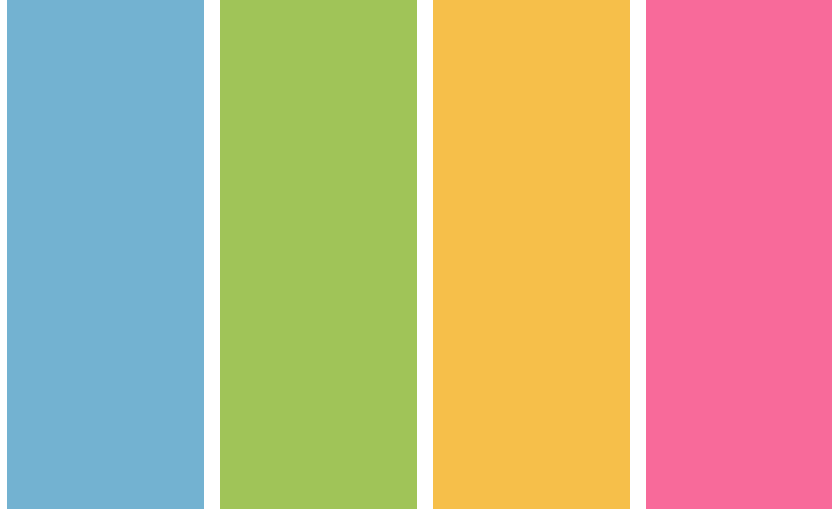


01 Machine Learning



1. 머신러닝
2. 머신러닝과 딥러닝
3. 머신러닝 프로젝트



1. 머신러닝

AI vs. ML vs. DL

1. MACHINE LEARNING

인공지능 \supset 머신러닝 \supset 딥러닝
머신러닝은 인공지능의 일부, 딥러닝은 머신러닝의 일부

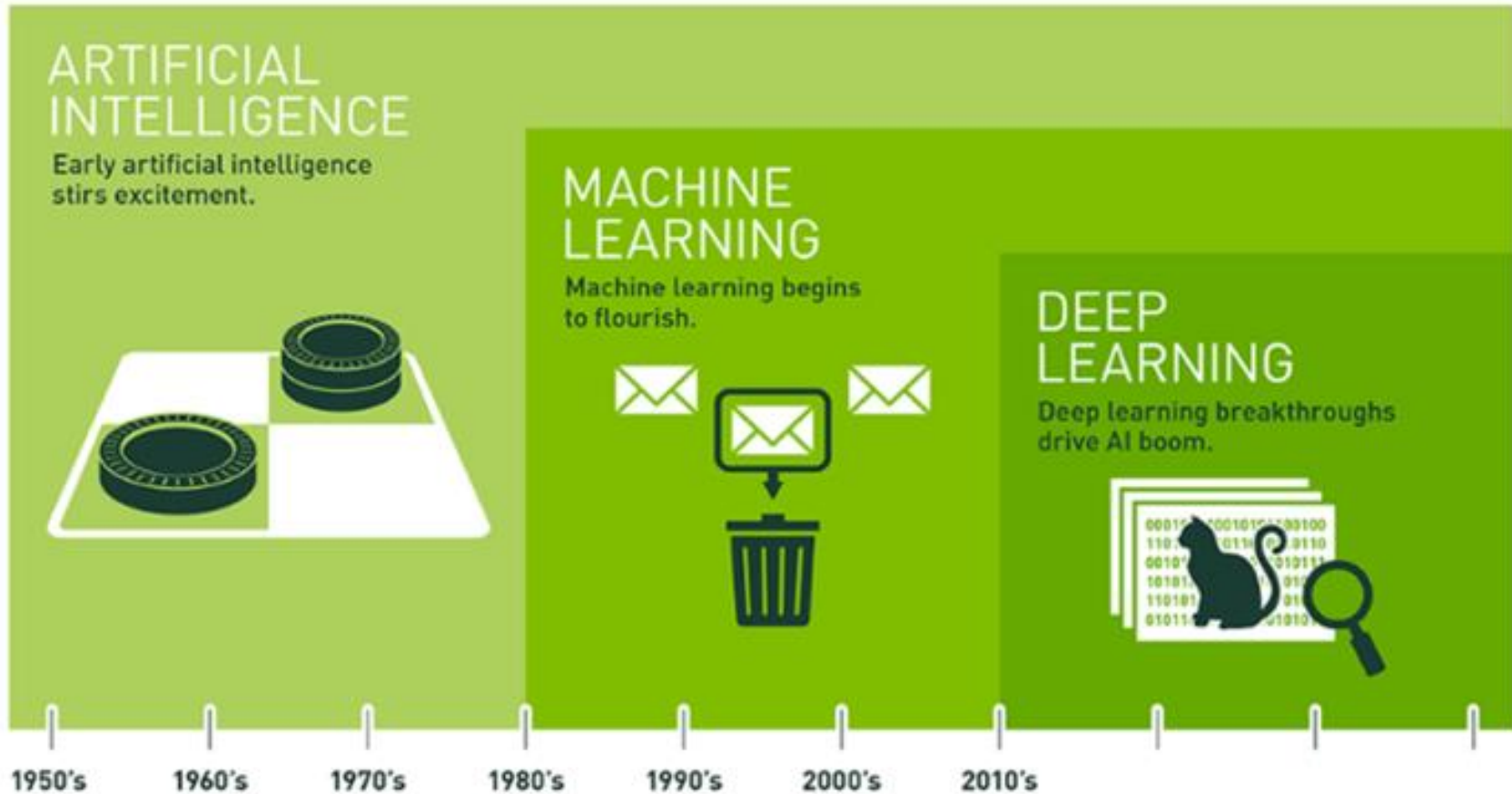
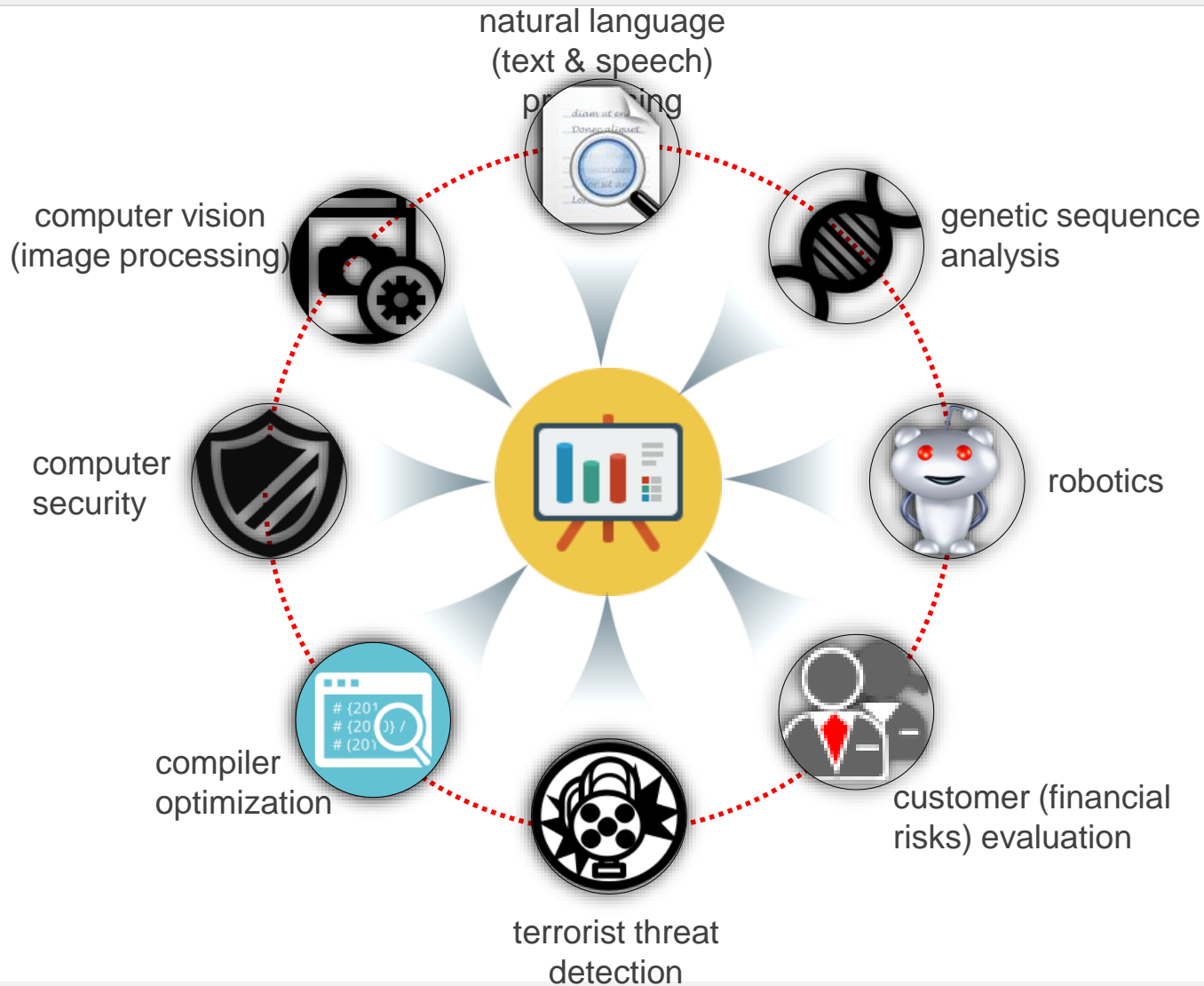


그림 출처:

http://blogs.nvidia.co.kr/2016/08/03/difference_ai_learning_machinelearning/

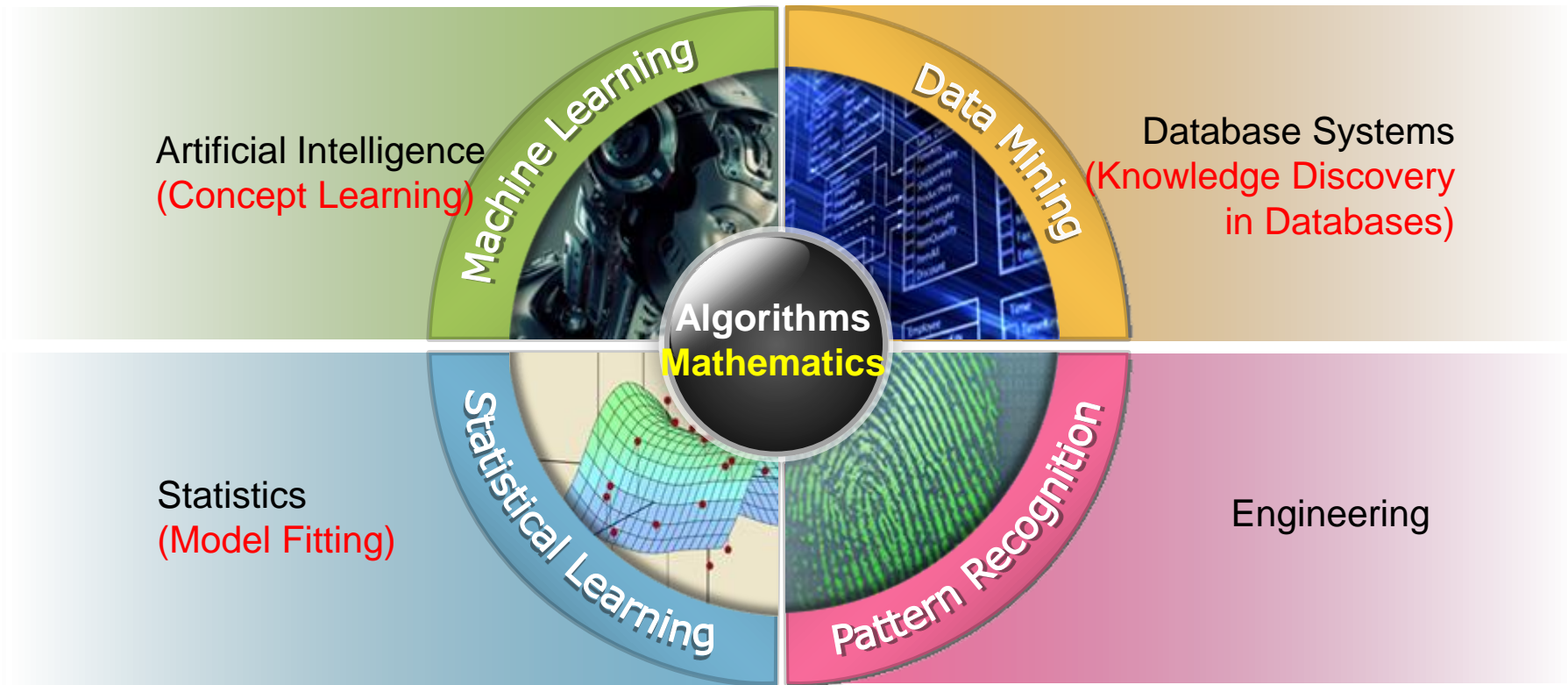
머신러닝이 어디에 좋을까?

1. MACHINE LEARNING



ML - A Multi-domain view

1. MACHINE LEARNING



Machine Learning

1. MACHINE LEARNING



ML :상황을 인지하고 판단을 수행하는 분야에 꼭 필요한 모형

- 카드회사에서 이 카드 사용 패턴이 정상인가, 부정사용인가를 판단해서 스스로 승인여부를 결정
- 자율주행 자동차는 주변 상황 정보를 독립변수로 입력받아 운전엔 필요한 의사결정을 수행



모형 설정, 학습, 실제 상황에서 판단 수행이라는 단계로 진행

- 종속변수는 클래스를 나타내는 질적자료, 독립변수는 이 클래스의 특징을 나타내는 양적자료
- ML 모형은 독립변수를 이용하여 종속변수의 클래스를 예측하는 방법론



분야의 모형은 전통적 통계 모형과 기계학습모형(ML)으로 나뉨

- 통계적인 모형 : 판별분석, 로지스틱 회귀모형, Tree, k-NN 모형이 대표적
- 기계학습 모형 : 신경망모형, SVM, XGBoost 등.



ML 모형은 통계적 모형에 비해 비교적 좋은 예측 결과를

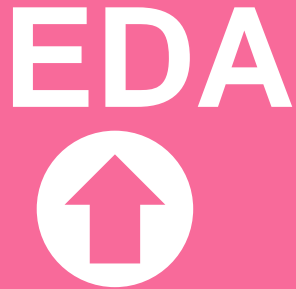
- Feature Variable의 수를 크게 하고 판별에 필요한 함수도 특별하게 설정하지 않아 학습 상황에서는 좋은 예측 결과를 주는 경우가 많음

ML 장점과 단점

- 장점 : Feature Variable의 설정에 크게 신경을 쓰지 않아도 좋은 추정을 준다.
- 단점 : 계산량이 방대하고 결과를 사람이 이해할 수 없기 때문에 적용 상 이슈가 많은 단점을 가지고 있어 모형의 해석이 필요한 분야에서는 사용이 어렵다

EDA와 CDA는 ML과 다르다.

1. MACHINE LEARNING



탐색적 자료 분석(Exploratory Data Analysis)

데이터의 특징과 내재하는 구조적인 관계를 알아내기 위한 분석 기법으로 이러한 자료의 탐색 과정을 통하여 얻은 정보를 기초로 통계모형을 세울 수 있음

미지의 특성을 파악하고 자료구조를 파악할 수 있는 증거 수집의 과정

Looking at data to see what it seems to say. It's concentrates on simple arithmetic and easy-to-draw picture. *John Tukey, 1977*

확증적 자료 분석(Confirmatory Data Analysis)

관측된 자료의 형태로 효과의 재현성을 평가하고 추정하는 전통적인 분석 과정, 신뢰구간의 추정이나 유의성 검정 등이 여기에 해당됨, 수집된 정보와 증거에 대한 차분한 실증적 평가에 중점을 뒀서 결론을 유도한다



ML - What is Machine Learning?

1. MACHINE LEARNING

ML studies algorithms that *improve with* experience.
learn from

Tom. Mitchell (1997, Definition of the [general] learning problem)

A computer program is said to *learn* from *experience E* with respect to some class of *tasks T* and *performance measure P*, if its performance at tasks in T, as measured by P, improves with experience E -



<http://www.edtpatips.com/wp-content/uploads/2014/12/teaching-strategy-184317176-1440x1008.jpg>

Example: A program for soccer tactics

T : Win the game

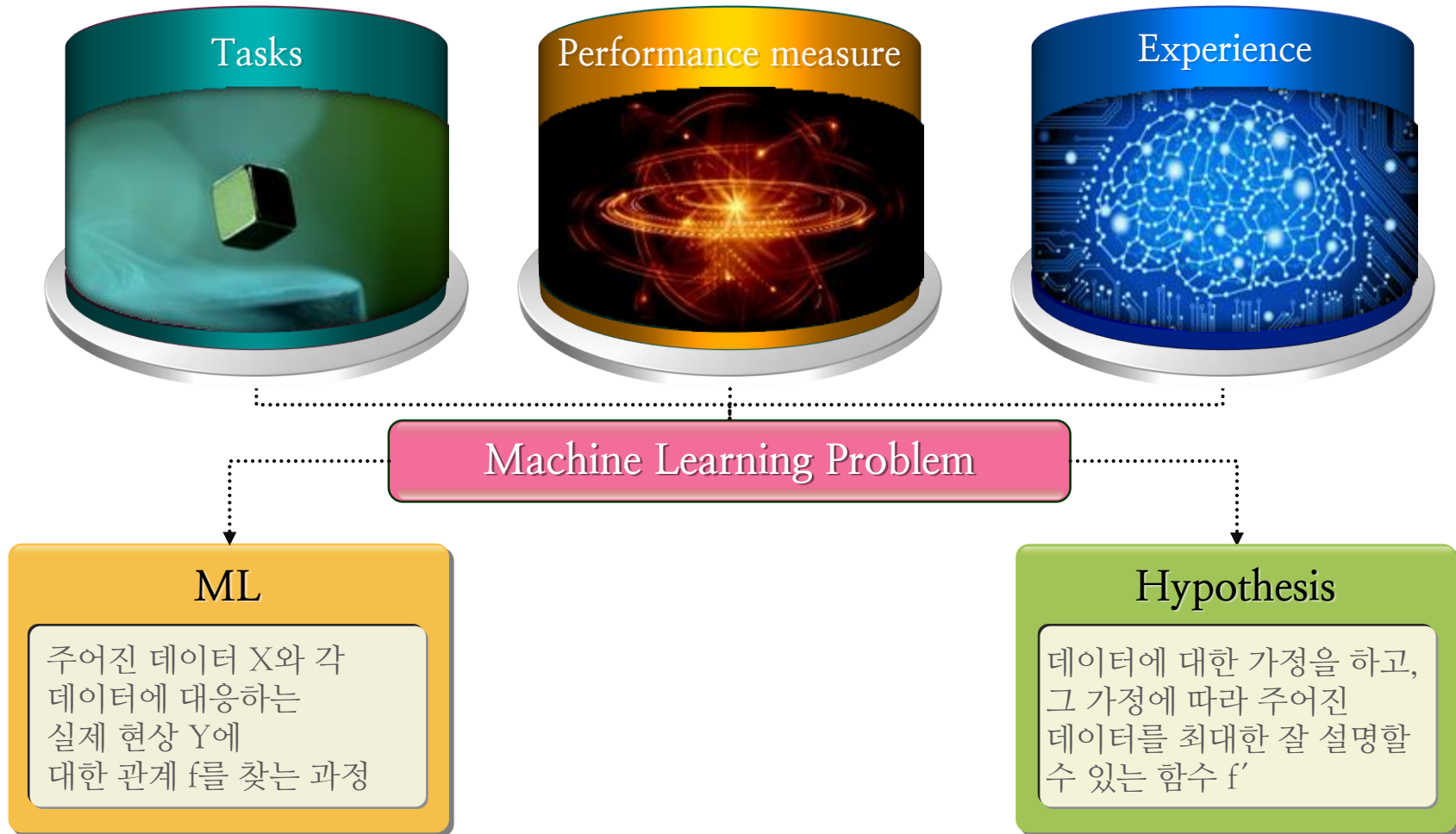
P : Goals

E : (x) Players' movements, (y) Evaluation

ML - T, P & E

1. MACHINE LEARNING

T를 달성하는 데 있어서 E를 통해 P를 향상시킨다.



ML - Tasks

1. MACHINE LEARNING

Tom. Mitchell(1997, Definition of the [general] learning problem)

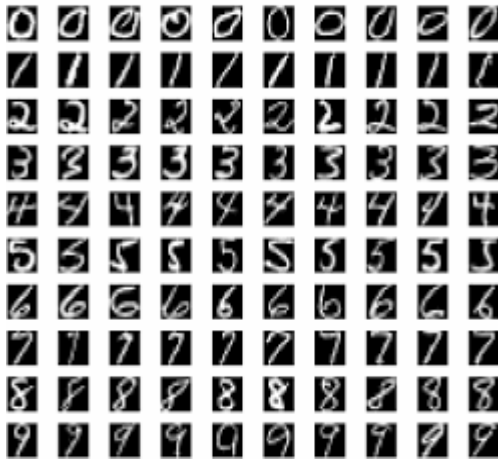
A computer program is said to *learn* from *experience E* with respect to some class of *tasks T* and *performance measure P*, if its performance at tasks in T, as measured by P, improves with experience E -

Classification

discrete target values

x : pixels (28×28)

y : 0,1,2,3,4,5,6,7,8,9

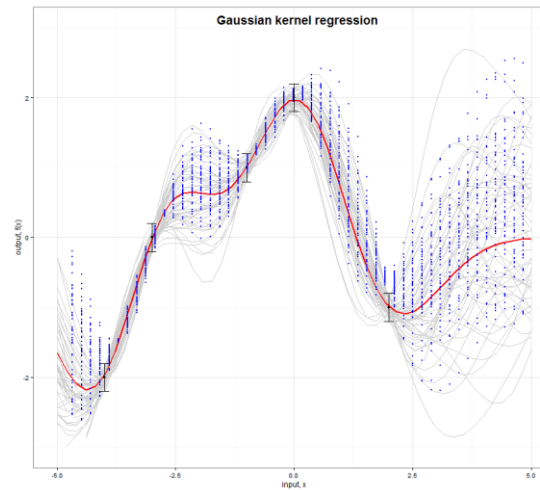


Regression

real target values

$x \in (0,100)$

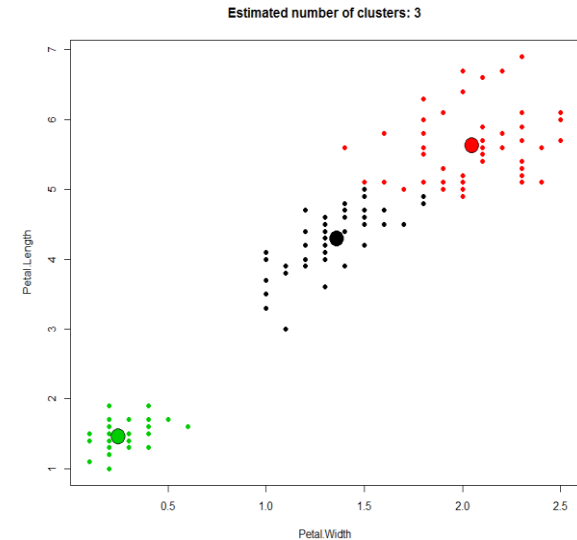
y : 0,1, 2,3,...,9



Clustering

no target values

$x \in (-3,3) \times (-3,3)$



ML - Performance measure

1. MACHINE LEARNING

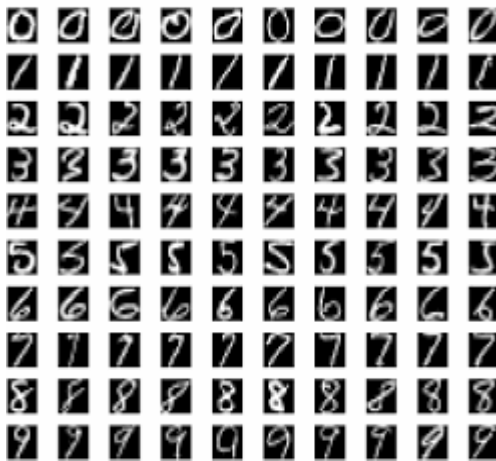
Tom. Mitchell(1997, Definition of the [general] learning problem)

