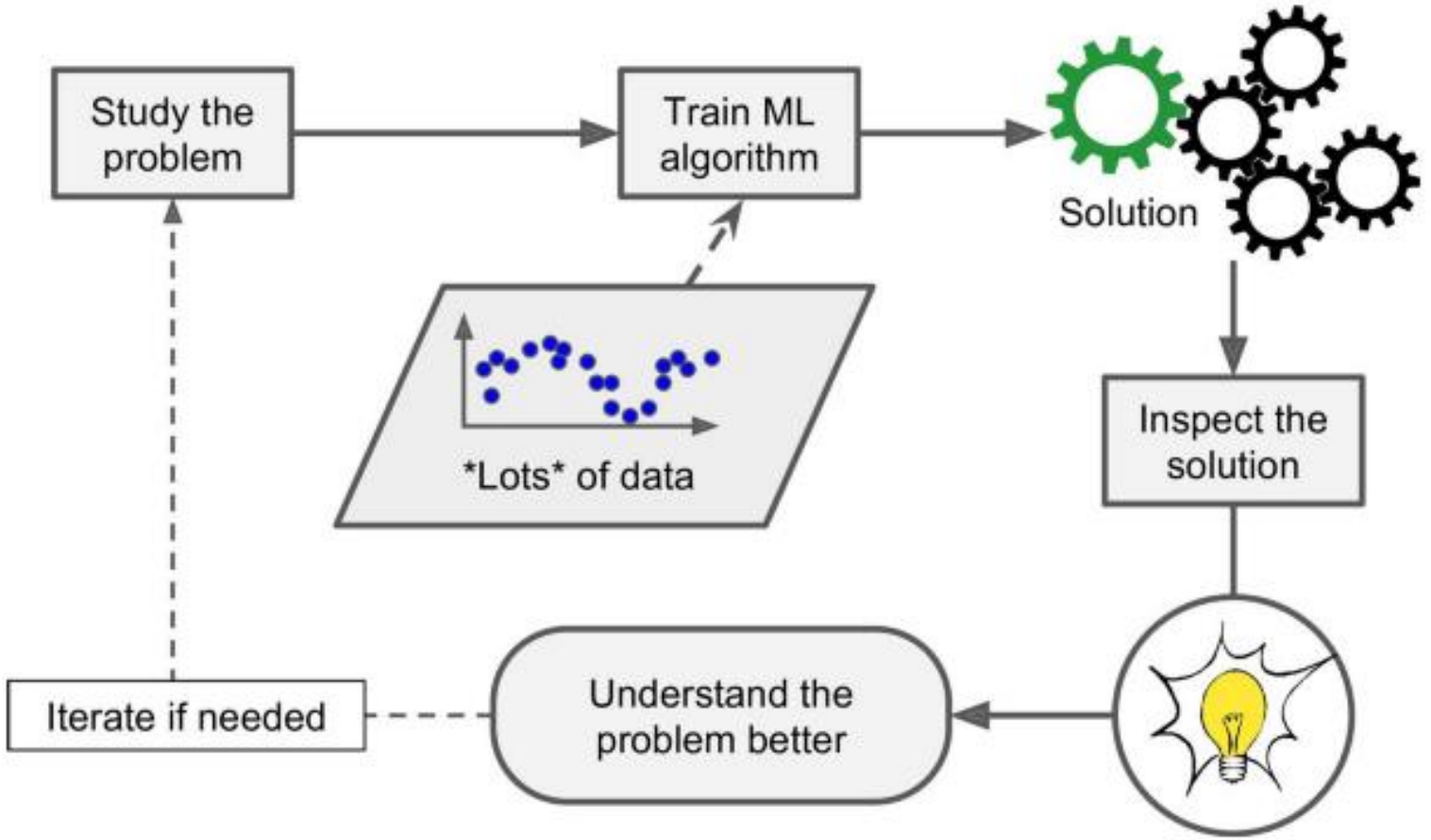
Spammers: "4U" -> "For U"로 변경



Automatically adapting to change

Machine Learning can help humans learn

- Data Mining: 대규모 데이터에 ML 기법 적용 => 숨어있는 패턴 발견



Types of Machine Learning Systems

ML 시스템의 종류:

- 인간 감독 하에 학습되었는지 여부(supervised, unsupervised, semisupervised, and Reinforcement Learning)
- 온라인에서 점진적으로 학습할 수 있는지 여부 (online versus batch learning)
- 새로운 데이터 포인트를 알려진 데이터 포인트와 단순히 비교하는 작업인지, 아니면 학습 데이터에서 패턴을 찾아서 예측 모델을 구축하는 작업인지 여부 (instance-based versus model-based learning)

위 기준은 배타적이지 않다.

- Spam filter => "spam과 ham 샘플들을 이용하여 학습된 deep neural network model을 사용하여 실시간으로 학습할 수 있다."

  ⇒ "online, model-based, supervised learning system"

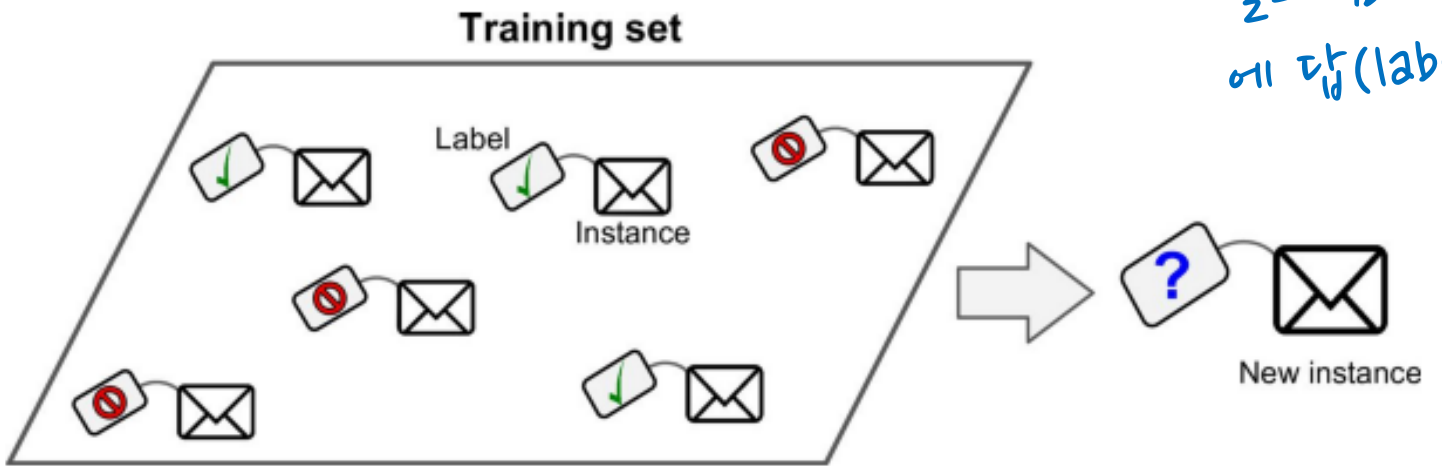
Supervised/Unsupervised Learning

학습 중에 받게 되는 감독의 양과 타입에 따라 ML 시스템을 분류 (4가지 부류):

- supervised learning
- unsupervised learning
- semisupervised learning
- Reinforcement Learning.

Supervised learning

알고리즘에 제공하는 훈련 데이터에 답(labels)이 포함되어 있다!



