# 원핫 인코딩 실습

## 학습 내용

- 기본 one-hot encoding 실습
- hello wordl 실습

## 목차



## 01. 기본 실습 - One-hot encoding

간단한 데이터를 준비하여, 목표 feature인 'target'를 라벨인코딩 (labelencoding) 후, 이 후, 결과값을 이용하여 one-hot-encoding를 수행한다.

In [1]:
```
### 01. 데이터 준비
import pandas as pd
data = { "feature1":[2,3,8,4],
         "feature2":[22,32,82,42],
         "target": ["b","c", "a", "d"]
       }
df = pd.DataFrame(data)
df
```

Out[1]:

| | feature1 | feature2 | target |
|---|---|---|---|
| **0** | 2 | 22 | b |
| **1** | 3 | 32 | c |
| **2** | 8 | 82 | a |
| **3** | 4 | 42 | d |

In [2]:
```
from sklearn import preprocessing
```

In [3]:
```
label_encoder = preprocessing.LabelEncoder()
df['lbl_en'] = label_encoder.fit_transform(df['target'])
df
```

Out[3]:

| | feature1 | feature2 | target | lbl_en |
|---|---|---|---|---|
| **0** | 2 | 22 | b | 1 |
| **1** | 3 | 32 | c | 2 |
| **2** | 8 | 82 | a | 0 |
| **3** | 4 | 42 | d | 3 |

In [4]:
```
print( len(df) )
print( df['lbl_en'].values.shape )
```

```
4
(4,)
```

In [5]:
```
train_y = df['lbl_en'].values.reshape(len(df), 1)
print(train_y.shape)
train_y
```

```
(4, 1)
```

Out[5]:
```
array([[1],
       [2],
       [0],
       [3]])
```

In [6]:
```
# 원핫 인코딩 수행
onehot_encoder = preprocessing.OneHotEncoder(sparse=False)
train_y_onehot = onehot_encoder.fit_transform(train_y)
print(train_y_onehot)
print(train_y_onehot.shape)
```

```
[[0. 1. 0. 0.]
 [0. 0. 1. 0.]
 [1. 0. 0. 0.]
 [0. 0. 0. 1.]]
(4, 4)
```

In [7]:
```
# 원래 df에 원핫 인코딩 한 내용을 열기준으로 붙이기
onehot_val = pd.DataFrame(train_y_onehot, dtype=int)
df_new = pd.concat([df, onehot_val], axis=1)
df_new
```

Out[7]:

| | feature1 | feature2 | target | lbl_en | 0 | 1 | 2 | 3 |
|---|---|---|---|---|---|---|---|---|
| **0** | 2 | 22 | b | 1 | 0 | 1 | 0 | 0 |
| **1** | 3 | 32 | c | 2 | 0 | 0 | 1 | 0 |
| **2** | 8 | 82 | a | 0 | 1 | 0 | 0 | 0 |
| **3** | 4 | 42 | d | 3 | 0 | 0 | 0 | 1 |

## 02. 원핫 인코딩 실습 - 사계절

목차로 이동하기

In [8]:
```
# 라이브러리 불러오기
import numpy as np
from numpy import argmax    # 가장 값이 큰 인덱스 반환
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder
```

In [9]:
```
data = ['spring', 'spring', 'summer', 'spring', 'autumn',
        'autumn', 'winter', 'spring', 'summer', 'autumn']
values = np.array(data)
print(values)

# 라벨 인코딩 수행 - 범주형 문자를 정수로 바꾸기
label_encoder = LabelEncoder()
label_encoded = label_encoder.fit_transform(values)
print(label_encoded)

# 원핫 인코딩 수행 - 범주형 문자를 0,1로 이루어진 벡터로 변경
print( label_encoded.shape ) # 1차원
onehot_encoder = OneHotEncoder(sparse=False)
lbl_encoded = label_encoded.reshape(len(label_encoded), 1)
print( lbl_encoded.shape ) # 2차원
```

```
onehot_encoded = onehot_encoder.fit_transform(lbl_encoded)
print(onehot_encoded)
```

```
['spring' 'spring' 'summer' 'spring' 'autumn' 'autumn' 'winter' 'spring'
 'summer' 'autumn']
[1 1 2 1 0 0 3 1 2 0]
(10,)
(10, 1)
[[0. 1. 0. 0.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]
 [0. 1. 0. 0.]
 [1. 0. 0. 0.]
 [1. 0. 0. 0.]
 [0. 0. 0. 1.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]
 [1. 0. 0. 0.]]
```

In [10]:
```
print( np.unique(values) )
print( onehot_encoded )
print( onehot_encoded[4] )    # 5번째 값 [1,0,0,0]

# 5번째 값 중에 가장 높은 값을 갖는 인덱스 확인
argmax(onehot_encoded[4, :])  # 5번째 값중에 [1]이 가장 크므로 인덱스 0 반환
```

```
['autumn' 'spring' 'summer' 'winter']
[[0. 1. 0. 0.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]
 [0. 1. 0. 0.]
 [1. 0. 0. 0.]
 [1. 0. 0. 0.]
 [0. 0. 0. 1.]
 [0. 1. 0. 0.]
 [0. 0. 1. 0.]
 [1. 0. 0. 0.]]
[1. 0. 0. 0.]
```

Out[10]:
```
0
```

In [11]:
```
# LabelEncoder에 입력하여 역변환 4번째 행의 값을 되돌리기
max_idx = [argmax(onehot_encoded[4, :])]
inverted = label_encoder.inverse_transform(max_idx)    # 만약 max_idx가 1이면 sp
print(inverted)
```

```
['autumn']
```

In [12]:
```
df = pd.DataFrame({"season":data, "lbl_season":label_encoded }, dtype=int)
onehot_val = pd.DataFrame(onehot_encoded, dtype=int)
onehot_val
df_new = pd.concat([df, onehot_val], axis=1)
df_new
```

Out[12]:

| | season | lbl_season | 0 | 1 | 2 | 3 |
|---|---|---|---|---|---|---|
| **0** | spring | 1 | 0 | 1 | 0 | 0 |
| **1** | spring | 1 | 0 | 1 | 0 | 0 |
| **2** | summer | 2 | 0 | 0 | 1 | 0 |
| **3** | spring | 1 | 0 | 1 | 0 | 0 |
| **4** | autumn | 0 | 1 | 0 | 0 | 0 |
| **5** | autumn | 0 | 1 | 0 | 0 | 0 |
| **6** | winter | 3 | 0 | 0 | 0 | 1 |

| | season | lbl_season | 0 | 1 | 2 | 3 |
|---|---|---|---|---|---|---|
| **7** | spring | 1 | 0 | 1 | 0 | 0 |
| **8** | summer | 2 | 0 | 0 | 1 | 0 |
| **9** | autumn | 0 | 1 | 0 | 0 | 0 |

## 03. 'hello world'를 원핫인코딩하기

목차로 이동하기

In [13]:
```
import numpy as np
from numpy import argmax
# define input string
data = 'hello world'
print(data)
```

```
hello world
```

In [14]:
```
# define universe of possible input values
alphabet = 'abcdefghijklmnopqrstuvwxyz '
# define a mapping of chars to integers
char_to_int = dict((c, i) for i, c in enumerate(alphabet))
int_to_char = dict((i, c) for i, c in enumerate(alphabet))

print("char_to_int : ", char_to_int)
print()
print("int_to_char : ", char_to_int)
```

```
char_to_int :  {'a': 0, 'b': 1, 'c': 2, 'd': 3, 'e': 4, 'f': 5, 'g': 6, 'h':
7, 'i': 8, 'j': 9, 'k': 10, 'l': 11, 'm': 12, 'n': 13, 'o': 14, 'p': 15, 'q':
16, 'r': 17, 's': 18, 't': 19, 'u': 20, 'v': 21, 'w': 22, 'x': 23, 'y': 24,
'z': 25, ' ': 26}

int_to_char :  {'a': 0, 'b': 1, 'c': 2, 'd': 3, 'e': 4, 'f': 5, 'g': 6, 'h':
7, 'i': 8, 'j': 9, 'k': 10, 'l': 11, 'm': 12, 'n': 13, 'o': 14, 'p': 15, 'q':
16, 'r': 17, 's': 18, 't': 19, 'u': 20, 'v': 21, 'w': 22, 'x': 23, 'y': 24,
'z': 25, ' ': 26}
```

In [15]:
```
# integer encode input data
integer_encoded = [char_to_int[char] for char in data]
print(integer_encoded)
```

```
[7, 4, 11, 11, 14, 26, 22, 14, 17, 11, 3]
```

In [16]:
```
# one hot encode
onehot_encoded = list()
for value in integer_encoded:
    letter = [0 for _ in range(len(alphabet))]
    letter[value] = 1
    onehot_encoded.append(letter)

print(onehot_encoded)
```

```
[[0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0], [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 1], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0,
```

```
0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]
```

In [17]:
```
# invert encoding
inverted = int_to_char[argmax(onehot_encoded[0])]
print(inverted)
```

```
h
```