공공 데이터를 활용한 머신러닝 실습

- 데이터 셋 : 버섯과 관련된 데이터
  - https://archive.ics.uci.edu/ml/datasets.php (https://archive.ics.uci.edu/ml/datasets.php)
- 머신러닝 사용모델 : 랜덤 포레스트

1-1 머신러닝 모델 만들기

- 대표적 모델 중의 하나인 랜덤 포레스트 모델 사용
- 2001년에 레오 브라이만(Leo Breiman)이 제안한 머신러닝 알고리즘.
- 다수의 결정 트리를 만들고, 만들어진 의사결정트리를 기반으로 결과를 유도하므로 높은 정밀도를 자랑.

1-2 데이터 준비

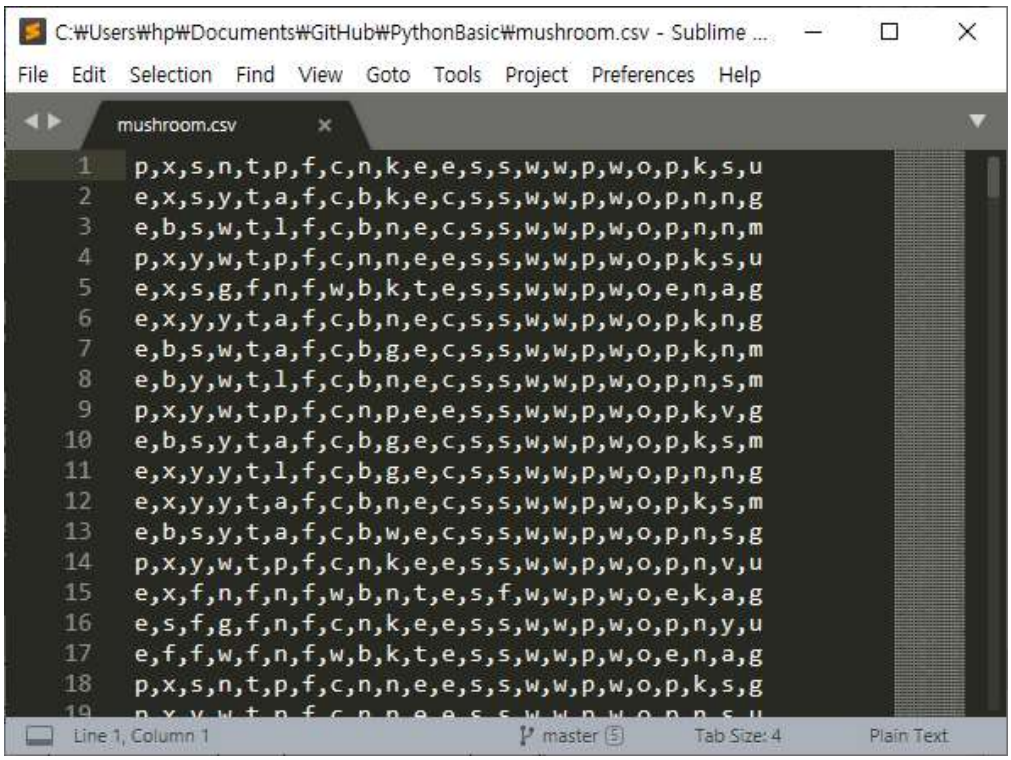
- mushroom :
  - 8,124 종류의 버섯과 특징과 독이 있는지가 적혀 있는 데이터 세트.
  - 버섯의 특징을 기반으로 독의 유무를 판정하기 위한 것.
- link : https://archive.ics.uci.edu/ml/machine-learning-databases/mushroom/agaricus-lepiota.data (https://archive.ics.uci.edu/ml/machine-learning-databases/mushroom/agaricus-lepiota.data)

- 데이터 다운로드

In [24]:

```
import urllib.request as req
local = "mushroom.csv"
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/mushroom/agaricus-lepiota.data"
req.urlretrieve(url, local)
print("다운로드 완료")
```

```
다운로드 완료
```