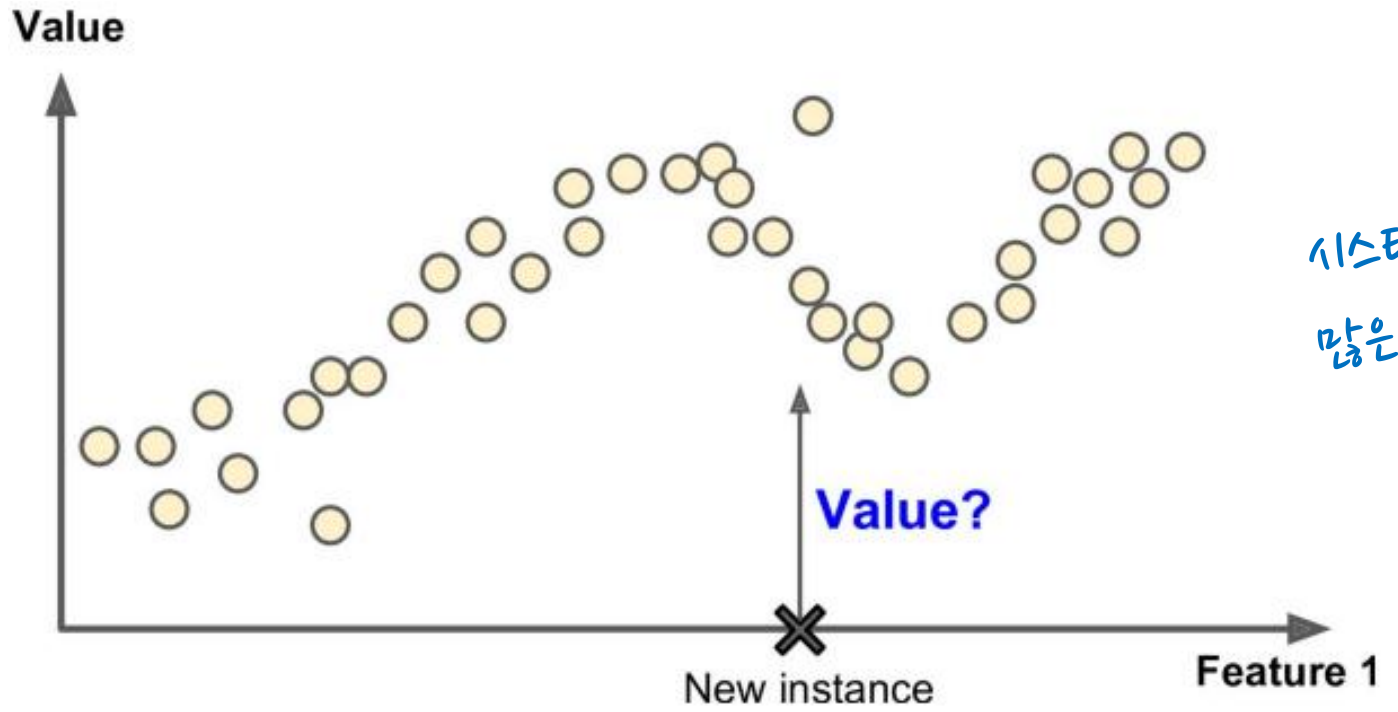
A labeled training set for supervised learning (e.g., spam classification)

전형적인 supervised learning task: classification

- Spam filter – 클래스(spam or ham)와 함께 많은 이메일 예제들로 학습
- 새로운 이메일을 분류하는 방법을 학습해야 함

Regression: 자동차 가격처럼 수치 값을 예측

- 학습을 위해 입력 특성(mileage, age, brand 등) + 라벨(price)을 포함하는 많은 예제 제공



시스템을 학습시키기 위해
많은 자동차 예제를 제공!

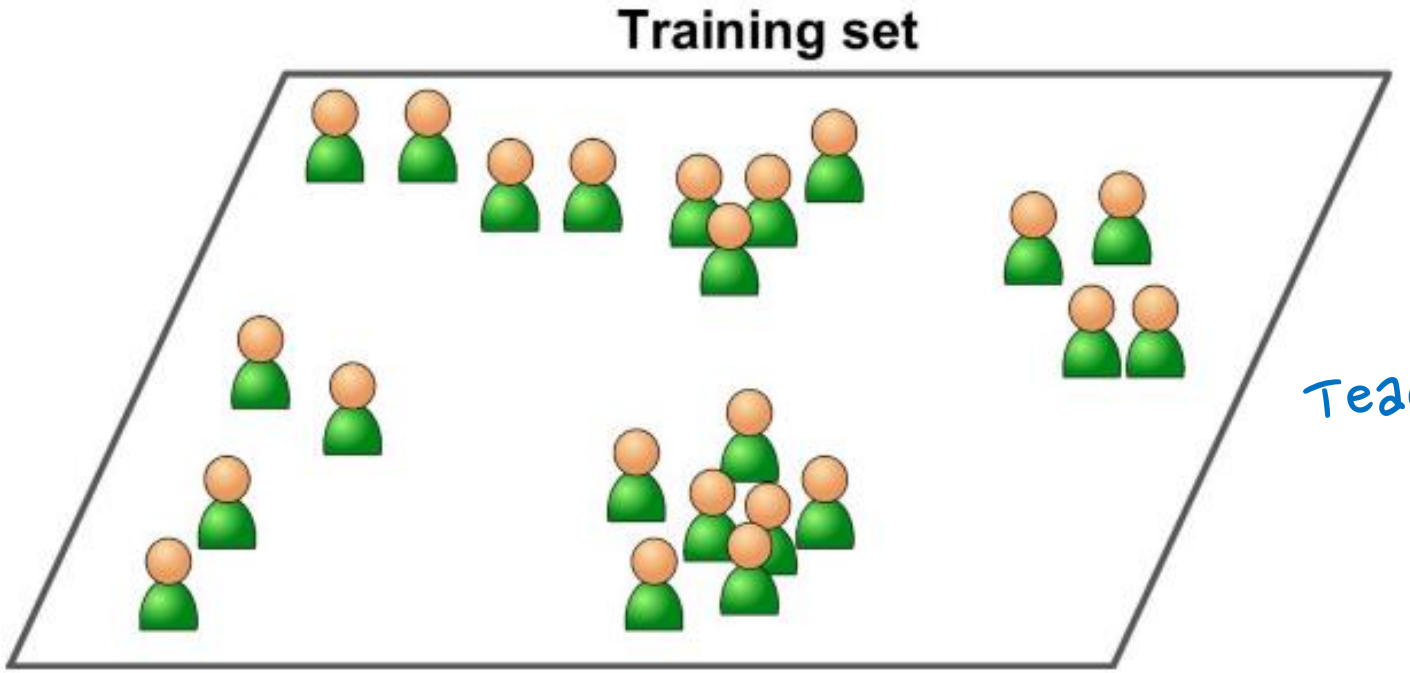
Regression

Here are some of the most important supervised learning algorithms (covered in this book):

- k-Nearest Neighbors
- Linear Regression
- Logistic Regression
- Support Vector Machines (SVMs)
- Decision Trees and Random Forests
- Neural networks

Unsupervised learning

Training set: unlabeled



Teacher 없이 학습한다!

