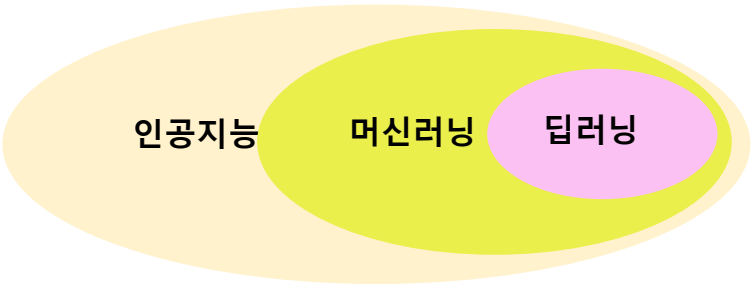
| | 개요 | 머신러닝 개요 |
|---|---|---|




머신 러닝이란 수많은 데이터를 학습시켜 거기에 있는 패턴을 찾아내는 것.
패턴을 찾으면 그러한 패턴을 기반으로 데이터를 분류하거나 미래를 예측.

(풀고자 하는 문제)

**1. 분류 (Classification)**

예) 손글씨 인식, 스팸메일 분류, 증권 사기, 자동차의 길이, 너비, 높이, 바퀴크기, 엔진 마력 등의 특징을 보고 경차, 준중형차, 대형차 중 한가지로 분류.

**2. 군집 (Clustering)**

비슷한 특징을 가지는 데이터 인스턴스들끼리 그룹화.
예) 마케팅을 위해 고객을 군집화. 사용자의 취향을 그룹으로 묶어 사용자 취향에 맞는 광고 제공

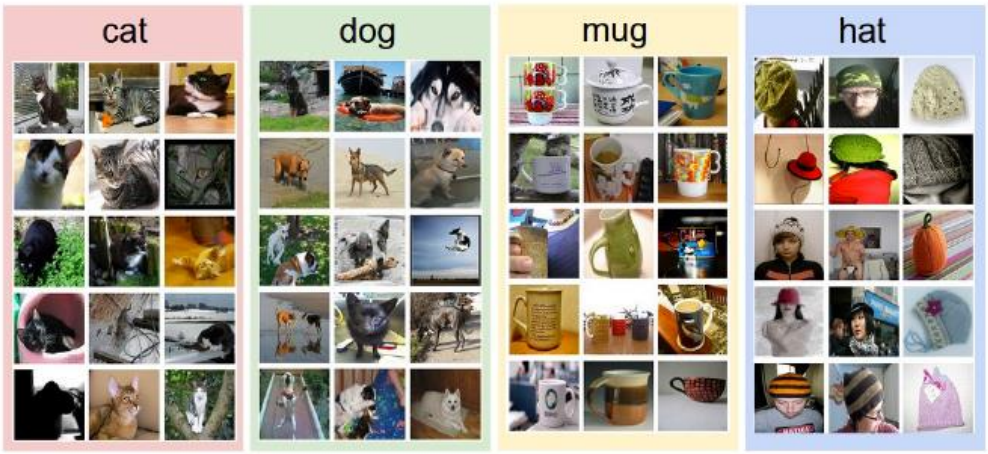
**3. 회귀 (Regression)**

과거 데이터를 기반으로 미래를 예측.
예) 판매 예측, 주가 변동 예측
불완전한 데이터의 값을 알아내기 위한 문제.
예) 자동차의 너비, 높이, 바퀴의 크기, 엔진 마력 등의 특징을 하는 데 길이를 모른다고 했을 때, 다른 값들을 근거로 불완전한 데이터로 구성된 데이터 인스턴스의 길이 특징을 예측.