A computer program is said to *learn* from *experience E* with respect to some class of *tasks T* and *performance measure P*, if its performance at tasks in T, as measured by P, improves with experience E -

Classification

0-1 loss function

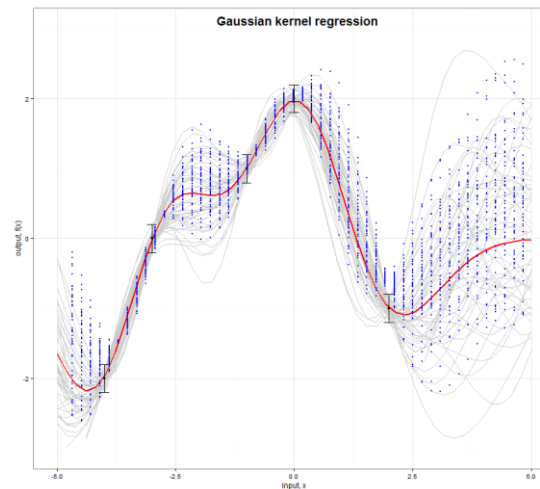
$$L(\hat{y}, y) = I(\hat{y} \neq y)$$



Regression

L2 loss function

$$L(f, \hat{f}) = \|f - \hat{f}\|_2^2$$

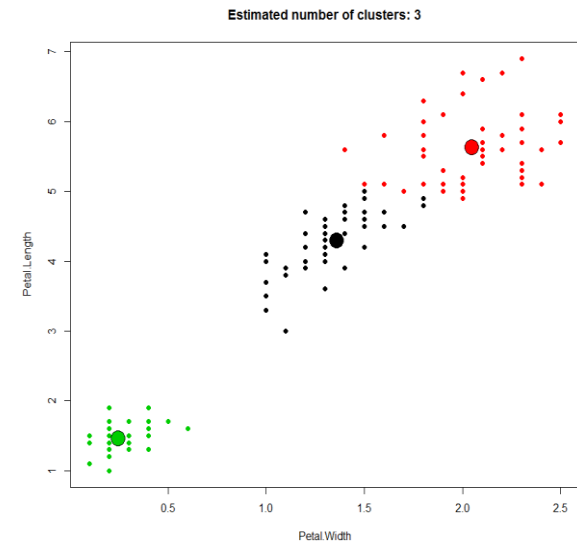


Clustering

no target values

$$L(y, \hat{y}) = I(\hat{y} \neq y)$$

$$L(f, \hat{f}) = \|f - \hat{f}\|_2^2$$



ML - Experience

1. MACHINE LEARNING

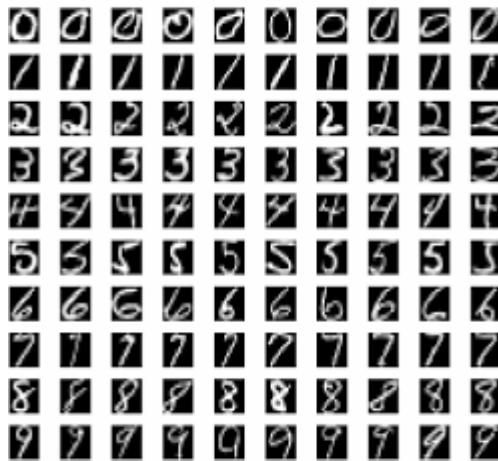
Tom. Mitchell(1997, Definition of the [general] learning problem)

A computer program is said to *learn* from *experience E* with respect to some class of *tasks T* and *performance measure P*, if its performance at tasks in T, as measured by P, improves with experience E -

Classification

labeled data

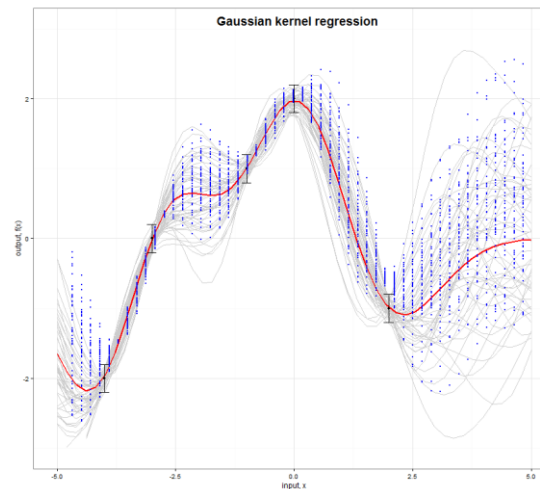
(pixels) \rightarrow (number)



Regression

labeled data

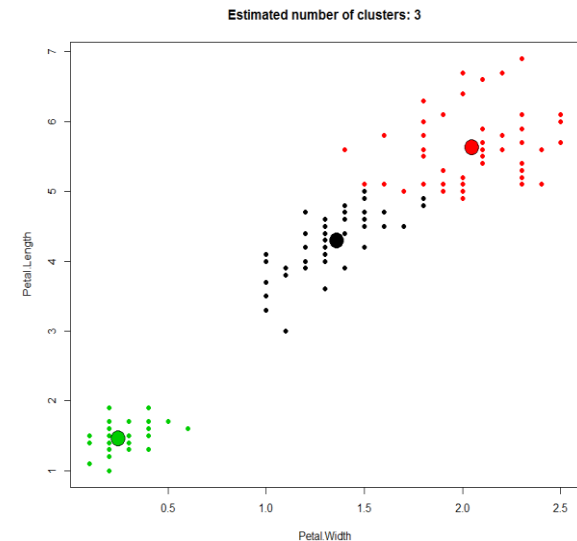
$(x) \rightarrow (y)$



Clustering

unlabeled data

(x_1, x_2)



지도학습 vs. 비지도학습(자율학습)

1. MACHINE LEARNING

Supervised Learning

Supervised Learning

Estimate an unknown mapping from known input and target output pairs

Learn f_w from training set $D=\{(x, y)\}$ s.t.

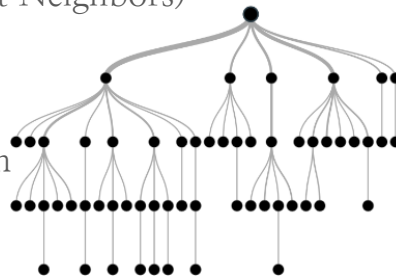
$$f_w(x)=xy = f(y)$$

- Classification : y is discrete
- Regression : y is continuous

Classification

Inputs are divided into two or more classes

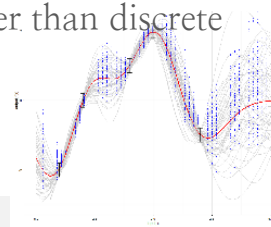
- SVM (Support Vector Machine)
- K-NN (k-Nearest Neighbors)
- Naïve Bayes
- Decision Tree
- Random Forest
- Logistic Regression
- Neural Network



Regression

Outputs are continuous rather than discrete

- Linear Regression
- K-NN
- SVM
- Random Forest



Unsupervised Learning

Unsupervised Learning

Only input values are provided

Learn f_w from $D=\{(x)\}$ s.t. $f_w(x)=x$

- Clustering
- Association

Clustering & Dimension Reduction

A set of inputs is to be divided into groups the group are now known beforehand

- K-means
- Hierarchical clustering
- PCA(Principal Component Analysis)
- Neural Network

Association

End to end connection

- Apriori (arules package)



Machine Learning - Reinforcement learning

1. MACHINE LEARNING



“Toward learning robot table tennis”, J. Peters et al. (2012)

<https://youtu.be/SH3bADiB7uQ>

Machine Learning - Clustering

1. MACHINE LEARNING

Unsupervised Learning의 일종

Label 데이터 없이 주어진 데이터들을 가장 잘 설명하는 Cluster를 찾는 문제

클러스터링이 필요한 이유

Classification을 하기 위해서는 데이터와 각각의 데이터의 Label이 필요

But, 실제로는 데이터는 존재하지만 그 데이터의 Label이나 Category가 무엇인지 알 수 없는 경우

Classification이 아닌 다른 방법을 통해 데이터들을 설명해야 하는 경우가 발생

$$\min_{b,w} \sum_i^n \sum_j^k w_{ij} \|x_i - b_j\|_2^2 \text{ s.t. } \sum_j w_{ij} = 1, \forall j$$

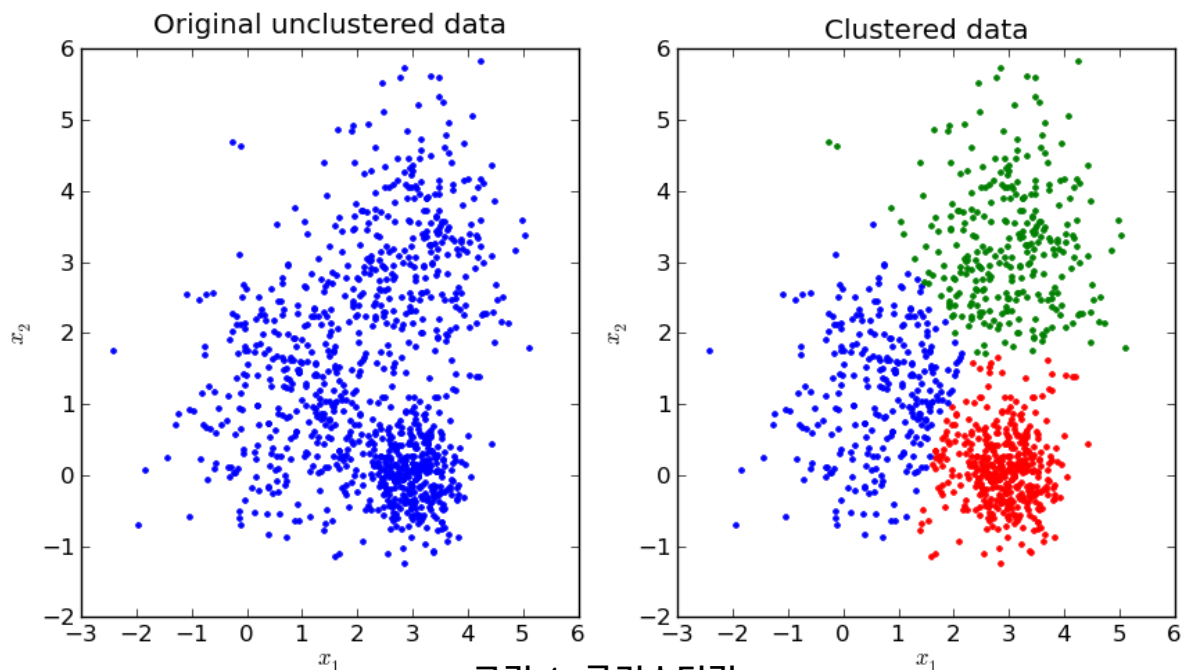
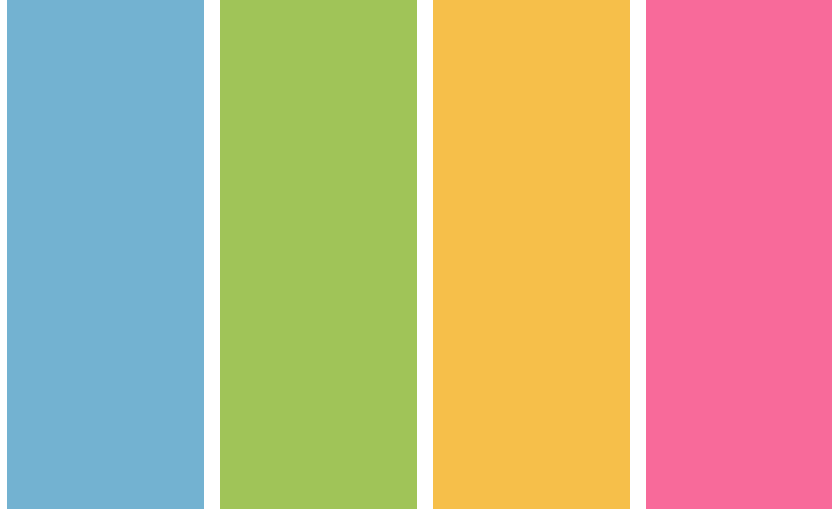


그림 1. 클러스터링

데이터 분석에서 가장 중요한 것은?

1. MACHINE LEARNING





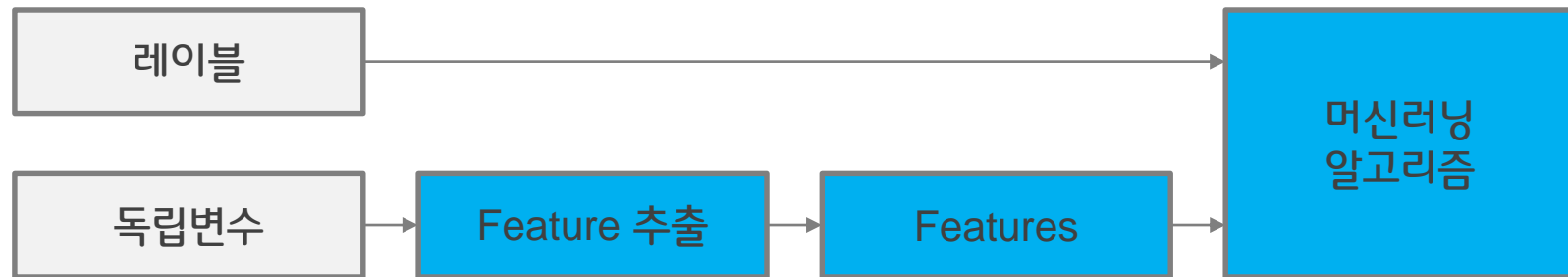
2. 머신러닝과 딥러닝

머신러닝 vs. 딥러닝

2. 머신러닝과 딥러닝

- 딥러닝은 학습데이터에서 주요 Feature를 추출/선택하는 과정까지도 학습

머신러닝 훈련 과정



딥러닝 훈련 과정



이해를 돕기 위해서... 이거는 ML

2. 머신러닝과 딥러닝

$$\begin{aligned}a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n &= b_1 \\a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n &= b_2 \\&\dots \\a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n &= b_n\end{aligned}$$



$$\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix}$$



간소화

$$AX = B$$



X를 구하려면?

$$X = A^{-1}B$$

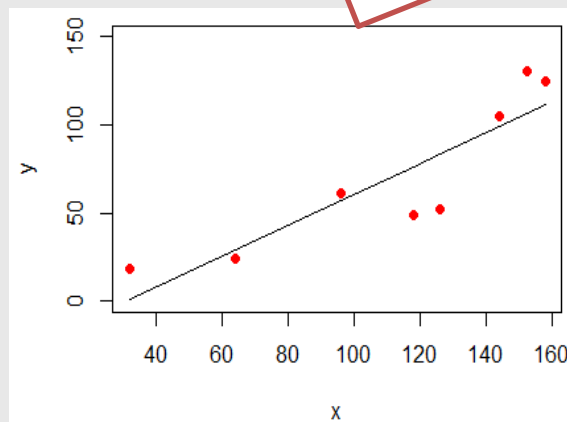


$$\begin{bmatrix} a \\ b \end{bmatrix} = (A^T A)^{-1} A^T B$$

A가 정방행렬이 아니고 행의 수가 열의 수보다 크므로 left pseudo inverse를 이용

```
> x <- c(32,64,96,118,126,144,152.5,158)
> y <- c(18,24,61.5,49,52,105,130.3,125)
> plot(x, y, col=2, pch=19, ylim=c(0,150));
> (A <- matrix(c(x,rep(1,NROW(x))), ncol=2))
```

```
[,1] [,2]
[1,] 32.0 1
[2,] 64.0 1
[3,] 96.0 1
[4,] 118.0 1
[5,] 126.0 1
[6,] 144.0 1
[7,] 152.5 1
[8,] 158.0 1
```



```
> (ab <- solve(t(A)%*%A) %*% t(A) %*% matrix(y, ncol=1))
[,1]
[1,] 0.8749313
[2,] -26.7907862
> lines(x, x*ab[1] + ab[2])
```

Rstudio에서 실행시켜 보세요.

이해를 돕기 위해서... 이거는 DL

2. 머신러닝과 딥러닝

```
import tensorflow as tf

x_data = [32.0,64.0,96.0,118.0,126.0,144.0,152.5,158.0] # x_data = [1., 2., 3.]
y_data = [18.0,24.0,61.5, 49.0, 52.0,105.0,130.3,125.0] # y_data = [1., 2., 3.]

# try to find values for w and b that compute y_data = W * x_data + b
W = tf.Variable(tf.random_normal([1], -10.0, 10.0)) # -1 ~ 1
b = tf.Variable(tf.random_normal([1], -100.0, 100.0)) # -1 ~ 1

# my hypothesis
hypothesis = W * x_data + b

# Simplified cost function
cost = tf.reduce_mean(tf.square(hypothesis - y_data))

# minimize
rate = tf.Variable(0.00001) # learning rate, alpha #0.1
optimizer = tf.train.GradientDescentOptimizer(rate)
train = optimizer.minimize(cost)

# before starting, initialize the variables. We will 'run' this first.
init = tf.global_variables_initializer()

# launch the graph
sess = tf.Session()
sess.run(init)

# fit the line
for step in range(2000001):
    sess.run(train)
    if step % 20000 == 0:
        print('{:4} {} {} {}'.format(step, sess.run(cost), sess.run(W), sess.run(b)))

sess.close() # learns best fit is W: [ 0.87189066], b: [-26.40464592]
```

DL로 회귀식을 계산하는데 얼마나 걸릴까?

2. 머신러닝과 딥러닝

```
jupyter Untitled Last Checkpoint: 10 minutes ago (unsaved changes)
File Edit View Insert Cell Kernel Help
+ [Icons] Code [CellToolbar]

# Launch the graph
sess = tf.Session()
sess.run(init)

print(dt.datetime.now())

# fit the line
for step in range(2000001):
    sess.run(train)
    if step % 20000 == 0:
        print('{:4} {} {}'.format(step, sess.run(cost),

# Learns best fit is W: [1] b: [0]

print(dt.datetime.now())
sess.close()

2017-10-25 07:44:31.422824
0 1694690.0 [-9.62430191] [-83.59861755]
20000 715.831237793 [ 1.29997289] [-80.77002716]
40000 682.072998047 [ 1.27954757] [-78.17603302]
60000 651.518554688 [ 1.26012802] [-75.70978546]
80000 623.755371094 [ 1.24159122] [-73.35568237]
100000 598.446289062 [ 1.2238363] [-71.10083771]
120000 575.627075195 [ 1.2070154] [-68.96460724]
140000 555.109619141 [ 1.19112754] [-66.94691467]
160000 536.234558105 [ 1.17577195] [-64.99679565]
180000 519.367370605 [ 1.16135406] [-63.16574097]
200000 503.999053955 [ 1.14755404] [-61.41320038]
220000 490.087463379 [ 1.13443041] [-59.74651337]
240000 477.473815918 [ 1.12192917] [-58.15888214]
260000 466.038269043 [ 1.11002111] [-56.64662552]
280000 455.667114258 [ 1.09867489] [-55.20566559]
300000 446.260681152 [ 1.08786142] [-53.83237457]
320000 437.798828125 [ 1.07764173] [-52.53450012]
340000 430.126647949 [ 1.06790841] [-51.29837799]
360000 423.154541016 [ 1.05861616] [-50.11832428]
380000 416.798248291 [ 1.04971504] [-48.98788834]
```

```
jupyter Untitled Last Checkpoint: 10 minutes ago (autosaved)
File Edit View Insert Cell Kernel Help
+ [Icons] Code [CellTool]

1400000 356.218231201 [ 0.88672084] [-28.28803062]
1500000 356.190795898 [ 0.88612008] [-28.21173668]
1520000 356.164703369 [ 0.88551933] [-28.13544273]
1540000 356.140106201 [ 0.88491857] [-28.05914879]
1560000 356.116943359 [ 0.88431782] [-27.98285484]
1580000 356.101104736 [ 0.88388431] [-27.92779732]
1600000 356.090545654 [ 0.8835839] [-27.88965034]
1620000 356.080413818 [ 0.88328356] [-27.85150337]
1640000 356.070587158 [ 0.88298315] [-27.8133564]
1660000 356.061126709 [ 0.8826828] [-27.77520943]
1680000 356.052124023 [ 0.88238239] [-27.73706245]
1700000 356.043212891 [ 0.88208205] [-27.69891548]
1720000 356.034912109 [ 0.8817817] [-27.66076851]
1740000 356.026855469 [ 0.88148129] [-27.62262154]
1760000 356.019256592 [ 0.88118094] [-27.58447456]
1780000 356.011962891 [ 0.88088053] [-27.54632759]
1800000 356.004974365 [ 0.88058019] [-27.50818062]
1820000 355.998535156 [ 0.88027978] [-27.47003365]
1840000 355.9921875 [ 0.87997943] [-27.43188667]
1860000 355.986328125 [ 0.87967908] [-27.3937397]
1880000 355.980804443 [ 0.87937868] [-27.35559273]
1900000 355.975646973 [ 0.87907833] [-27.31744576]
1920000 355.970977783 [ 0.87877792] [-27.27929878]
1940000 355.966491699 [ 0.87847757] [-27.24115181]
1960000 355.962432861 [ 0.87817717] [-27.20300484]
1980000 355.959777832 [ 0.87797189] [-27.17692947]
2000000 355.959777832 [ 0.87797189] [-27.17692947]
2017-10-25 07:50:55.128509
```

In []: 07:50:55 – 07:44:31 = 6분 14초
어디에서?

사용한 GPU

2. 머신러닝과 딥러닝

```
root@1554758ff6b4:~# nvidia-smi
Thu Jan  5 06:45:34 2017
```

M10

NVIDIA-SMI 367.55					Driver Version: 367.55				
GPU	Name	Persistence-M	Bus-Id	Disp.A	Volatile	Uncorr.	ECC		
Fan	Temp	Perf	Pwr:Usage/Cap	Memory-Usage	GPU-Util	Compute	M.		
0	Tesla M10	Off	0000:06:00.0	Off			N/A		
N/A	28C	P8	8W / 53W	0MiB / 8127MiB	0%		Default		
Processes:									
GPU	PID	Type	Process name	GPU Memory Usage					
No running processes found									




SPECIFICATIONS

Virtualization Use Case	Density-Optimized Graphics Virtualization
GPU Architecture	NVIDIA Maxwell™
GPUs per Board	4
Max User per Board	64 (16 per GPU)
NVIDIA CUDA® Cores	2560 NVIDIA CUDA Cores (640 per GPU)
GPU Memory	32 GB of GDDR5 Memory (8 per GPU)
H.264 1080p30 Streams	28
Max Power Consumption	225 W
Thermal Solution	Passive
Form Factor	PCIe 3.0 Dual Slot

참고 : <http://images.nvidia.com/content/tesla/pdf/188359-Tesla-M10-DS-NV-Aug19-A4-fn1-Web.pdf>

2. 머신러닝과 딥러닝

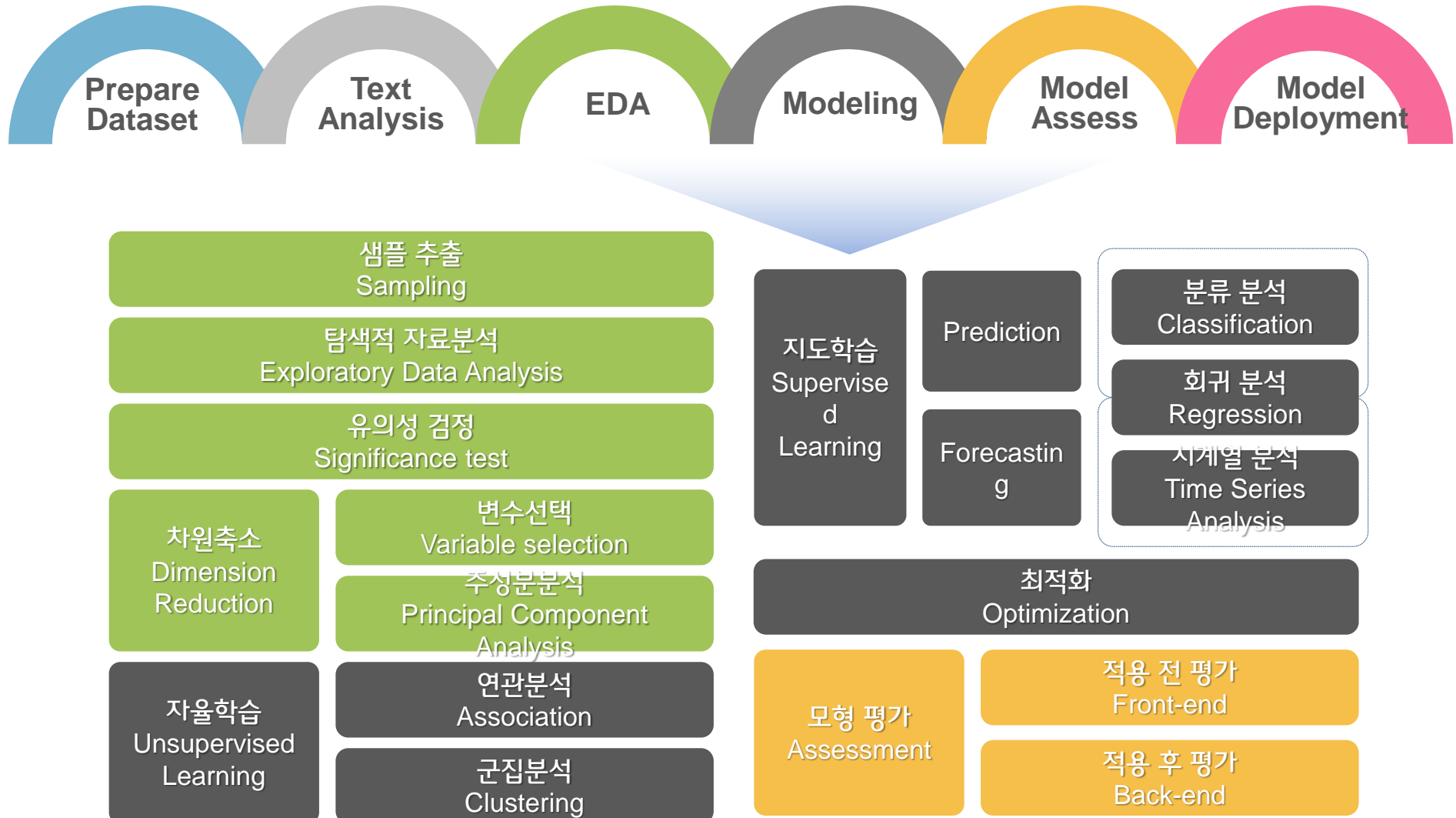
- | | | | |
|---|---|---|---|
| 5 | 0 | 4 | 1 |
|---|---|---|---|

- 

[illegible]

데이터 분석 단계에서 머신러닝

2. 머신러닝과 딥러닝



머신러닝 단계

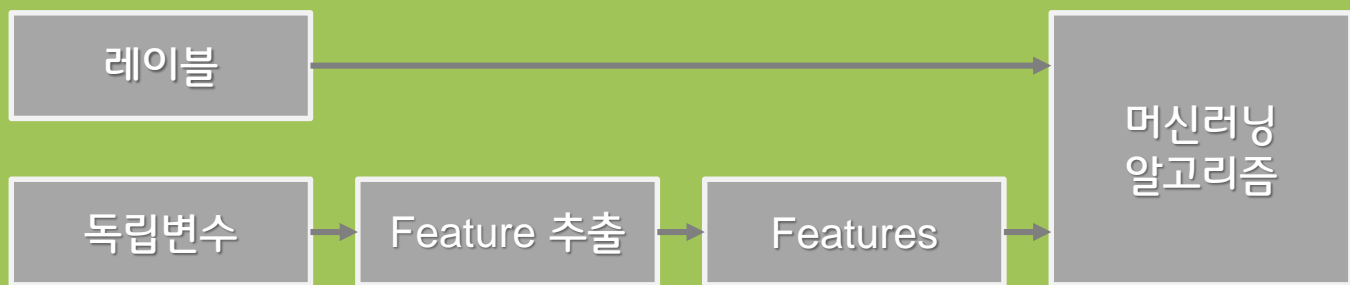
2. 머신러닝과 딥러닝

학습(training) vs. 추론(Prediction/Inference)

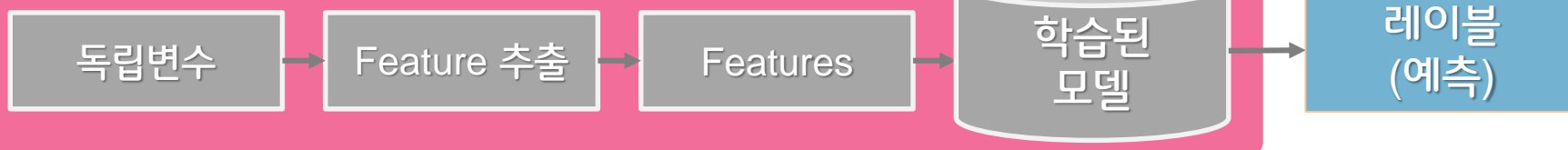
학습: 훈련 데이터를 이용하여 모델을 학습하는 과정

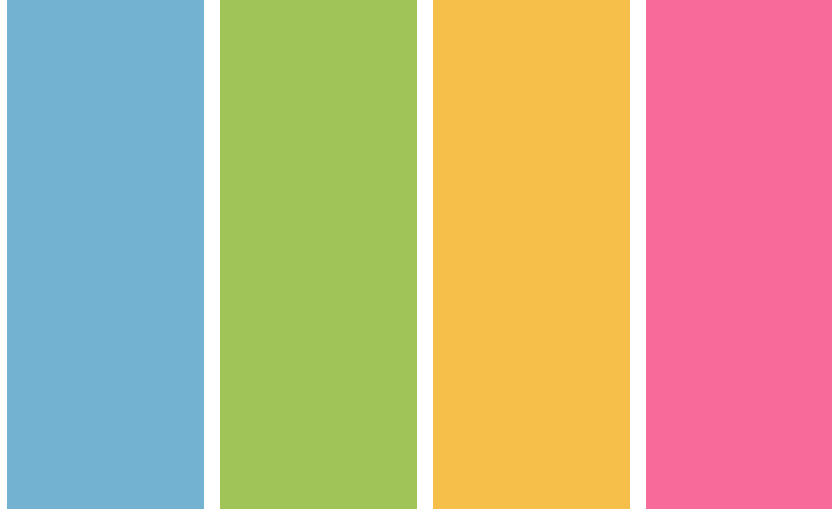
추론: 학습된 모델을 이용하여 미래의 새로운 데이터를 추론/예측하는 과정

훈련 단계



추론 단계



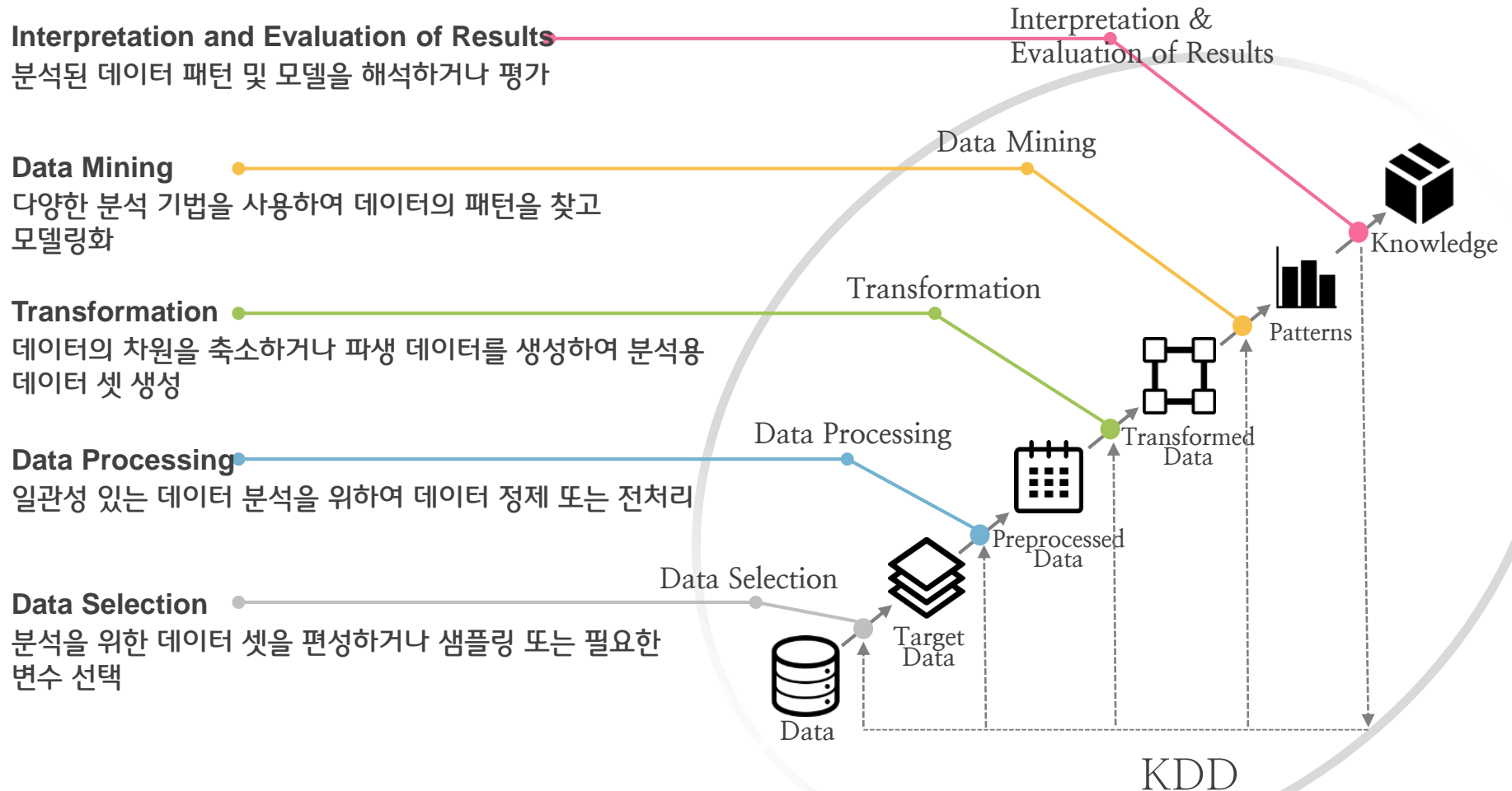


3. 머신러닝 프로젝트

KDD 방법론

3. 머신러닝 프로젝트

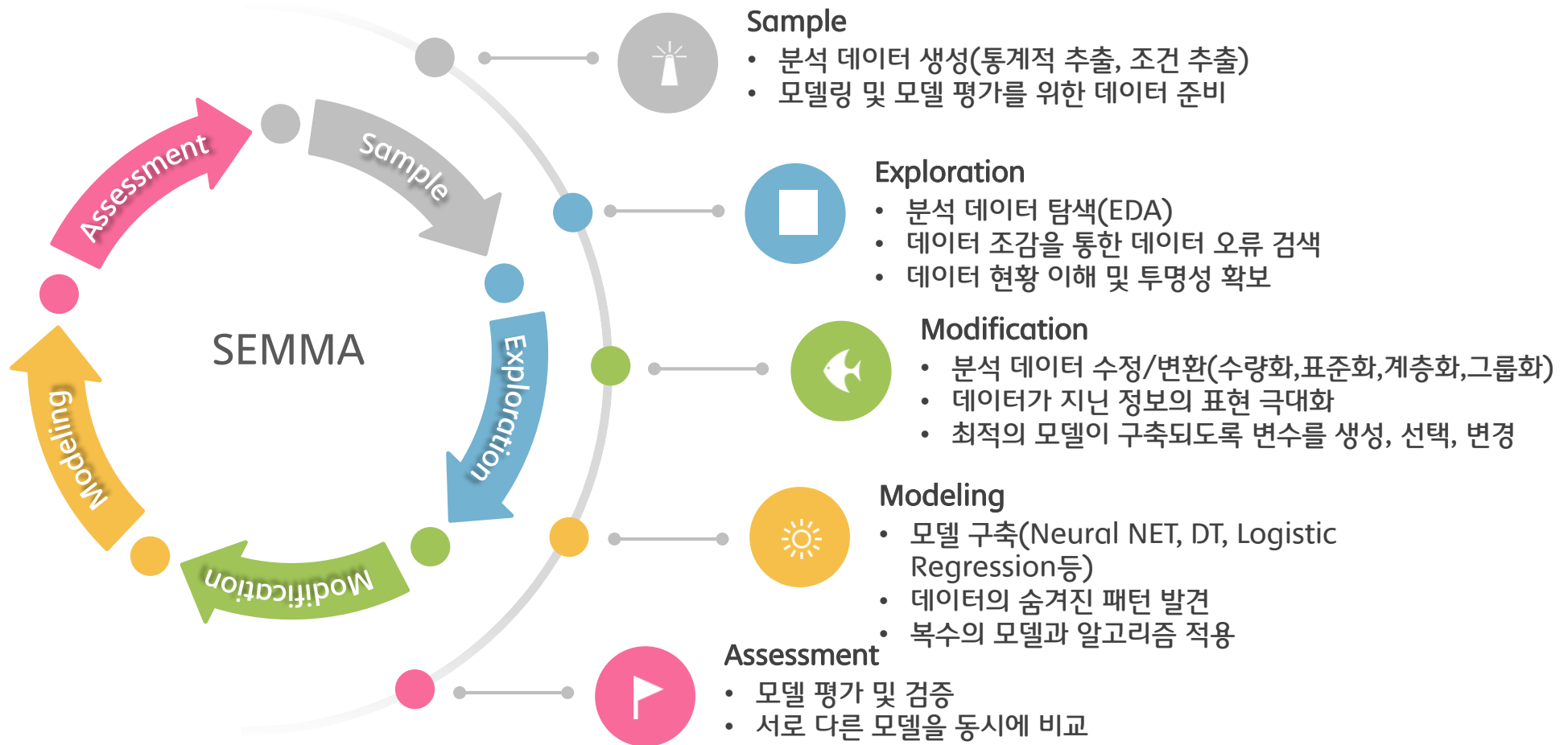
Knowledge Discovery and Data Mining



SEMMA 방법론

3. 머신러닝 프로젝트

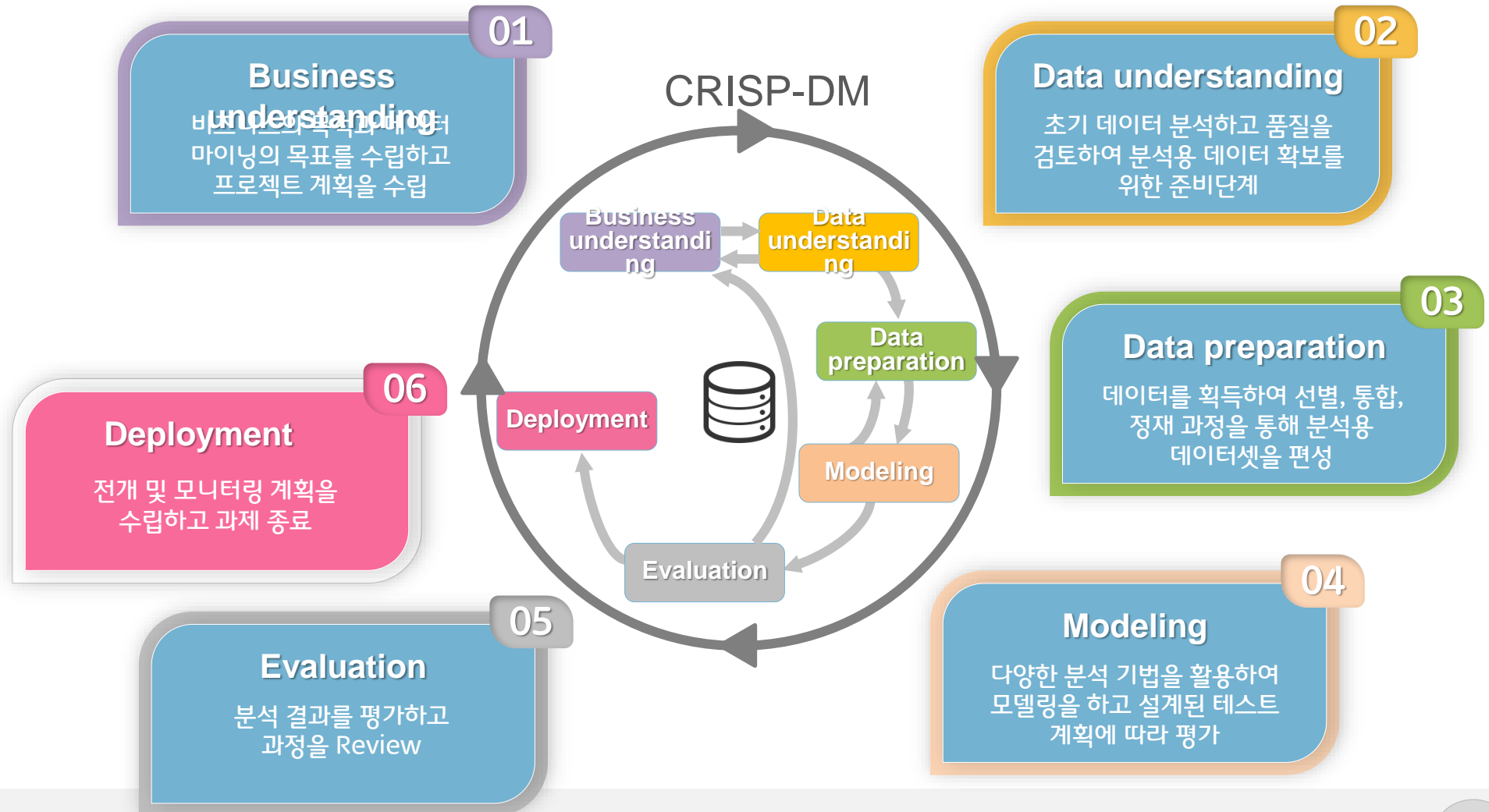
Sample, Exploration, Modification, Modeling, Assessment



CRISP-DM 방법론

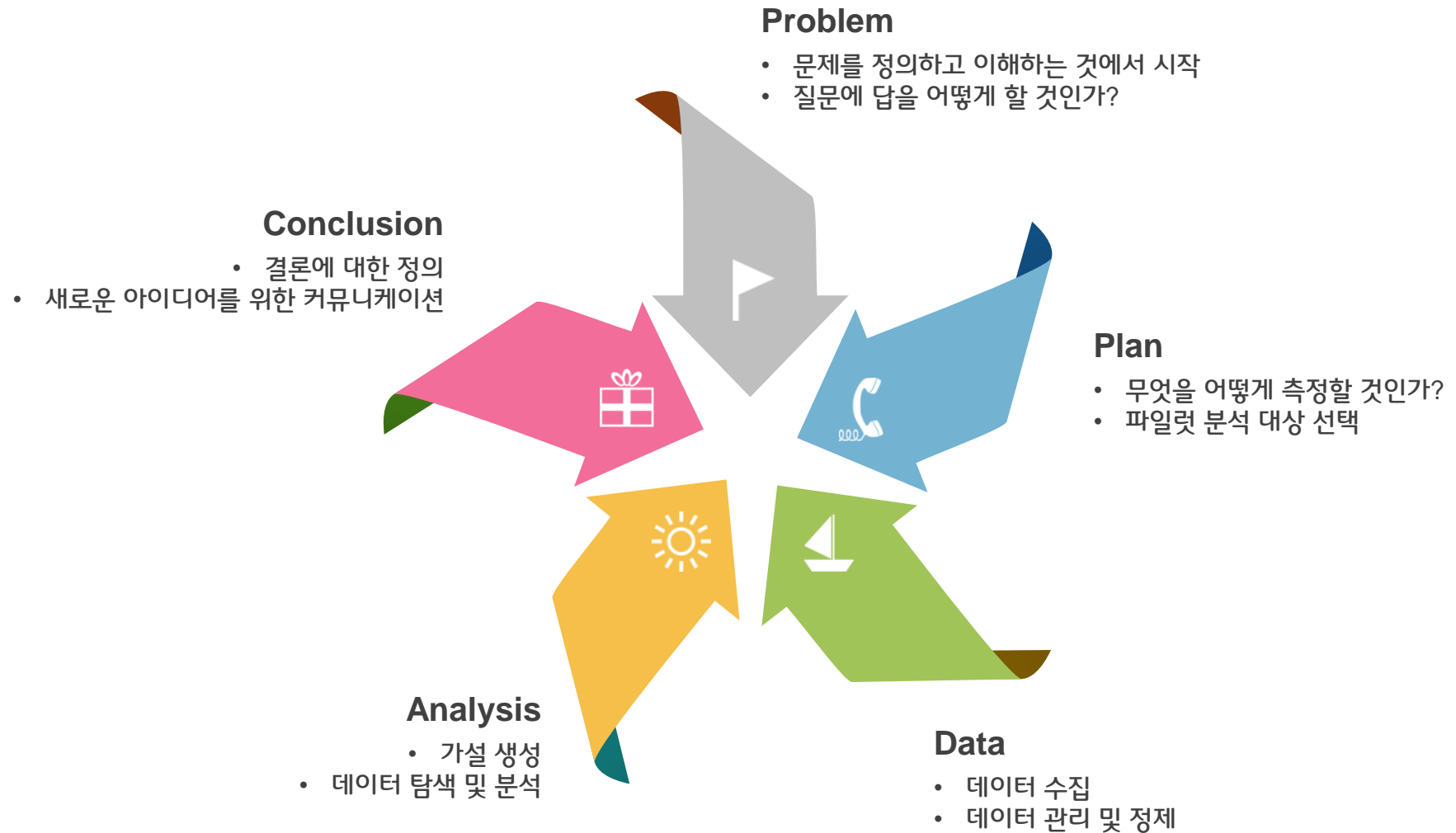
3. 머신러닝 프로젝트

Cross Industry Standard Process for Data Mining



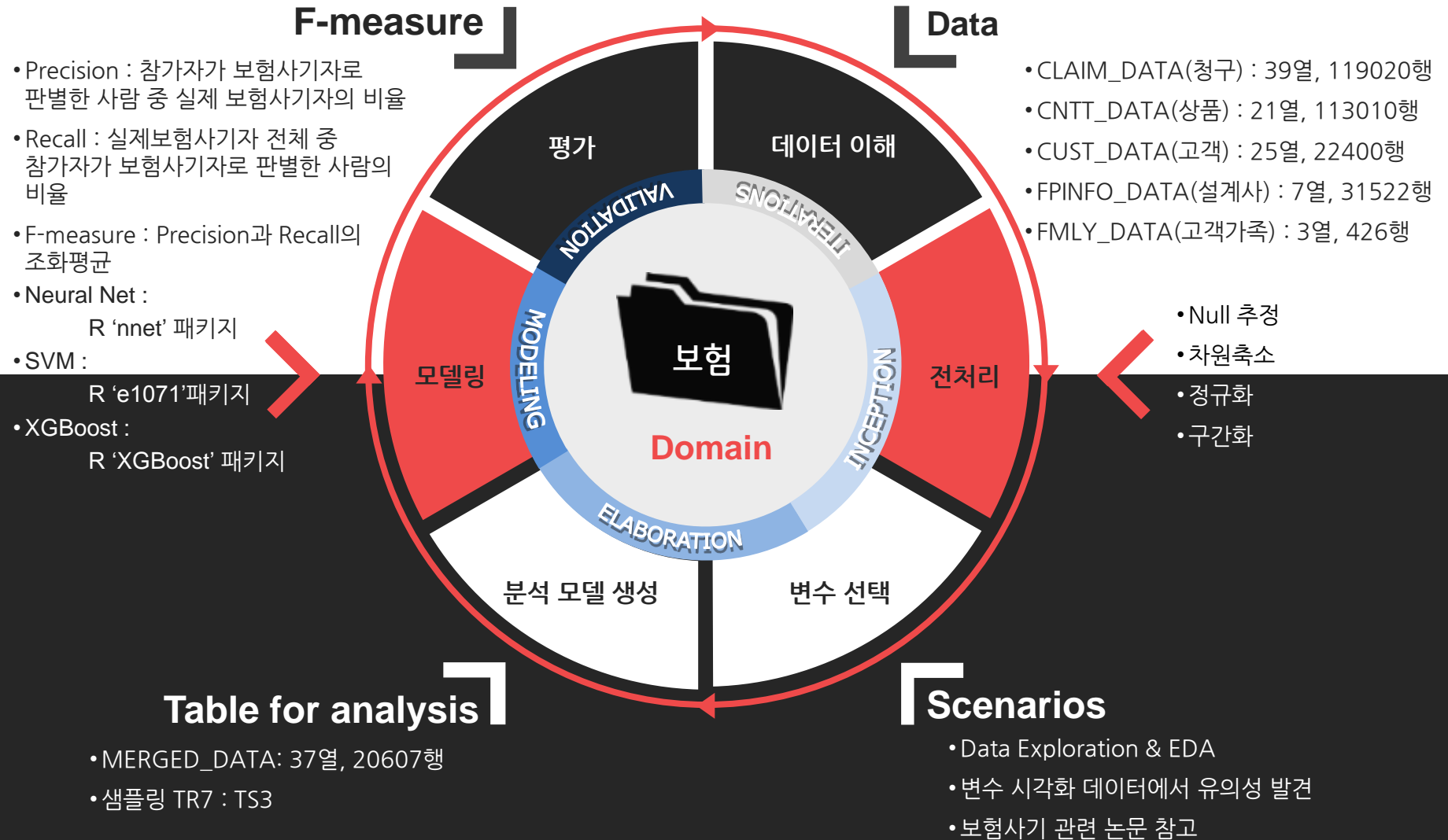
PPDAC 방법론

3. 머신러닝 프로젝트



Data Analysis Flow (예시)

3. 머신러닝 프로젝트

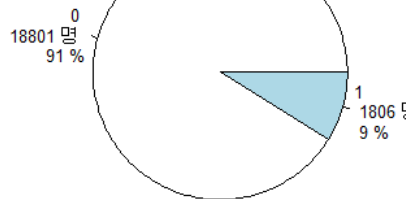


Data Exploration - Visualization (예시)

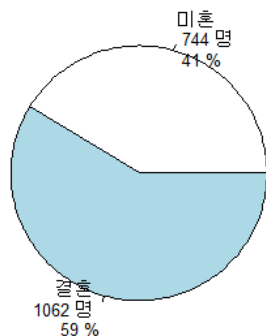
3. 머신러닝 프로젝트

시간 시각화, 분포 시각화, 관계 시각화, 비교 시각화, 공간 시각화

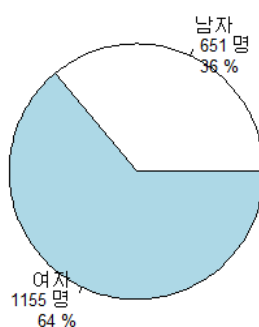
사기자 수



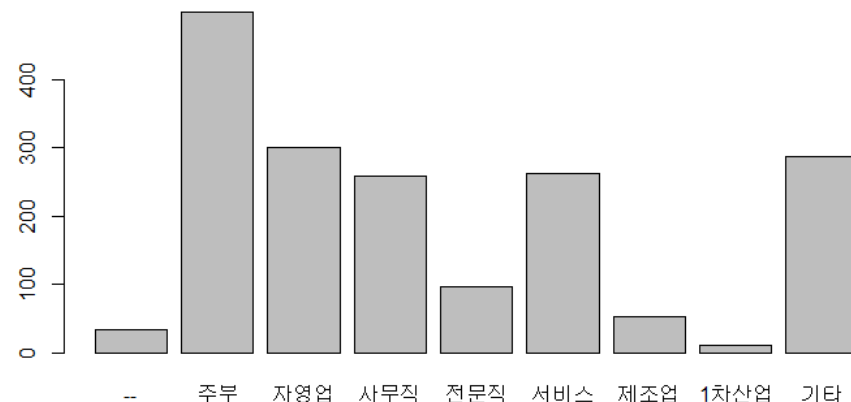
결혼 여부별 사기자 수



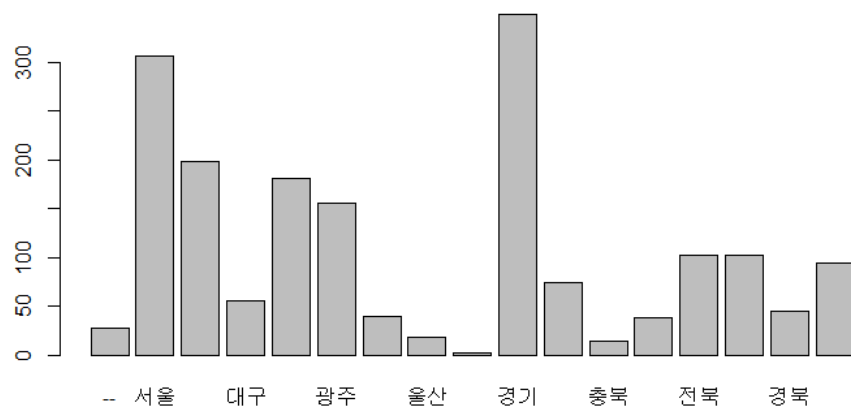
성별별 사기자 수



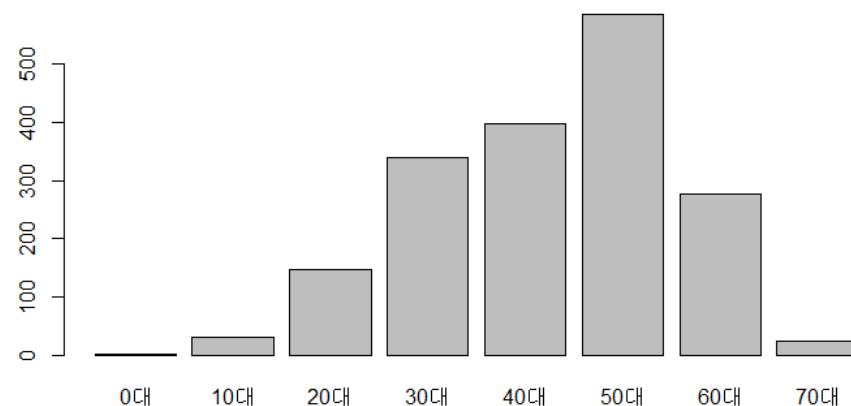
직업별 사기자 수



거주지별 사기자 수



연령대별 사기자 수



Preprocessing (예시)

3. 머신러닝 프로젝트

결측치 추정

Null 비율 20% 미만

연속형

이상치 제외 평균
ex) 추정가구소득

범주형

Null 값을 미리 정의된
값으로 대체
ex) 미혼 고객 배후자의
직업, 막내 나이 등 은 0



차원 축소

Null 비율 20% 이상

Column 삭제

Null값을 추정하는 데이터의
양이 많아져 모델의 정확도가
저하되므로 모델에서
제외시킴

ex) 납입 총 보험료, 추정
개인 소득, 최대 보험료,
신용등급 등

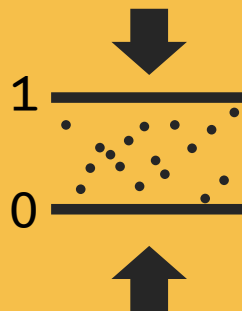


정규화

최대값 및 최소값 정의

0~1 정규화

연속형 데이터의 경우 모든
데이터를 차원간 Scale이
다르면 모델링 과정이
제대로 수행이 안되기
때문에 정규화 시킴



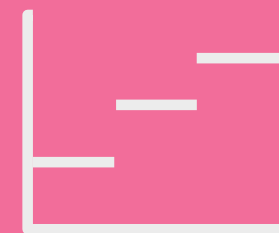
구간화

연속형 ▶ 범주형

의미부여

연속형으로 산재되어 있는
데이터를 일정 규칙에
맞추어 범주형으로 변형
함으로써 데이터에 의미
부여

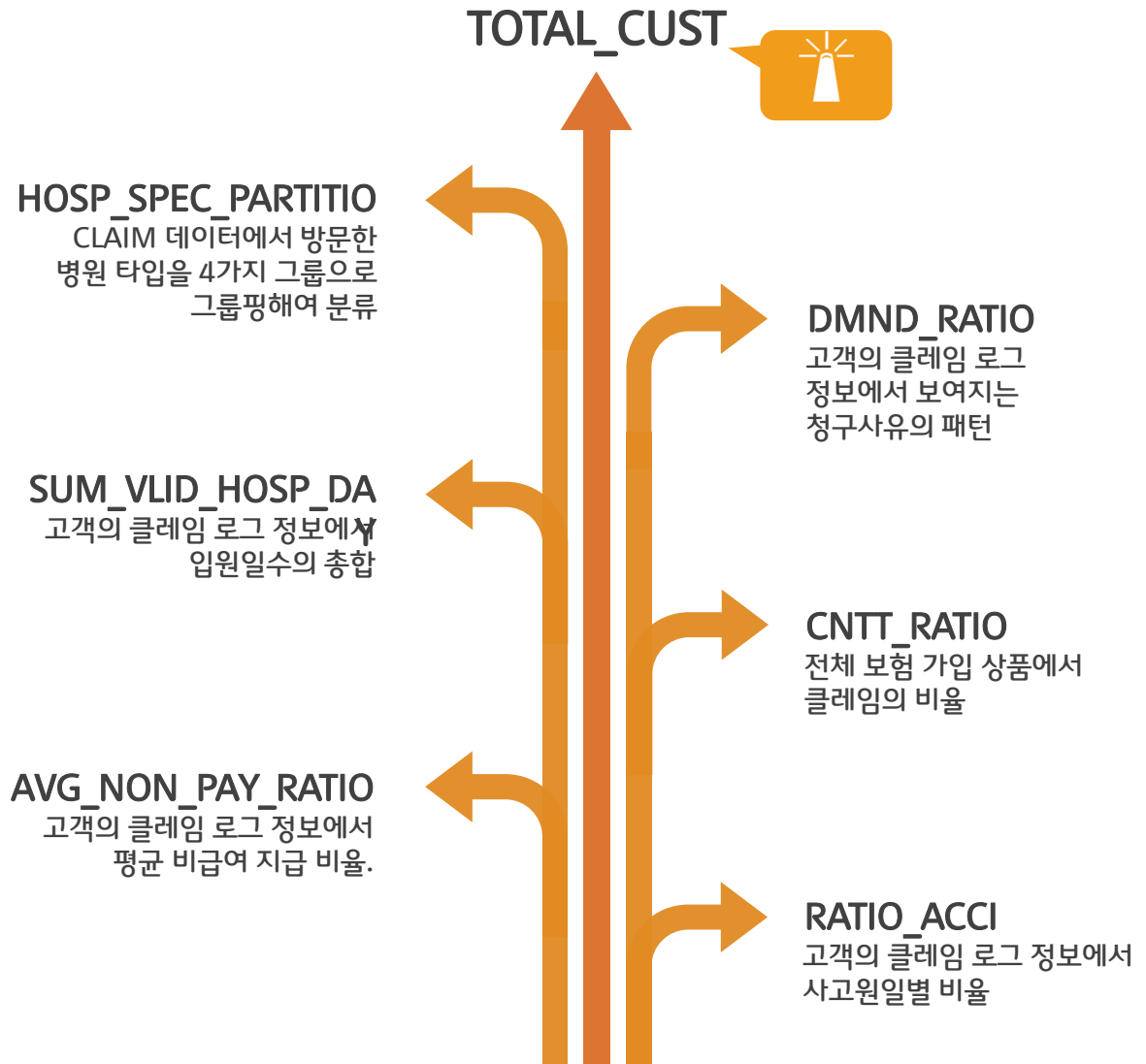
ex) 나이 ▶ 세대

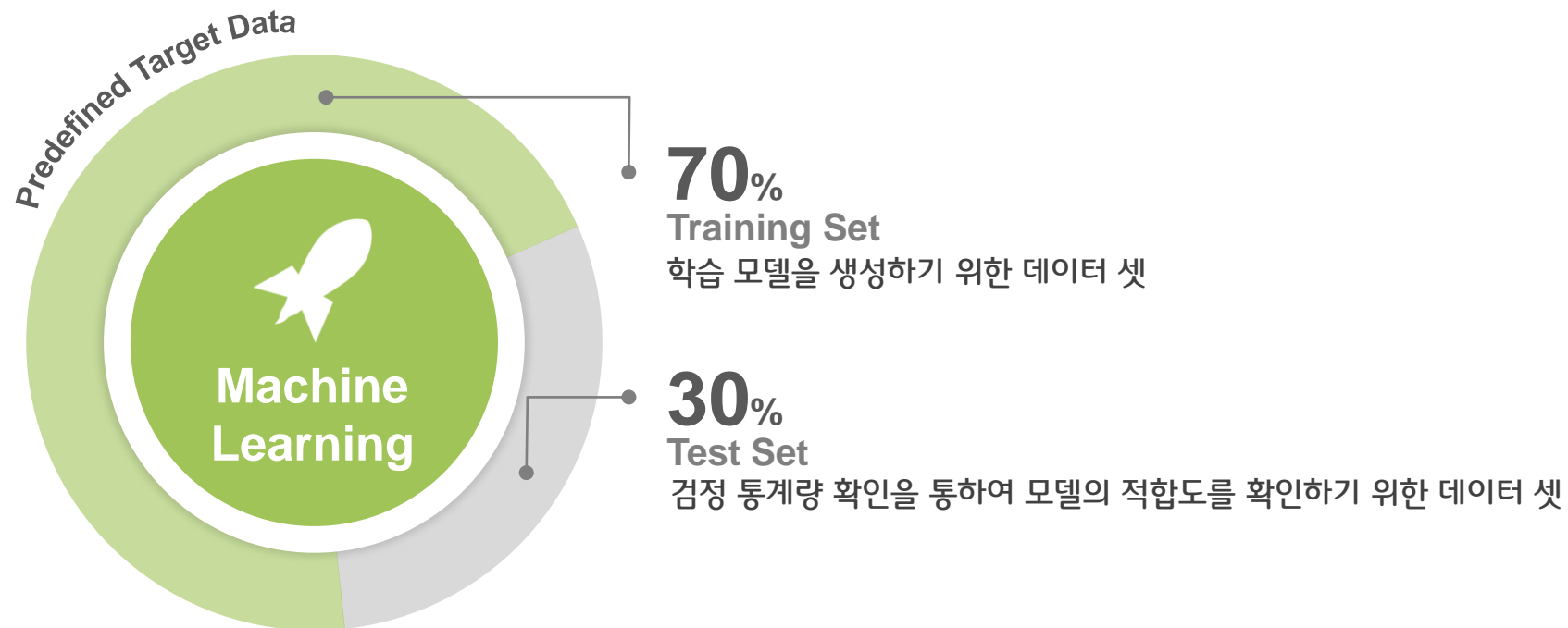


가설에 따른 파생변수 추가 (예시)

3. 머신러닝 프로젝트

원래 데이터가
가지고 있지 않은
변수를 파악하고
추가하는 것은
데이터 분석에
있어서 매우 중요





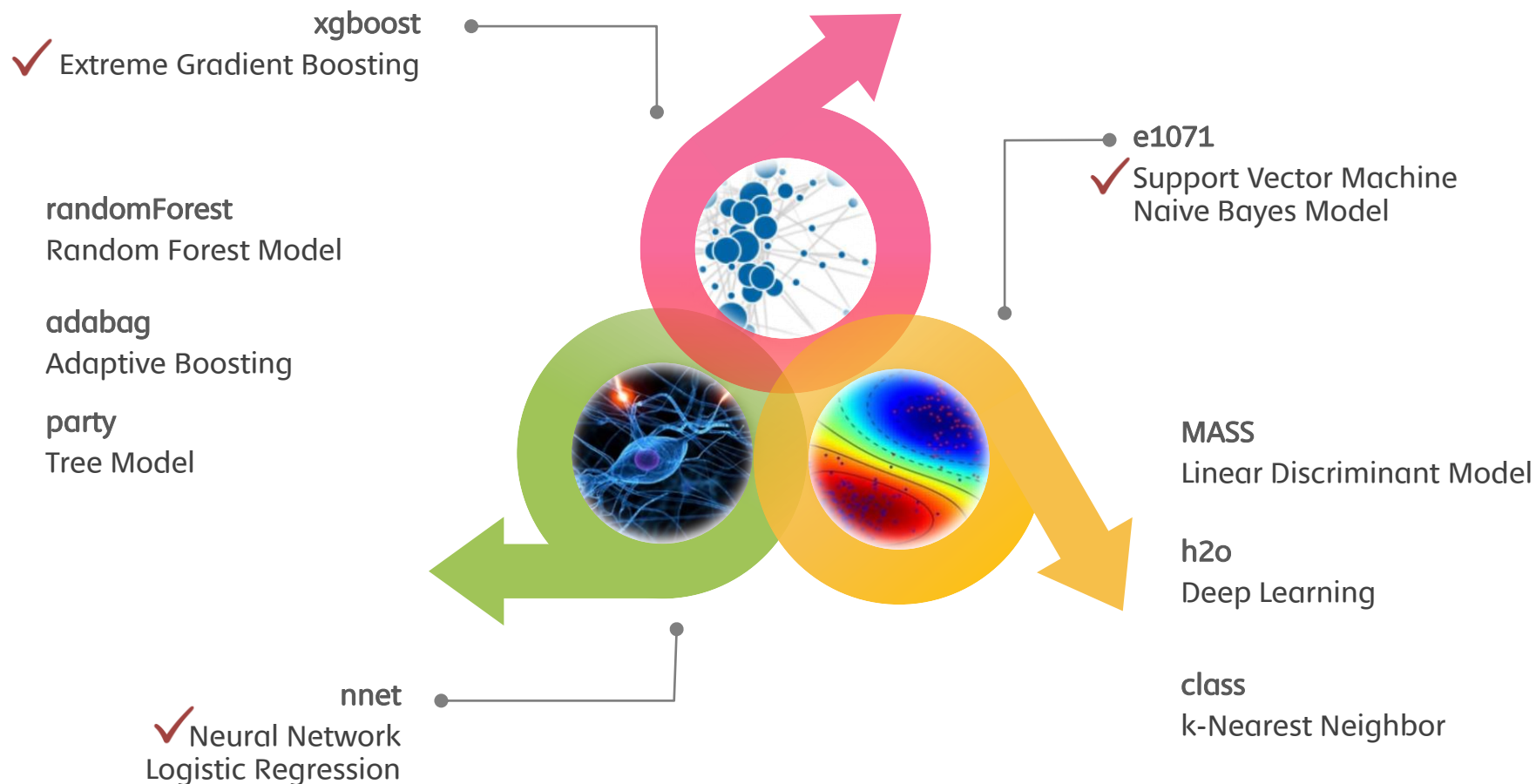
Training Set & Test Set

이미 분류되어 있는 데이터 셋에 대하여 트레이닝 셋과 테스트 셋으로 나누고 트레이닝 셋을 이용하여 학습 모델을 생성한다. 테스트셋과 예측 데이터를 비교한 검정 통계량 확인을 통하여 모델의 적합도를 확인한다.

ML(Classification) Model

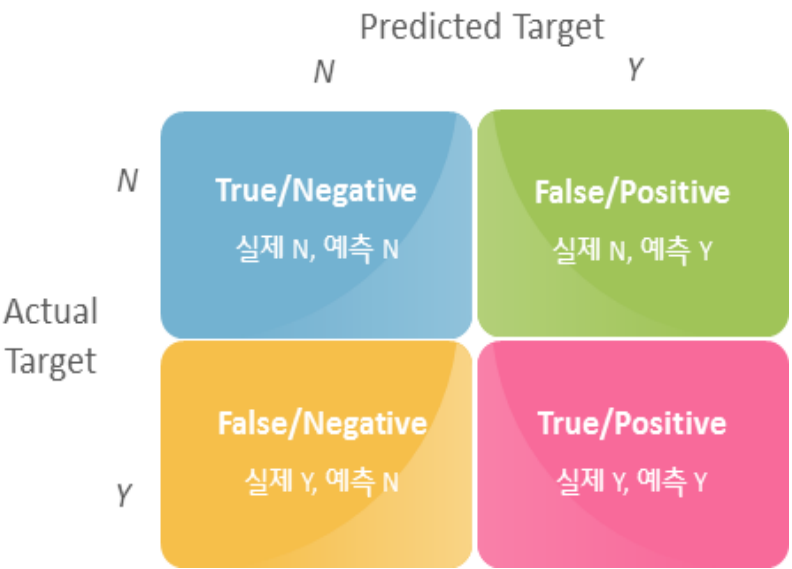
3. 머신러닝 프로젝트

모델링을 위해 어떤 패키지와 어떤 모델 알고리즘을 사용 할 것인가?



분류 모델의 평가

3. 머신러닝 프로젝트



메트릭	계산식	의미
정밀도 (Precision)	$TP/(FP+TP)$	Y으로 예측된 것 중 실제로도 Y인 경우의 비율
민감도 (Recall)	$TP/(FN+TP)$	실제로 Y인 것들 중 예측이 Y로 된 경우 비율 (=Sensitivity)
정확도 (Accuracy)	$(TP+TN)/(TP+FP+FN+TN)$	전체 예측에서 옳은 예측의 비율
특이도 (Specificity)	$TN/(FP+TN)$	실제로 N인 것들 중에서 예측이 N으로 된 경우의 비율
오류율 (FP Rate)	$FP/(FP+TN)$	Y가 아닌데 Y로 예측된 비율.(=errorrate) 1-Specificity와 같은 값
F-measure	$2 * \frac{Precision * Recall}{Precision + Recall}$	Precision과 Recall의 조화 평균. 시스템의 성능을 하나의 수치로 표현하기 위해 사용하는 점수. 0~1 사이의 값을 가짐. Precision과 Recall 두 값이 골고루 클 때 큰 값을 가짐.
Kappa	$K = \frac{P_{accuracy} - P(e)}{1 - P(e)}$	코헨의카파. 두 평가자의 평가가 얼마나 일치하는지 평가하는 값. 0~1사이의 값을 가짐. P(e)는 두 평가자의 평가가 우연히 일치할 확률. 코헨의 카파는 두 평가자의 평가가 우연히 일치할 확률을 제외한 뒤의 점수.