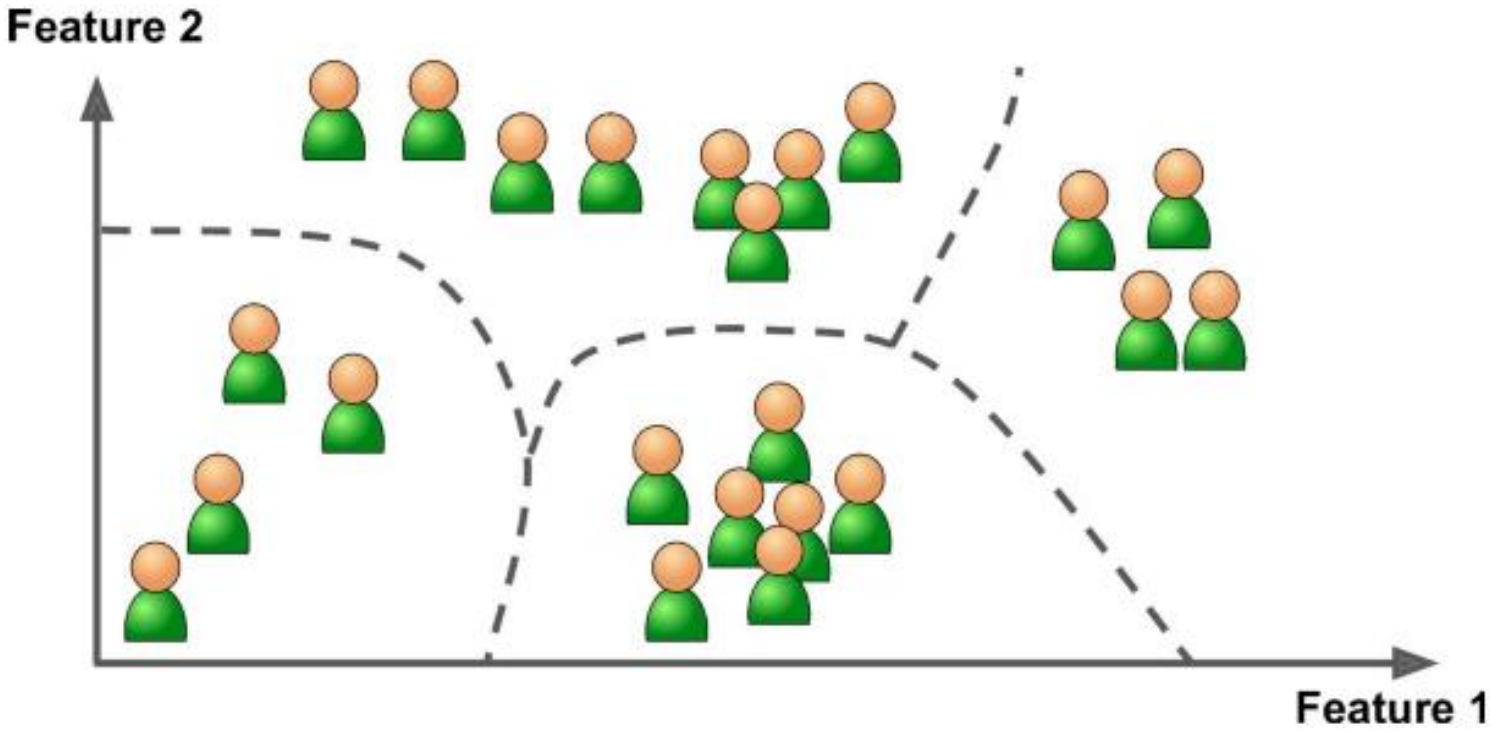
An unlabeled training set for unsupervised learning

Here are some of the most important unsupervised learning algorithms:

- Clustering
  - k-Means
  - Hierarchical Cluster Analysis (HCA)
  - Expectation Maximization
- Visualization and dimensionality reduction
  - Principal Component Analysis (PCA)
  - Kernel PCA
  - Locally-Linear Embedding (LLE)
  - t-distributed Stochastic Neighbor Embedding (t-SNE)
- Association rule learning
  - Apriori
  - Eclat

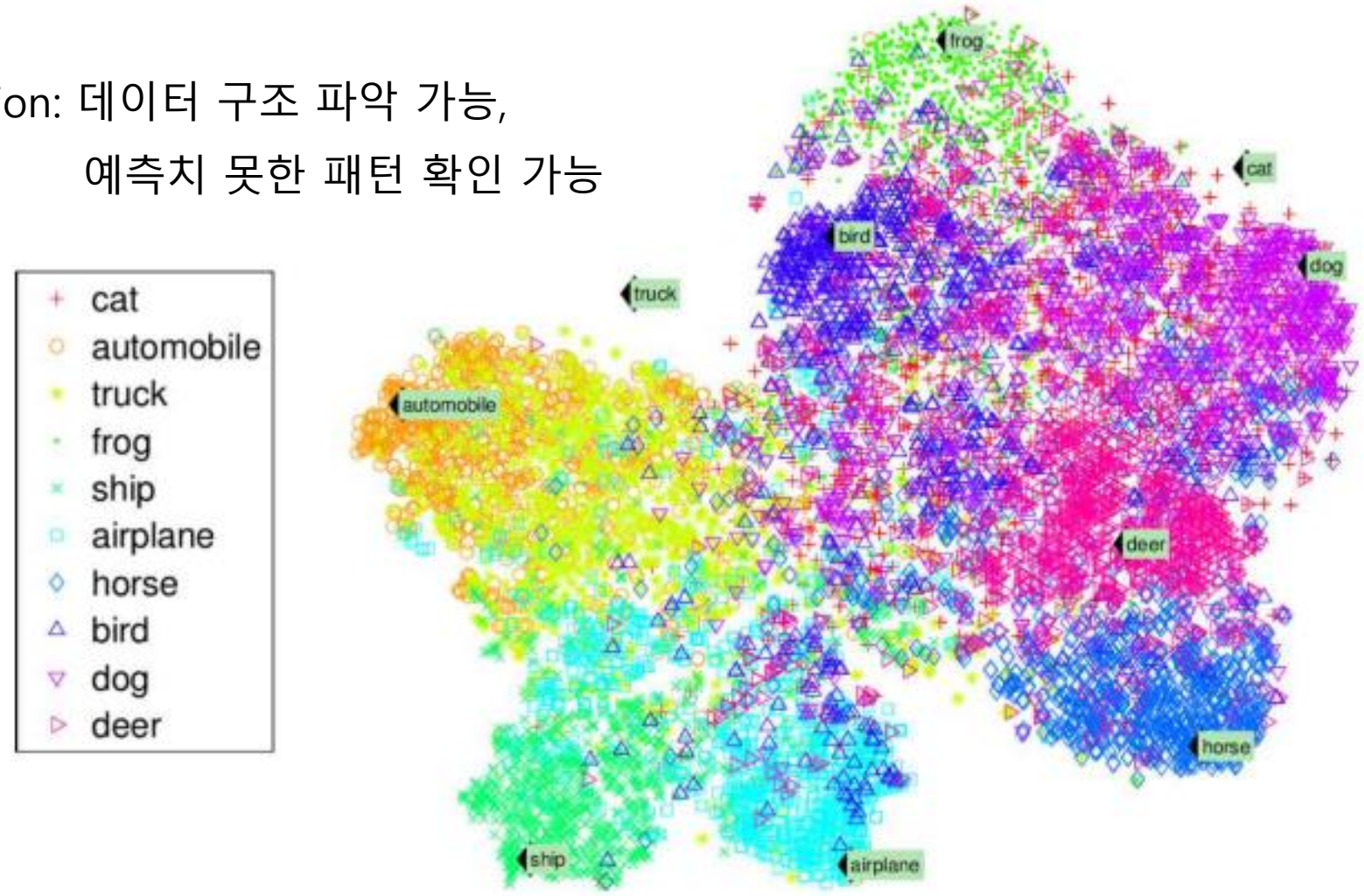
Clustering 알고리즘 예: 블로그 방문객을 더 작은 유사 그룹들로 나눔

- 각 그룹별로 포스트를 다르게 할 수 있다.



Clustering

Visualization: 데이터 구조 파악 가능,
예측치 못한 패턴 확인 가능



Example of a t-SNE visualization highlighting semantic clusters

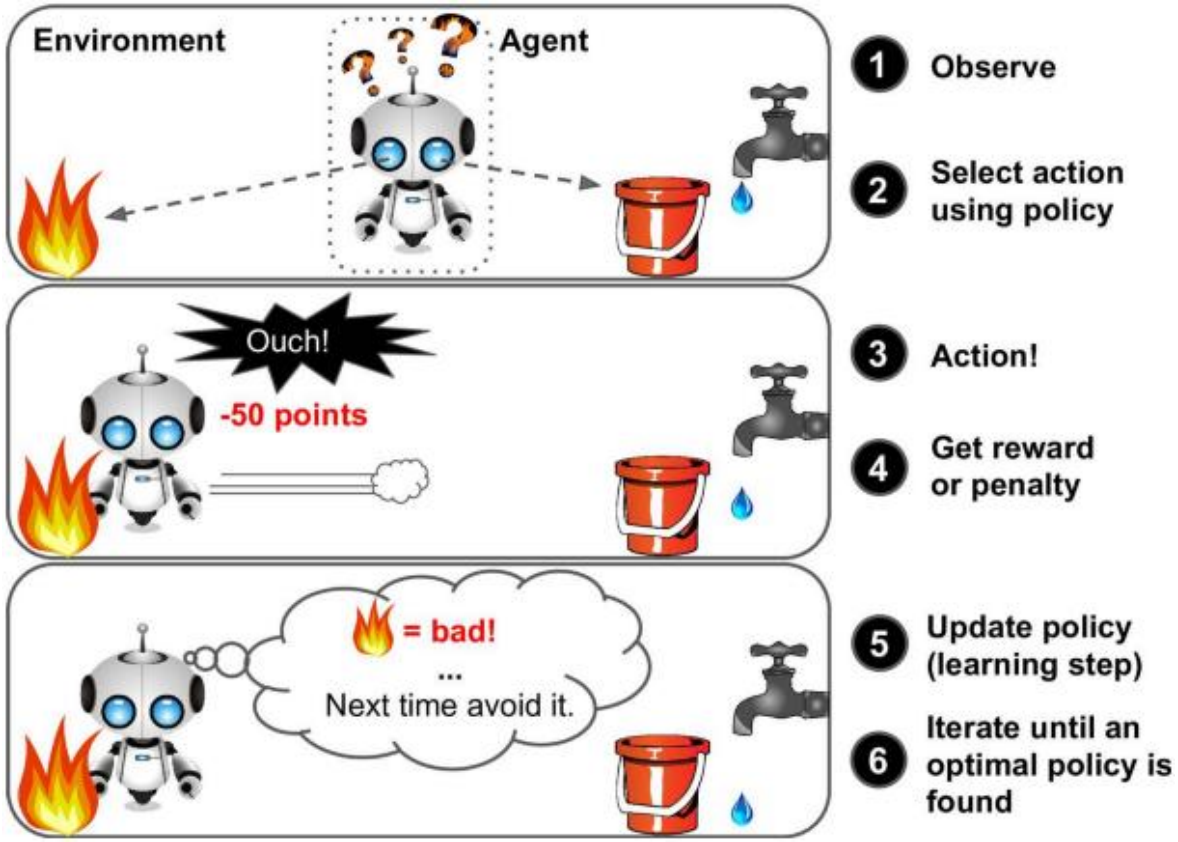
Anomaly detection: 특이한 신용카드 거래 탐지 (사기 방지)

- 정상적인 인스턴스로 학습 => 새 인스턴스 입력 => 정상인지 비정상인지 판단

Anomaly detection

Reinforcement Learning

에이전트라고 불리는 학습 시스템 => 환경을 관찰하고, 행동을 선택하고 수행하며, 그에 대한 대가로 보상 (또는 부정적인 보상의 형태로 처벌)을 받을 수 있다.



시간이 지남에 따라 가장 많은 보상을 얻으려면 정책이라고하는 최고의 전략이 무엇인지 스스로 배워야 한다.

정책은 특정 상황에 있을 때 에이전트가 선택해야하는 작업을 말한다.

Batch and Online Learning

시스템에 입력되는 스트림 데이터를 점진적으로 학습할 수 있는 가에 따라 분류

- Batch learning(offline learning)
- Online learning

Batch learning(Offline learning)

Batch learning

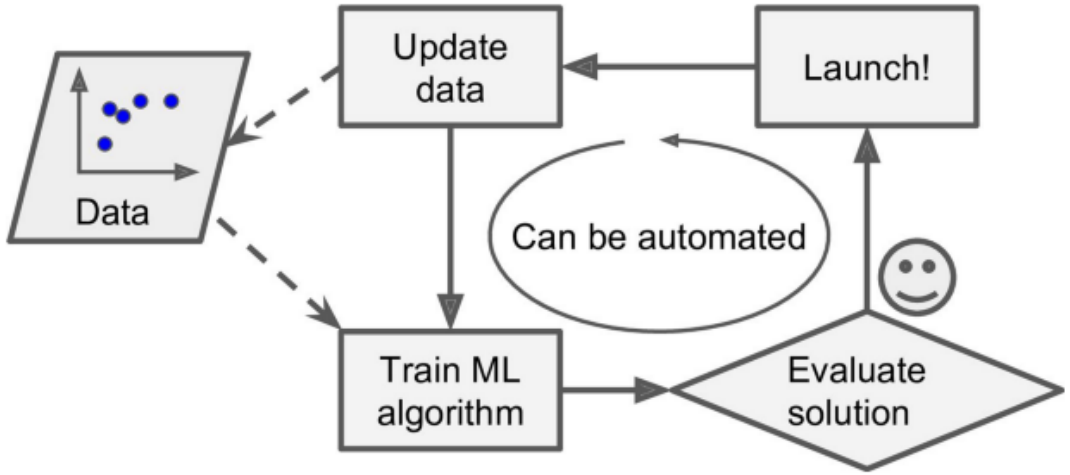
- 시스템이 점진적으로 학습할 수 없음: 이용가능한 모든 데이터를 이용하여 한꺼번에 학습되어야 함
- 많은 시간과 자원이 소요되어야 하므로 전형적으로 offline으로 수행됨
- ① 시스템 학습 => ② 프로덕션으로 론칭 => ③ 더 이상 학습없이 실행; 학습한대로만 수행

새로운 데이터에 대해 학습시키기를 원하면:

=> 기존 데이터 + 새 데이터 => 새 버전의 시스템 학습시킨 후

=> Old 시스템 멈추고 => New 시스템으로 대체

Batch learning(Offline learning)



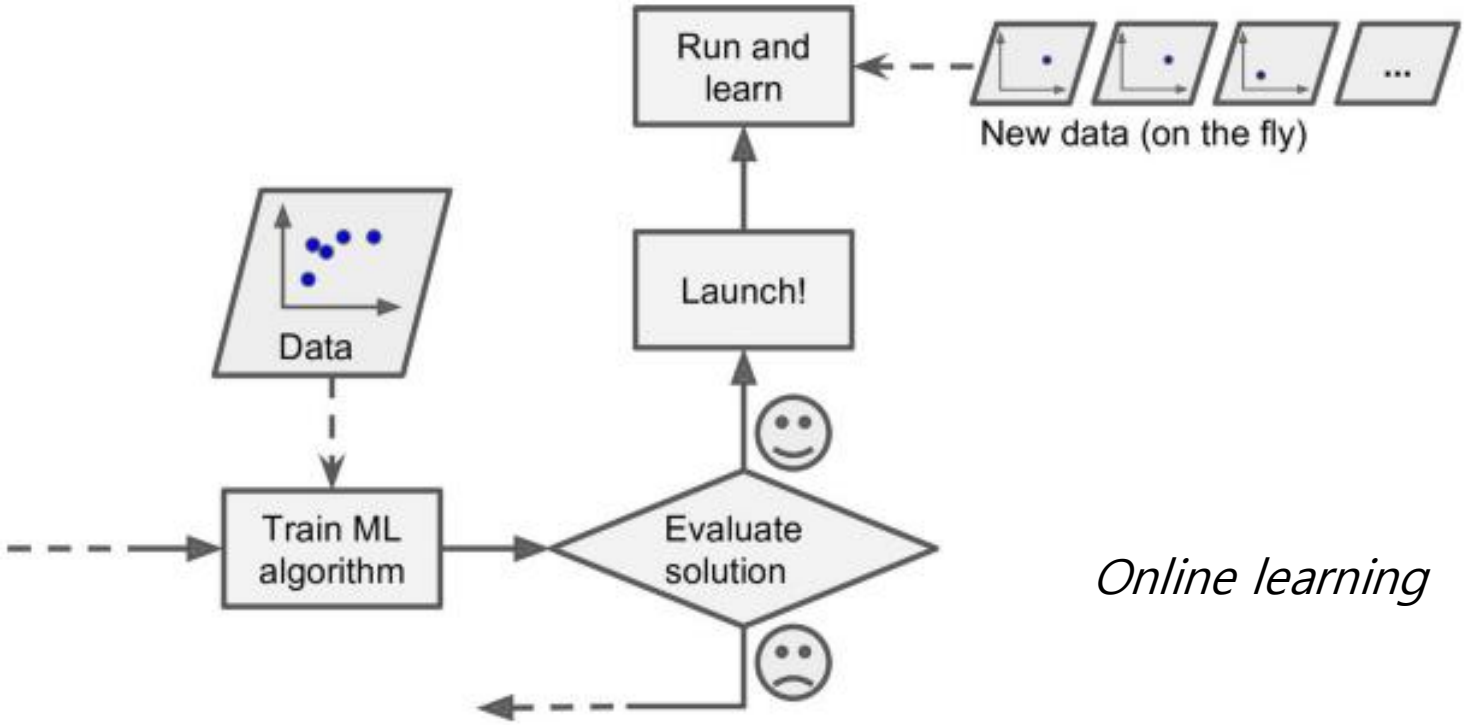
ML 시스템을 training, evaluating, launching하는 전체 과정 => 자동화 가능

- 변화에 적응 가능 => 단순히 데이터 업데이트 => 새로운 버전의 시스템 학습
- 문제점:
  - ✓ 전체 데이터 셋 학습 => 많은 시간 소요 => 보통 24시간 또는 1주일 주기로 학습
- 시스템이 빠르게 변하는 데이터에 적응해야 하는 경우 (예: 주가 예측) => 더 반응적인 솔루션 필요
- 더 좋은 솔루션 => 점진적으로 학습할 수 있는 알고리즘 사용 => Online learning!

Online learning

데이터 인스턴스를 순차적으로 제공함으로써 시스템을 점진적으로 학습시킴

- 학습 단계: fast and cheap, 새로운 데이터가 도착하는 즉시 학습 가능
- 입력 데이터를 연속적인 흐름(예: stock prices)으로 받는 시스템에 적합

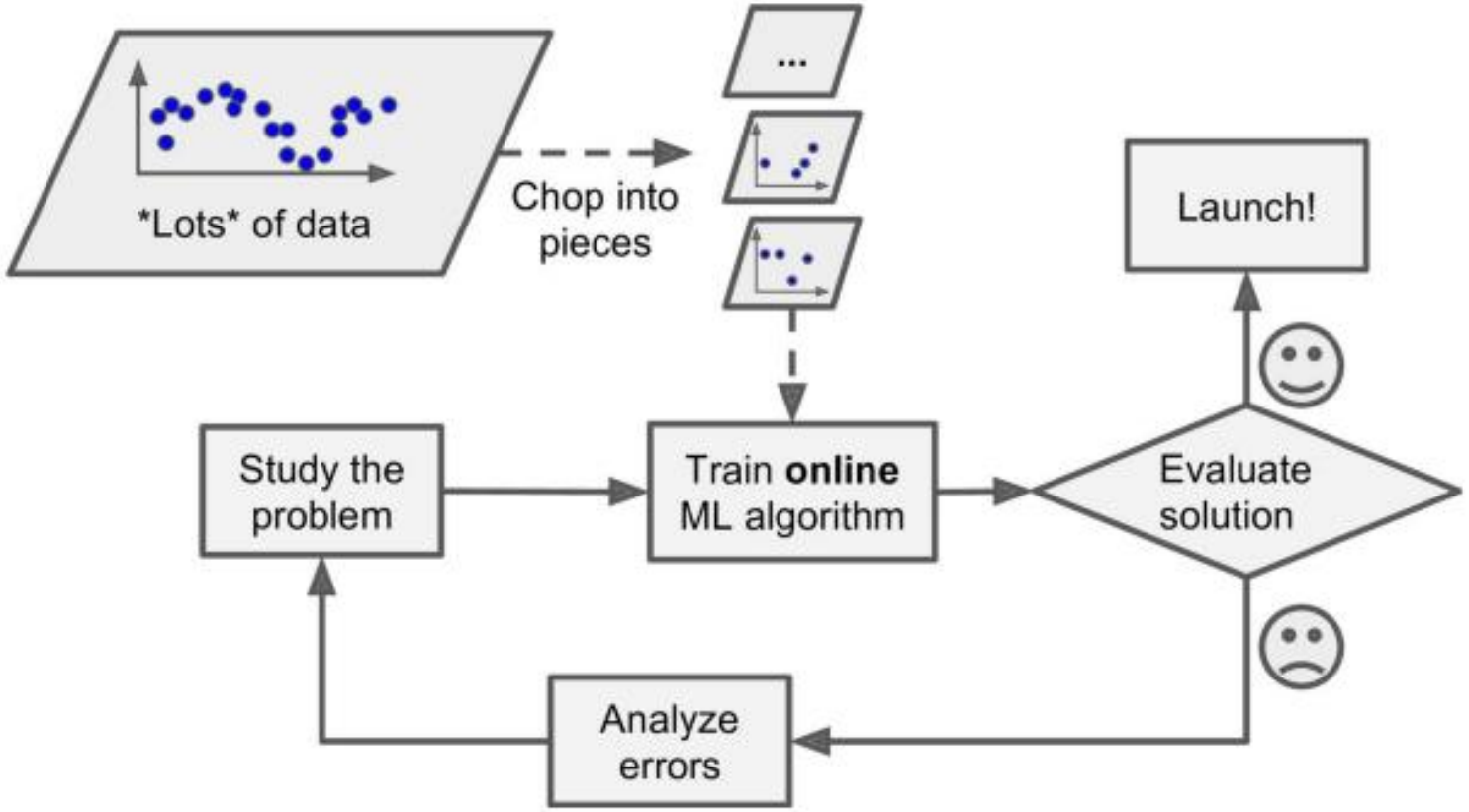


Online learning

Online learning

Online learning: 대규모 데이터 셋에 적용 가능(한꺼번에 메인 메모리에 로딩 불가할 경우)

- 알고리즘: 데이터의 일부분만 로딩, 전체 데이터가 모두 실행될 때까지 반복("incremental learning")
- 학습률(learning rate) => 변화하는 데이터에 얼마나 빠르게 적응하는가를 결정



Using online learning to handle huge datasets

Instance-Based Versus Model-Based Learning

Another ML 시스템 분류 방법: 일반화(generalization)하는 방법에 따라

- 대부분 ML 시스템이 예측을 수행함에 있어서
  - 학습할 때 보지 못했던 예제 데이터(instance)에 대해서도 잘 예측할 수 있어야
  - 학습 데이터에 대해 평가해서 성능이 좋으면 good! => 그러나 충분하지 않음!!
  - 진정한 목표는 새로운 인스턴스에 대해 잘 예측해야

일반화 방법에 대한 2가지 분류: Instance-Based and Model-Based Learning

Instance-Based learning

가장 쉬운 형태의 학습 => 외워서 학습

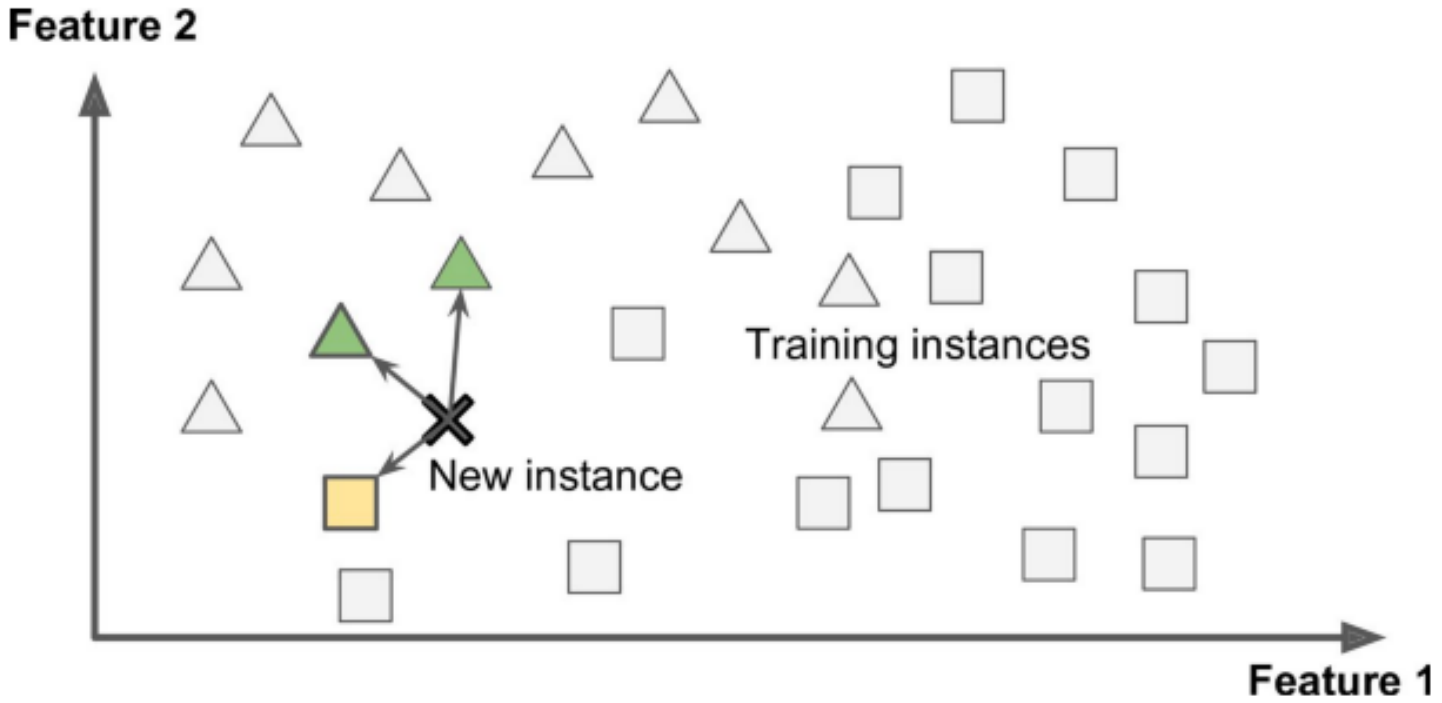
- Spam Filter => 사용자가 이미 spam으로 표시한 이메일과 **동일한** 이메일만 spam으로 분류 가능

개선된 학습 방법 : 알려진 스팸 이메일과 유사한 이메일도 표시하도록 Spam Filter를 프로그램화 가능

- 2개의 이메일 사이에 유사도(measure of similarity) 측정
- 기본적인 유사도 측정 방법 => 2개의 이메일이 공통으로 갖는 단어의 개수 카운트
  - ✓ 새 이메일과 이미 알려진 스팸 이메일의 많은 단어들이 겹치면 새 이메일을 스팸으로 분류

Instance-Based learning:

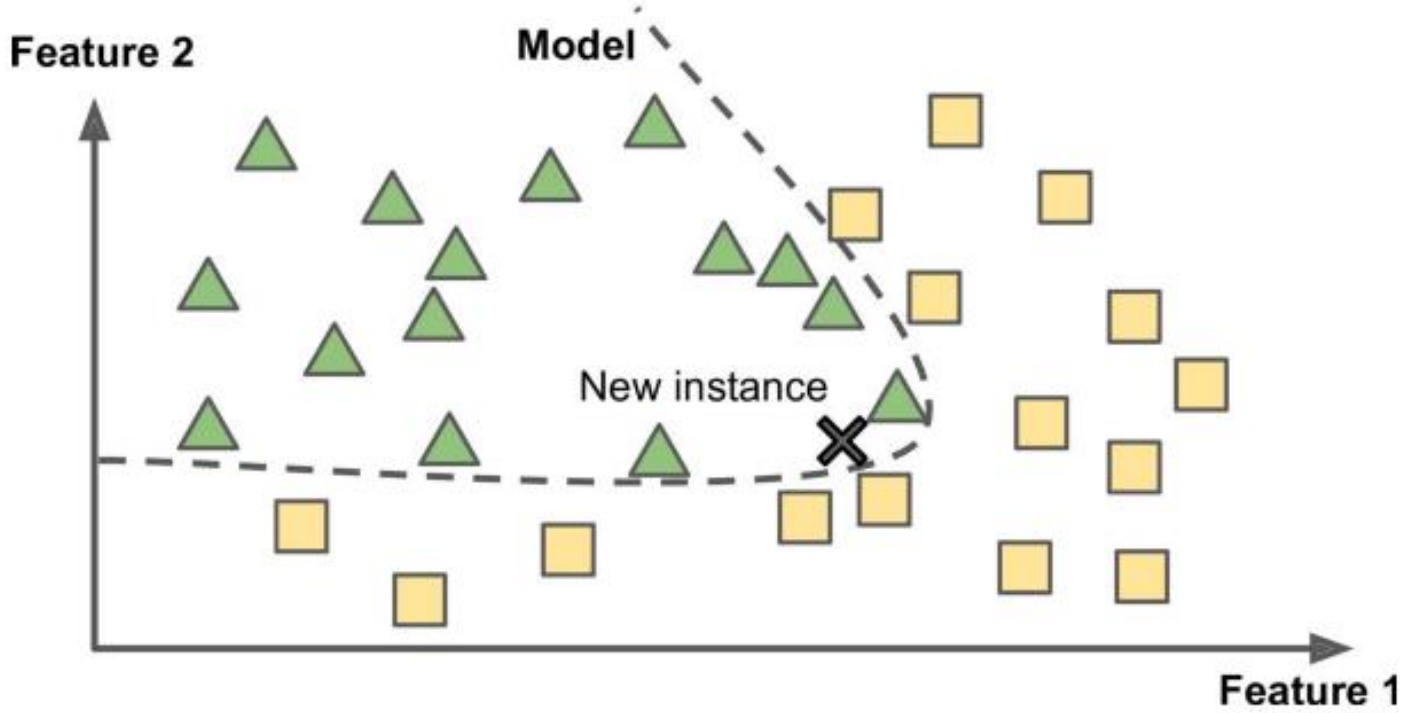
- 훈련 데이터를 외워서 학습 후 => 유사도를 이용하여 새로운 인스턴스를 예측



Instance-Based learning

Model-based learning

일반화의 또 다른 방법: 예제 집합의 모델 구축 => 모델을 이용하여 예측 수행



Model-based learning

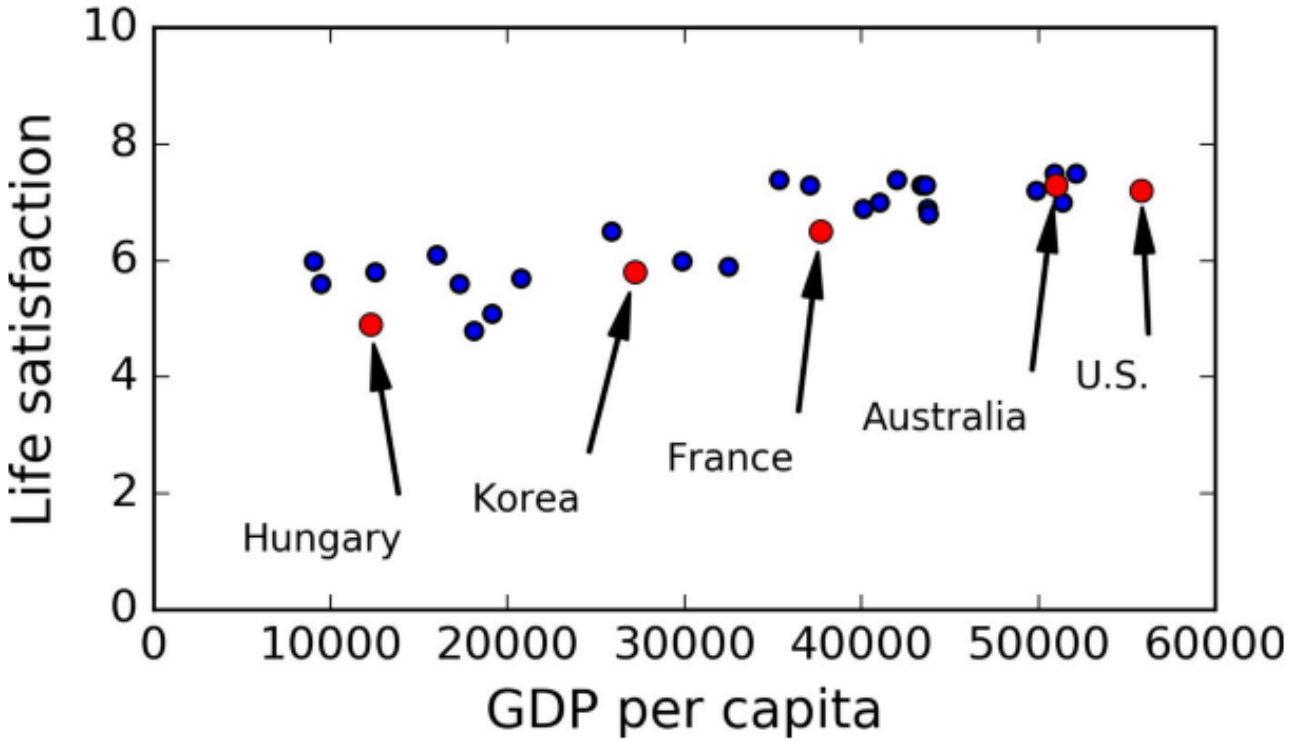
Money가 사람들을 더 행복하게 만드는가를 알고 싶다 하자.

OECD 웹 사이트에서 Better Life Index 데이터를 다운로드하고 IMF 웹 사이트에서 1 인당 GDP에 대한 통계를 다운로드한다. 그런 다음 테이블을 조인하고 1 인당 GDP를 기준으로 정렬한다:

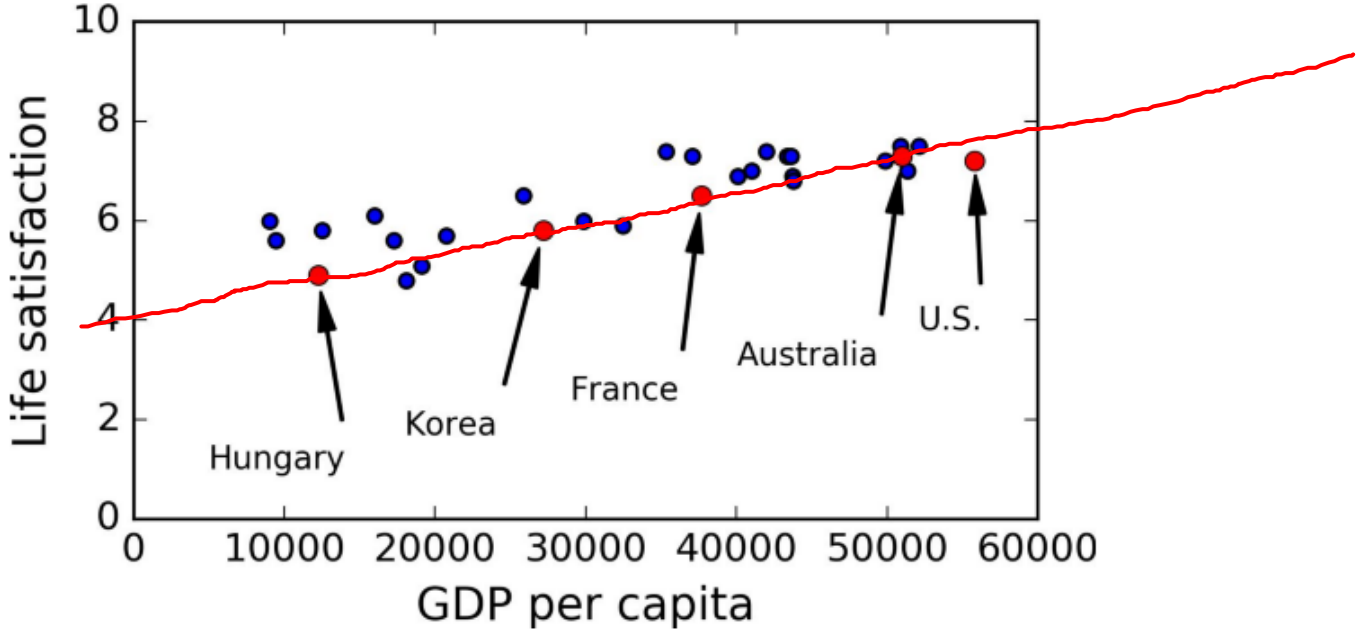
Does money make people happier?

Country	GDP per capita (USD)	Life satisfaction
Hungary	12,240	4.9
Korea	27,195	5.8
France	37,675	6.5
Australia	50,962	7.3
United States	55,805	7.2

Let's plot the data for these countries:



Do you see a trend here?



국가의 1 인당 GDP가 증가함에 따라 삶의 만족도는 다소 선형적으로 증가하는 것처럼 보인다.

⇒ 삶의 만족도 모델을 1 인당 GDP의 선형 함수로 결정한다(모델 선택):

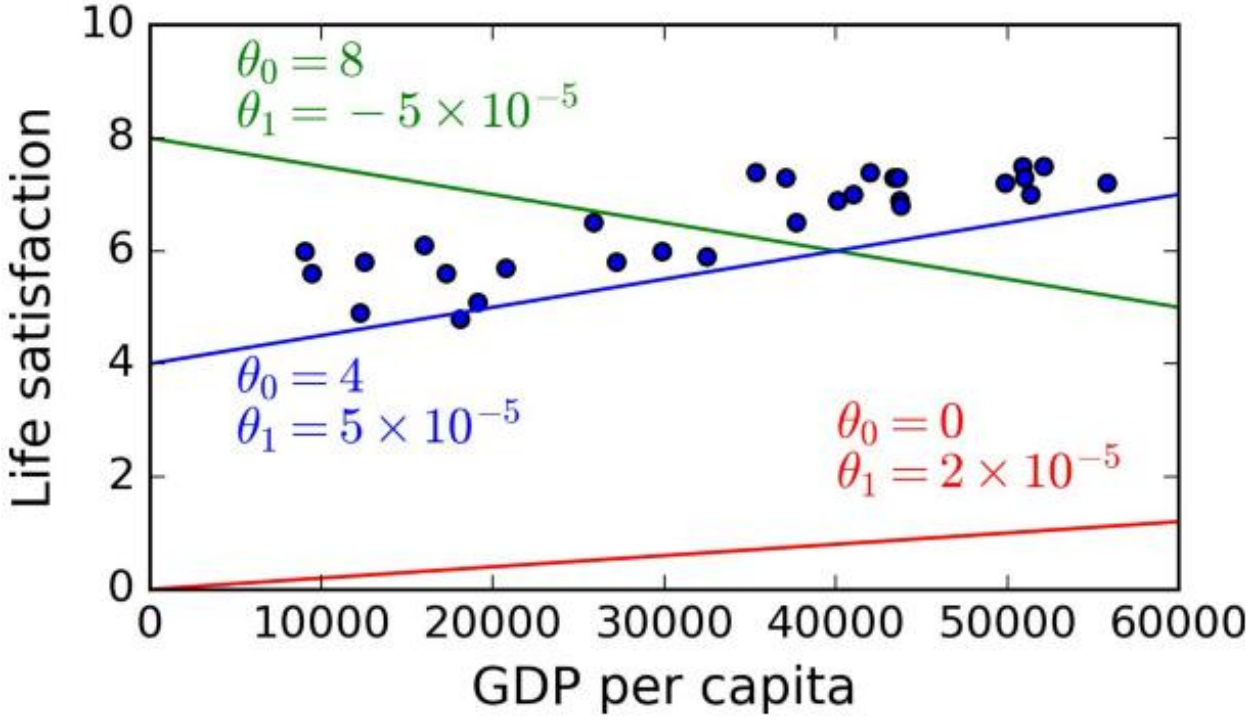
<u>삶의 만족도 Linear Model:</u>

$$\text{life\_satisfaction} = \theta_0 + \theta_1 \times \text{GDP\_per\_capita}$$

2개의 모델 파라메터

A simple linear model: 2개의 파라메터를 가짐 $(\theta_0 \; and \; \theta_1)$

- 2개의 파라메터를 조율함으로써 모델이 임의의 선형 함수를 표현할 수 있게 한다.



A few possible linear models

모델을 사용하기 전에 파라메터 값 θ_0 및 θ_1을 정의해야 한다.

두 파라메터가 어떤 값을 가져야 모델이 최상으로 동작하도록 만들 수 있을까?

이 질문에 대답하기 위해서는 성능 측정 도구를 지정해야 한다:

- Fitness 함수 – how **good** your model is
- Cost 함수 – how **bad** your model is

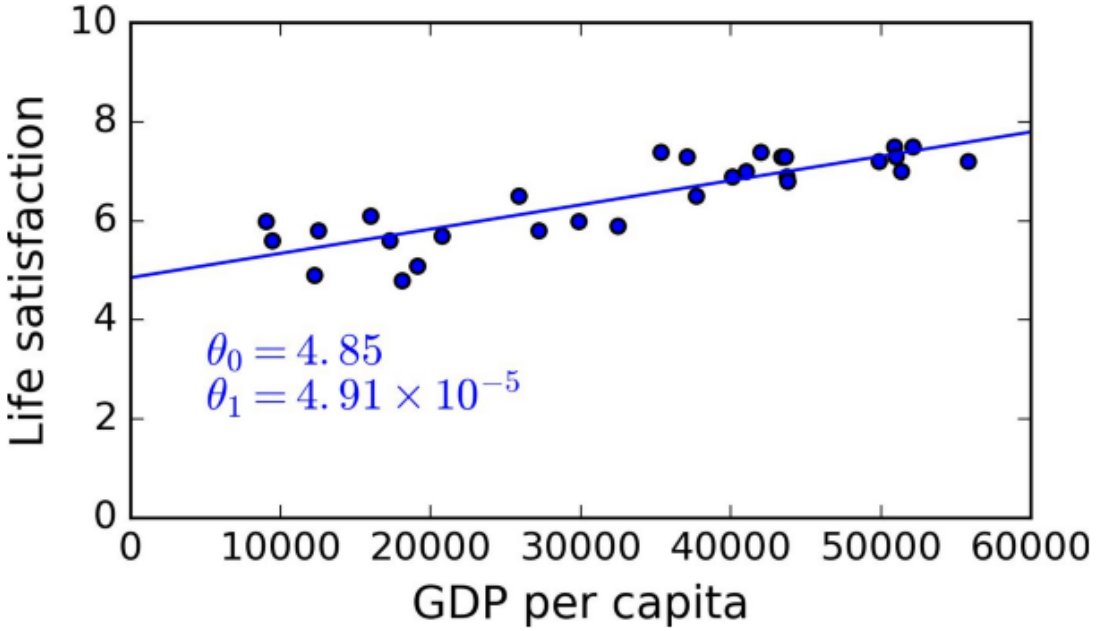
선형회귀 문제의 경우 보통 Cost function 사용

- 선형 모델이 예측한 값과 훈련 예제들의 실제 값 사이의 거리를 측정하는 Cost 함수 사용
- Cost 함수의 목표: 거리 최소화

LR 알고리즘:

- 모델에 훈련 예제를 제공하고 선형 모델을 데이터에 가장 적합하게 만드는 매개 변수를 찾는다.
- 이것을 '모델을 학습시킨다'라고 말한다.
- 최적의 파라메터: $\theta_0 = 4.85 \; and \; \theta_1 = 4.91 \times 10^{-5}$

모델은 훈련 데이터에 가능한 한 가까이 fitting!



The linear model that fits the training data best

이제 모델을 이용하여 예측을 수행할 준비가 되었다.

예로써, 키프로스 사람들이 얼마나 행복한 지 알고 싶다고 하자.

OECD 데이터에는 답이 없다.

다행히도, 모델을 사용하여 좋은 예측을 할 수 있다: 1인당 GDP를 조회해서 22,587 달러를 찾은 다음 모델을 적용하여 삶의 만족도가 약 $4.85+22{,}587\times4.91\times10^{-5}=$**5.96** 정도임을 예측한다.

데이터를 로드하고, 준비시키고, 시각화를 위한 scatterplot을 만든 다음 선형 모델을 학습시키고 예측하는 Python 코드:

```
import matplotlib
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import sklearn.linear_model

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

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

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

# Train the model
model.fit(X, y)

# Make a prediction for Cyprus
X_new = [[22587]]  # Cyprus' GDP per capita
print(model.predict(X_new))
```

실습과제 1-1 *Training and running a linear model using Scikit-Learn*

Load the data, prepare it, create a scatterplot for visualization, and then train a linear model and makes a prediction.

oecd_bli_2015.csv

gdp_per_capita.csv

Download: *https://drive.google.com/drive/folders/1j6juXzCSD79WJAT4AMIJG13a_JA9gzfT?usp=sharing*

(참조: 제 01강 실습과제 #1 The Machine Learning Landscape.pdf)

실습과제 1-1

```
# To support both python 2 and python 3
from __future__ import division, print_function, unicode_literals

# Common imports
import numpy as np
from sklearn import linear_model
import matplotlib.pyplot as plt
import pandas as pd

# Download CSV from http://stats.oecd.org/index.aspx?DataSetCode=BLI
# https://drive.google.com/open?id=1v65V5BNMjlSkzhINhQuj2JOyinkt1Ibl
# datapath = "http://ksamkeun.dothome.co.kr/datasets/lifesat/"
datapath = "datasets/lifesat/"

oecd_bli = pd.read_csv(datapath+"oecd_bli_2015.csv", thousands=',')
oecd_bli = oecd_bli[oecd_bli["INEQUALITY"]=="TOT"]
oecd_bli = oecd_bli.pivot(index="Country", columns="Indicator", values="Value")

print("oecd_bli.head(2):\n")
print(oecd_bli.head(2))
print("\n\noecd_bli['Life satisfaction'].head():\n")
print(oecd_bli["Life satisfaction"].head())

# Download data from http://goo.gl/j1MSKe (=> imf.org)
gdp_per_capita = pd.read_csv(datapath + "gdp_per_capita.csv",thousands=',',delimiter='\t', ',
                             encoding='latin1', na_values="n/a")
gdp_per_capita.rename(columns={"2015": "GDP per capita"}, inplace=True)
gdp_per_capita.set_index("Country", inplace=True)
print("\n\ngdp_per_capita.head(2):\n")
print(gdp_per_capita.head(2))
```

실습과제 1-1

```
31
32  full_country_stats = pd.merge(left=oecd_bli, right=gdp_per_capita, left_index=True, right_index=True)
33  # print(full_country_stats)
34  # print(full_country_stats[["GDP per capita", 'Life satisfaction']].loc["United States"])
35
36  keep_indices = list(set(range(36)))
37  sample_data = full_country_stats[["GDP per capita", 'Life satisfaction']].iloc[keep_indices]
38
39  def prepare_country_stats(oecd_bli, gdp_per_capita):
40      return sample_data
41
42  # Prepare the data
43  country_stats = prepare_country_stats(oecd_bli, gdp_per_capita)
44  X = np.c_[country_stats["GDP per capita"]]
45  y = np.c_[country_stats["Life satisfaction"]]
46
47  # Visualize the data
48  country_stats.plot(kind='scatter', x="GDP per capita", y='Life satisfaction')
49  plt.show()
50
51  # Select a linear model
52  model = linear_model.LinearRegression()
53
54  # Train the model
55  model.fit(X, y)
56
57  # Make a prediction for Cyprus
58  print("\n\n# Make a prediction for Cyprus\n")
59  X_new = np.array([[22587.0]])  # Cyprus' GDP per capita
60  print(model.predict(X_new))
```

```
C:\Users\sam\Anaconda3\envs\ai\python.exe F:/practice/ai/handson(pycharm)/ch01-(1.1)-better-life-index.py
oecd_bli.head(2):

Indicator  Air pollution         ...          Years in education
Country                          ...
Australia           13.0         ...                        19.4
Austria             27.0         ...                        17.0

[2 rows x 24 columns]


oecd_bli['Life satisfaction'].head():

Country
Australia    7.3
Austria      6.9
Belgium      6.9
Brazil       7.0
Canada       7.3
Name: Life satisfaction, dtype: float64


gdp_per_capita.head(2):

                                    Subject Descriptor          ...          Estimates Start After
Country                                                         ...
Afghanistan  Gross domestic product per capita, current prices  ...                         2013.0
Albania      Gross domestic product per capita, current prices  ...                         2010.0

[2 rows x 6 columns]


# Make a prediction for Cyprus

[[6.28653637]]

Process finished with exit code 0
```