- 한줄 한줄이 버섯 한 종류를 나타냄.
- 쉼표로 구분된 열은 23개, 23개의 특징이 알파벳으로 기록이 되어 있음.
- 가장 왼쪽열은 독의 유무 독이 있으면 p(poisonous), 식용이면 e(edible)
- 두 번째 열은 버섯의 머리 모양 b(벨형태), c(원뿔 형태), x(볼록한 형태), f(평평한 형태), k(혹 형태), s(오목한 형태)
- 네 번째 열은 버섯의 머리 색. n(갈색), b(황갈색) 등
  - 설명은 UCI 사이트에서 확인 가능

In [25]:

```
### 모델 만들기
import pandas as pd
from sklearn import metrics
from sklearn.model_selection import train_test_split
```

In [26]:

```
# 데이터 읽기
mush = pd.read_csv("mushroom.csv", header=None)
mush
```

Out[26]:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
0	p	x	s	n	t	p	f	c	n	k	e	e	s	s	w	w	p	w	o	p	k	s	u
1	e	x	s	y	t	a	f	c	b	k	e	c	s	s	w	w	p	w	o	p	n	n	g
2	e	b	s	w	t	l	f	c	b	n	e	c	s	s	w	w	p	w	o	p	n	n	m
3	p	x	y	w	t	p	f	c	n	n	e	e	s	s	w	w	p	w	o	p	k	s	u
4	e	x	s	g	f	n	f	w	b	k	t	e	s	s	w	w	p	w	o	e	n	a	g
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	..
8119	e	k	s	n	f	n	a	c	b	y	e	?	s	s	o	o	p	o	o	p	b	c	
8120	e	x	s	n	f	n	a	c	b	y	e	?	s	s	o	o	p	n	o	p	b	v	
8121	e	f	s	n	f	n	a	c	b	n	e	?	s	s	o	o	p	o	o	p	b	c	
8122	p	k	y	n	f	y	f	c	n	b	t	?	s	k	w	w	p	w	o	e	w	v	
8123	e	x	s	n	f	n	a	c	b	y	e	?	s	s	o	o	p	o	o	p	o	c	

8124 rows × 23 columns

기호를 숫자로 변경

In [27]:

```
from sklearn import preprocessing

encoder_le = preprocessing.LabelEncoder()
mush['label'] = encoder_le.fit_transform(mush.iloc[:, 0])  # 1열의 값을 변경하여 label을 만든다.
mush
```

Out[27]:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
0	p	x	s	n	t	p	f	c	n	k	e	e	s	s	w	w	p	w	o	p	k	s	u
1	e	x	s	y	t	a	f	c	b	k	e	c	s	s	w	w	p	w	o	p	n	n	g
2	e	b	s	w	t	l	f	c	b	n	e	c	s	s	w	w	p	w	o	p	n	n	m
3	p	x	y	w	t	p	f	c	n	n	e	e	s	s	w	w	p	w	o	p	k	s	u
4	e	x	s	g	f	n	f	w	b	k	t	e	s	s	w	w	p	w	o	e	n	a	g
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
8119	e	k	s	n	f	n	a	c	b	y	e	?	s	s	o	o	p	o	o	p	b	c	
8120	e	x	s	n	f	n	a	c	b	y	e	?	s	s	o	o	p	n	o	p	b	v	
8121	e	f	s	n	f	n	a	c	b	n	e	?	s	s	o	o	p	o	o	p	b	c	
8122	p	k	y	n	f	y	f	c	n	b	t	?	s	k	w	w	p	w	o	e	w	v	
8123	e	x	s	n	f	n	a	c	b	y	e	?	s	s	o	o	p	o	o	p	o	c	

8124 rows × 24 columns

2열부터 나머지 열을 숫자로 변경

In [28]:

```
for i in range(1,23,1):
    mush['col' + str(i)] = encoder_le.fit_transform(mush.iloc[:, i])  # 각 열의 값을 변경하여 featur

mush
```