# [ 머신 러닝 종류/학습 방법 ]

## 1. 지도 학습 Supervised Learning

데이터와 함께 답을 입력. Training Data Set 필요. learning with labeled examples
다른 데이터의 답을 예측,
예) - Email spam filter: learning from labeled (spam or ham) email
- Image labeling: learning from tagged images




## 2. 비지도 학습 Unsupervised learning

데이터만 입력, 답은 입력하지 않음. 데이터의 규칙을 찾아냄.
Unlabeled Data
최종적으로 내야 하는 답이 정해져 있지 않고, 사람도 알 수 없는 본질적인 구조 확인.
예) Google news grouping, Clustering analysis

## 3. 강화 학습

현재 상태를 관찰해서 어떻게 대응해야 할지와 관련된 문제를 다룸.
행동의 주체가 환경을 관찰하고, 이를 기반으로 의사결정을 내리며 어떤 보상을 받음.
행동의 주체는 더 많은 보상을 얻을 수 있는 방법으로 행동을 학습. 예) 주식 투자 시뮬레이션

**(머신 러닝 흐름)**
**데이터 수집,전처리→ 데이터 저장→ 데이터 학습(학습 방법선택, 매개변수조정, 학습반복)→ 평가/검증**

# [ 지도 학습 ]

예) Predicting final exam score based on time spent

**regression**

예) 학습 데이터가 다음과 같을 때 7시간 공부하면 몇 점을 받을까 ?

| x (hours) | y (score) |
|---|---|
| 10 | 90 |
| 9 | 80 |
| 3 | 50 |
| 2 | 30 |

**binary classification**

Pass/non-pass based on time spent

예) 학습 데이터가 다음과 같을 때 7시간 공부하면 Pass or non-pass ?

| x (hours) | y (pass/fail) |
|---|---|
| 10 | P |
| 9 | P |
| 3 | F |
| 2 | F |

**multi-label classification**

Letter grade (A, B, C, D and F) based on time spent

예) 학습 데이터가 다음과 같을 때 7시간 공부하면 학점은? A or B or C or D or F ?

| x (hours) | y (grade) |
|---|---|
| 10 | A |
| 9 | B |
| 3 | D |
| 2 | F |

| Chapter 1 | machine Learning with Scikit-Learn | binary classification |
|---|---|---|

scikit-learn 은 분류, 회귀, 클러스터링, 차원 축소 등 머신 러닝에서 자주 사용되는 다양한 알고리즘 지원.
지도학습으로 binary classification을 실습해 보겠습니다.

**[ binary classification ]**

**실습**

ex01_xor.py **(학습함수 fit, 예측 함수 predict)**

```
from sklearn import svm

clf = svm.SVC()
clf.fit([
    [0,0],
    [1,0],
    [0,1],
    [1,1]
], [0,1,1,0])
results = clf.predict([
        [0,0],
        [1,0]
])
print(results)
```

(출력 결과)

[0 1]

(핵심 함수)

```
clf = svm.SVC()          # SVC 알고리즘 선택
clf.fit(data, label)     #학습
pre = clf.predict(data)  #예측
```

**#정답률 구하는 함수 metrics.accuracy_score**

ex02_xor.py (간단한 함수로 정답률 구하기. 정답률로 가설의 신뢰성 확인)

```
import pandas as pd
from sklearn import svm, metrics

# XOR 연산
xor_input = [
    [0, 0, 0],
    [0, 1, 1],
    [1, 0, 1],
    [1, 1, 0]
]

# 입력을 학습 전용 데이터와 테스트 전용 데이터로 분류
xor_df = pd.DataFrame(xor_input)
xor_data  = xor_df.loc[:,0:1] # 데이터
xor_label = xor_df.loc[:,2]   # 레이블

# 데이터 학습과 예측
clf = svm.SVC()
clf.fit(xor_data, xor_label)
pre = clf.predict(xor_data)

# 정답률 구하기
ac_score = metrics.accuracy_score(xor_label, pre)
print("정답률 =", ac_score)
```

(출력 결과)

정답률 = 1.0

**(핵심 함수)**

```
ac_score = metrics.accuracy_score(xor_label, pre)    #정답률 구하는 함수
```

| Chapter 1 | machine Learning with Scikit-Learn | multi-label classification |
|---|---|---|

**[ multi-label classification ]**

붓꽃의 품종 분류하기(지도 학습으로 multi-label classification을 실습해 보겠습니다. )

붓꽃은 150 종류 이상의 품종이 있어서 일반적인 사람은 물론이고 꽃을 조금 아는 사람도 품종을 분류하기 굉장히 어려움. 러닝 머신을 이용하여 꽃잎과 꽃받침의 크기를 기반으로 분류해 보겠습니다

"iris.csv" 파일(공유폴더에 주어짐)은 다음과 같은 형태의 데이터입니다.

| SepalLength | SepalWidth | PetalLength | PetalWidth | Name |
|---|---|---|---|---|
| 5.1 | 3.5 | 1.4 | 0.2 | Iris-setosa |
| 4.9 | 3.0 | 1.4 | 0.2 | Iris-setosa |
| 4.7 | 3.2 | 1.3 | 0.2 | Iris-setosa |
| 4.6 | 3.1 | 1.5 | 0.2 | Iris-setosa |
| 5.0 | 3.6 | 1.4 | 0.2 | Iris-setosa |
| 5.4 | 3.9 | 1.7 | 0.4 | Iris-setosa |
| 4.6 | 3.4 | 1.4 | 0.3 | Iris-setosa |
| 5.0 | 3.4 | 1.5 | 0.2 | Iris-setosa |
| 4.4 | 2.9 | 1.4 | 0.2 | Iris-setosa |
| 4.9 | 3.1 | 1.5 | 0.1 | Iris-setosa |
| 5.4 | 3.7 | 1.5 | 0.2 | Iris-setosa |
| 4.8 | 3.4 | 1.6 | 0.2 | Iris-setosa |
| 4.8 | 3.0 | 1.4 | 0.1 | Iris-setosa |

ex03_iris-train.py

```
import pandas as pd
from sklearn import svm, metrics
from sklearn.model_selection import train_test_split

# 붓꽃의 CSV 데이터 읽어 들이기
csv = pd.read_csv('iris.csv')

# from sklearn.datasets import load_iris
# data=load_iris()
# print(type(data))

# 필요한 열 추출
csv_data = csv[["SepalLength","SepalWidth","PetalLength","PetalWidth"]]
csv_label = csv["Name"]

# 학습 전용 데이터와 테스트 전용 데이터로 나누기

train_data, test_data, train_label, test_label = \
train_test_split(csv_data, csv_label)

# 데이터 학습시키고 예측
clf = svm.SVC()
clf.fit(train_data, train_label)
pre = clf.predict(test_data)

# 정답률 구하기 --- (※5)
ac_score = metrics.accuracy_score(test_label, pre)
print("정답률 =", ac_score)
```

(출력 결과)

정답률 = 1.0

**(핵심 함수)**

```
clf = svm.SVC()       # 머신 러닝 SVC 알고리즘을 선택.
clf.fit(train_data, train_label)    # 학습
pre = clf.predict(test_data)       # 예측
ac_score = metrics.accuracy_score(test_label, pre)     # 정답률 구하기
```

참고) # 학습 전용 데이터와 테스트 전용 데이터로 나누어주는 함수

train_data, test_data, train_label, test_label = train_test_split(csv_data, csv_label)

정리

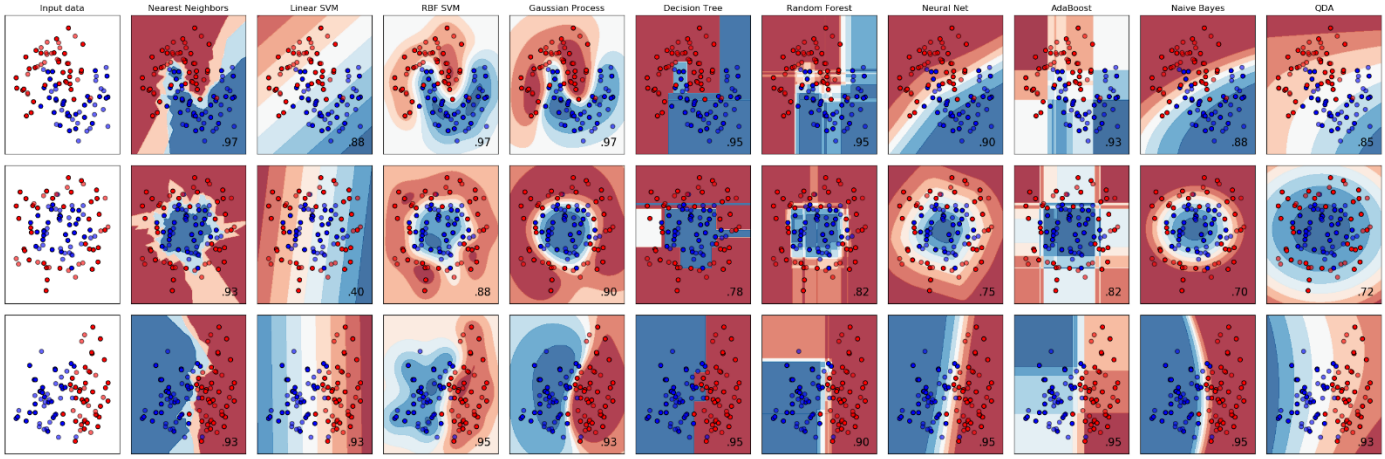
머신 러닝으로 데이터 분류를 실습해 보았습니다.

지도 학습을 할 때는 데이터와 정답 레이블을 지정합니다.

훈련 전용 데이터와 테스트 전용 데이터로 분할하여, 정답률을 확인합니다.

# [ Classifier comparison ]

http://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html



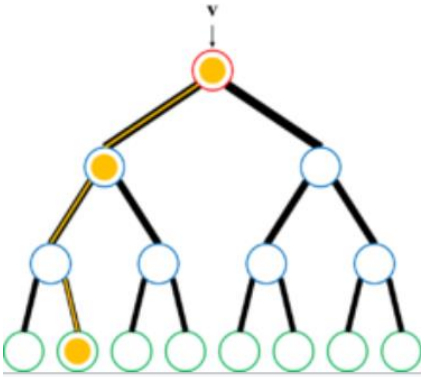

input data의 분포에 따라 적당한 classifier를 선택한다.

**렌덤 포레스트 사용하기**

**(핵심 함수)**

```
clf = RandomForestClassifier()   # 랜덤포레스트 학습하기
clf.fit(data_train, label_train)      # 학습
clf.predict(data_test)                # 예측
```

기계 학습에서의 랜덤 포레스트는 분류, 회귀 분석 등에 사용되는 앙상블 학습 방법의 일종으로, 훈련 과정에서 구성한 다수의 결정 트리로부터 부류(분류) 또는 평균 예측치(회귀 분석)를 출력함으로써 동작한다. 랜덤 포레스트는 여러 개의 결정 트리들을 임의적으로 학습하는 방식의 앙상블 방법이다



결정 트리 테스트 단계

랜덤 포레스트의 장점은

1. 딥러닝과 비교했을 때 사람이 직관적으로 이해하기 쉽습니다.
2. 성능이 우수합니다. 딥러닝이 성능이 우수하다고 하나 많은 데이터가 있는 경우이고, 데이터가 적은 경우에는 오히려 랜덤 포레스트가 더 좋은 성능을 발휘하기도 합니다. 클래스가 많은 경우에 사용시 성능이 좋지 않습니다.
3. classification과 regression이 모두 가능합니다.

실습 버섯 분류하기

mushroom.csv 파일

8124종류의 버섯들을 22개의 특성과 함께 독이 있는지 기록.

첫 열의 p는 폭이 있는, e는 독이 없다는 의미. 두번째 열부터의 설명은 아래 설명

```
p,x,s,n,t,p,f,c,n,k,e,e,s,s,w,w,p,w,o,p,k,s,u
e,x,s,y,t,a,f,c,b,k,e,c,s,s,w,w,p,w,o,p,n,n,g
e,b,s,w,t,l,f,c,b,n,e,c,s,s,w,w,p,w,o,p,n,n,m
p,x,y,w,t,p,f,c,n,n,e,e,s,s,w,w,p,w,o,p,k,s,u
e,x,s,g,f,n,f,w,b,k,t,e,s,s,w,w,p,w,o,e,n,a,g
e,x,y,y,t,a,f,c,b,n,e,c,s,s,w,w,p,w,o,p,k,n,g
e,b,s,w,t,a,f,c,b,g,e,c,s,s,w,w,p,w,o,p,k,n,m
e,b,y,w,t,l,f,c,b,n,e,c,s,s,w,w,p,w,o,p,n,s,m
p,x,y,w,t,p,f,c,n,p,e,e,s,s,w,w,p,w,o,p,k,v,g
e,b,s,y,t,a,f,c,b,g,e,c,s,s,w,w,p,w,o,p,k,s,m
e,x,y,y,t,l,f,c,b,g,e,c,s,s,w,w,p,w,o,p,n,n,g
e,x,y,y,t,a,f,c,b,n,e,c,s,s,w,w,p,w,o,p,k,s,m
e,b,s,y,t,a,f,c,b,w,e,c,s,s,w,w,p,w,o,p,n,s,g
p,x,y,w,t,p,f,c,n,k,e,e,s,s,w,w,p,w,o,p,n,v,u
e,x,f,n,f,n,f,w,b,n,t,e,s,f,w,w,p,w,o,e,k,a,g
e,s,f,g,f,n,f,c,n,k,e,e,s,s,w,w,p,w,o,p,n,y,u
e,f,f,w,f,n,f,w,b,k,t,e,s,s,w,w,p,w,o,e,n,a,g
p,x,s,n,t,p,f,c,n,n,e,e,s,s,w,w,p,w,o,p,k,s,g
p,x,y,w,t,p,f,c,n,n,e,e,s,s,w,w,p,w,o,p,n,s,u
p,x,s,n,t,p,f,c,n,k,e,e,s,s,w,w,p,w,o,p,n,s,u
e,b,s,y,t,a,f,c,b,k,e,c,s,s,w,w,p,w,o,p,n,s,m
p,x,y,n,t,p,f,c,n,n,e,e,s,s,w,w,p,w,o,p,n,v,g
e,b,y,y,t,l,f,c,b,k,e,c,s,s,w,w,p,w,o,p,n,s,m
```

1. cap-shape: bell=b,conical=c,convex=x,flat=f, knobbed=k,sunken=s
2. cap-surface: fibrous=f,grooves=g,scaly=y,smooth=s
3. cap-color: brown=n,buff=b,cinnamon=c,gray=g,green=r, pink=p,purple=u,red=e,white=w,yellow=y
4. bruises?: bruises=t,no=f
5. odor: almond=a,anise=l,creosote=c,fishy=y,foul=f, musty=m,none=n,pungent=p,spicy=s
6. gill-attachment: attached=a,descending=d,free=f,notched=n
7. gill-spacing: close=c,crowded=w,distant=d
8. gill-size: broad=b,narrow=n
9. gill-color: black=k,brown=n,buff=b,chocolate=h,gray=g, green=r,orange=o,pink=p,purple=u,red=e, white=w,yellow=y
10. stalk-shape: enlarging=e,tapering=t
11. stalk-root: bulbous=b,club=c,cup=u,equal=e, rhizomorphs=z,rooted=r,missing=?
12. stalk-surface-above-ring: fibrous=f,scaly=y,silky=k,smooth=s
13. stalk-surface-below-ring: fibrous=f,scaly=y,silky=k,smooth=s
14. stalk-color-above-ring: brown=n,buff=b,cinnamon=c,gray=g,orange=o, pink=p,red=e,white=w,yellow=y
15. stalk-color-below-ring: brown=n,buff=b,cinnamon=c,gray=g,orange=o, pink=p,red=e,white=w,yellow=y
16. veil-type: partial=p,universal=u
17. veil-color: brown=n,orange=o,white=w,yellow=y
18. ring-number: none=n,one=o,two=t
19. ring-type: cobwebby=c,evanescent=e,flaring=f,large=l, none=n,pendant=p,sheathing=s,zone=z
20. spore-print-color: black=k,brown=n,buff=b,chocolate=h,green=r, orange=o,purple=u,white=w,yellow=y
21. population: abundant=a,clustered=c,numerous=n, scattered=s,several=v,solitary=y
22. habitat: grasses=g,leaves=l,meadows=m,paths=p, urban=u,waste=w,woods=d

ex04_mushroom-train.py

```python
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics
from sklearn.model_selection import train_test_split

mr = pd.read_csv("mushroom.csv", header=None)

label = []
data = []
attr_list = []

# data와 label로 나누고
# data 내부의 기호를 숫자로 변환하기 (fit함수의 학습 데이터는 숫자이거나 숫자로 형변환이 가능해야함)

for row_index, row in mr.iterrows(): #DataFrame객체 의 iterrows() 메소드를 for문과 함께 사용하면
                                     # 행인덱스와, 행데이터를  한행 씩 반환
    label.append(row.ix[0]) #0번 컬럼에 독이 있는지 없는지 정보를 label리스트에 담는다.
    row_data = []
    for v in row.ix[1:]:
        row_data.append(ord(v)) #Return the Unicode code point for a one-character string.

    data.append(row_data)
```

```
# 학습 전용과 테스트 전용 데이터로 나누기
data_train, data_test, label_train, label_test = \
    train_test_split(data, label)

# 데이터 학습
clf = RandomForestClassifier()
clf.fit(data_train, label_train)

# 데이터 예측
predict = clf.predict(data_test)

# 결과 테스트
ac_score = metrics.accuracy_score(label_test, predict)
cl_report = metrics.classification_report(label_test, predict)

print("정답률 =", ac_score)
print("리포트 =\n", cl_report)
```

(출력 결과)

정답률 = 1.0

리포트 =

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| e | 1.00 | 1.00 | 1.00 | 1081 |
| p | 1.00 | 1.00 | 1.00 | 950 |
| avg / total | 1.00 | 1.00 | 1.00 | 2031 |

**데이터를 숫자로 변경할 때 주의 사항**

다음과 같이 데이터를 숫자로 할당했을 때 문제점

red=1, blue=2, yellow=3

scikit-learn에서는 연속형 자료로 인식하여 yellow는 blue나 red보다 높다 로 받아들임. 그 결과, 퍼포먼스는 형편없이 떨어지게 됨

아래 같이, 변경(**One hot encording**)후, 학습에 활용하면 연속형 자료로 인식하는 문제는 해결

red = 1 0 0

blue = 0 1 0

yellow= 0 0 1

데이터의 용량은 늘어나지만 색끼리 전혀 관련성이 없다면 위와 같이 다루는 것이 좋음.

**[ One hot encording 방법 ]**

ex05_one_hot_encoding.py

```
from sklearn.preprocessing import OneHotEncoder

def main():
    enc = OneHotEncoder()
    sourcedata = [
        [65],
        [66],
        [67]
    ]
    enc.fit(sourcedata)

    data =[
        [65],
        [66],
        [67],
        [65],
        [67]
    ]
    print(enc.transform(data).toarray())

if __name__ == '__main__':
    main()
```

1) OneHotEncoder객체의 fit 함수는
65, 66, 67에 각각의 원핫인코딩 값 생성하고 기억.

2) OneHotEncoder객체의 transform함수는
매개인자(data)의 각 값에 대하여 1)에서 생성한 값으로 번역 .

실행결과 :

```
[[1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]
 [1. 0. 0.]
 [0. 0. 1.]]
```