Out[28]:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
0	p	x	s	n	t	p	f	c	n	k	e	e	s	s	w	w	p	w	o	p	k	s	u
1	e	x	s	y	t	a	f	c	b	k	e	c	s	s	w	w	p	w	o	p	n	n	g
2	e	b	s	w	t	l	f	c	b	n	e	c	s	s	w	w	p	w	o	p	n	n	m
3	p	x	y	w	t	p	f	c	n	n	e	e	s	s	w	w	p	w	o	p	k	s	u
4	e	x	s	g	f	n	f	w	b	k	t	e	s	s	w	w	p	w	o	e	n	a	g
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	..
8119	e	k	s	n	f	n	a	c	b	y	e	?	s	s	o	o	p	o	o	p	b	c	
8120	e	x	s	n	f	n	a	c	b	y	e	?	s	s	o	o	p	n	o	p	b	v	
8121	e	f	s	n	f	n	a	c	b	n	e	?	s	s	o	o	p	o	o	p	b	c	
8122	p	k	y	n	f	y	f	c	n	b	t	?	s	k	w	w	p	w	o	e	w	v	
8123	e	x	s	n	f	n	a	c	b	y	e	?	s	s	o	o	p	o	o	p	o	c	

8124 rows × 46 columns

In [29]:

```
col_all = list(range(0,23,1))
col_all
```

Out[29]:

```
[0,
 1,
 2,
 3,
 4,
 5,
 6,
 7,
 8,
 9,
 10,
 11,
 12,
 13,
 14,
 15,
 16,
 17,
 18,
 19,
 20,
 21,
 22]
```

기존의 정보 열을 삭제

In [30]:

```
mush.drop(col_all, axis=1)
```

Out[30]:

	label	col1	col2	col3	col4	col5	col6	col7	col8	col9	col10	col11	col12	col13	c
0	1	5	2	4	1	6	1	0	1	4	0	3	2	2	
1	0	5	2	9	1	0	1	0	0	4	0	2	2	2	
2	0	0	2	8	1	3	1	0	0	5	0	2	2	2	
3	1	5	3	8	1	6	1	0	1	5	0	3	2	2	
4	0	5	2	3	0	5	1	1	0	4	1	3	2	2	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
8119	0	3	2	4	0	5	0	0	0	11	0	0	2	2	
8120	0	5	2	4	0	5	0	0	0	11	0	0	2	2	
8121	0	2	2	4	0	5	0	0	0	5	0	0	2	2	
8122	1	3	3	4	0	8	1	0	1	0	1	0	2	1	
8123	0	5	2	4	0	5	0	0	0	11	0	0	2	2	

8124 rows × 23 columns

In [31]:

```
X = mush.loc[ , "col1":"col7"]  # 모든 행, col1~col7까지 선택
y = mush['label']               # 예측하고자 하는 열 선택

print(X.shape, y.shape)
```

```
(8124, 7) (8124,)
```

모델을 위해 사용하는 데이터

- 6093행 학습시키는 데이터 행의 수
- 2031행 만들어진 모델로 확인해 보는 데이터 수

In [32]:

```
### 학습 데이터와 테스트 데이터 나누기
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0)

print(X_train.shape, y_train.shape)
print(X_test.shape, y_test.shape)
```

```
(6093, 7) (6093,)
(2031, 7) (2031,)
```

1-3 모델 선택 및 학습

In [33]:

```
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
```

In [34]:

```
### 모델 선택 및 학습
model = RandomForestClassifier()
model.fit(X_train, y_train)
```

Out[34]:

```
RandomForestClassifier()
```

In [35]:

```
### 새로운 데이터로 예측해 보기
predict = model.predict(X_test)
predict
```

Out[35]:

```
array([1, 0, 0, ..., 1, 0, 0])
```

In [19]:

```
print( len(predict), len(y_test) )
```

```
2031 2031
```

In [20]:

```
y_test.values
```

Out[20]:

```
array([1, 0, 0, ..., 1, 0, 0])
```

1-4 정확도 확인

In [22]:

```
# 얼마나 적중했을까?
import numpy as np
np.mean( predict==y_test.values )
```

Out[22]:

```
0.9945839487936977
```

Summary

- 6100여개 데이터를 이용하여 예측할 수 있는 머신러닝 모델을 만들었다.
- 2031여개 데이터를 이용하여 예측을 수행하였다.
- 정확도는 99%의 정확도를 확인할 수 있다.

In [ ]: