Pandas 라이브러리 IRIS 데이터 셋 실습해보기

학습 내용

- map() 함수의 이해
- apply() 함수의 이해
- applymap() 함수의 이해
- groupby() 함수의 이해

01 데이터 준비

In [1]:

```
import pandas as pd
import seaborn as sns

print(pd.__version__)
iris = sns.load_dataset("iris")
iris
```

```
1.1.3
```

Out[1]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

150 rows × 5 columns

01. map - 데이터 프레임의 컬럼 변환

- Series.map()

In [2]:

```
iris.columns
```

Out[2]:

```
Index(['sepal_length', 'sepal_width', 'petal_length', 'petal_width',
       'species'],
      dtype='object')
```

In [3]:

```
iris.species.unique()
```

Out[3]:

```
array(['setosa', 'versicolor', 'virginica'], dtype=object)
```

In [4]:

```
ch_val = { 'setosa':0, 'versicolor':1, 'virginica':2 }
iris['species_num'] = iris['species'].map(ch_val)
iris
```

Out[4]:

	sepal_length	sepal_width	petal_length	petal_width	species	species_num
0	5.1	3.5	1.4	0.2	setosa	0
1	4.9	3.0	1.4	0.2	setosa	0
2	4.7	3.2	1.3	0.2	setosa	0
3	4.6	3.1	1.5	0.2	setosa	0
4	5.0	3.6	1.4	0.2	setosa	0
...	...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	virginica	2
146	6.3	2.5	5.0	1.9	virginica	2
147	6.5	3.0	5.2	2.0	virginica	2
148	6.2	3.4	5.4	2.3	virginica	2
149	5.9	3.0	5.1	1.8	virginica	2

150 rows × 6 columns

In [5]:

```
### 02. 데이터 값과 해당 개수 Count
iris.species_num.value_counts()
```

Out[5]:

```
2    50
1    50
0    50
Name: species_num, dtype: int64
```

02. apply() - 데이터프레임, 시리즈 모두 사용 가능

- Series.apply()
- DataFrame.apply()

In [6]:

```
iris.petal_width.mean()
```

Out[6]:

```
1.1993333333333334
```

petal_width의 평균보다 같거나 크면 1, 아니면 0으로 하는 컬럼 생성

In [7]:

```
iris["gt_petal_w"] = iris['petal_width'].apply(lambda v: 1 if v >= 1.0 else 0)
iris
```

Out[7]:

	sepal_length	sepal_width	petal_length	petal_width	species	species_num	gt_petal_w
0	5.1	3.5	1.4	0.2	setosa	0	0
1	4.9	3.0	1.4	0.2	setosa	0	0
2	4.7	3.2	1.3	0.2	setosa	0	0
3	4.6	3.1	1.5	0.2	setosa	0	0
4	5.0	3.6	1.4	0.2	setosa	0	0
...	...	...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	virginica	2	1
146	6.3	2.5	5.0	1.9	virginica	2	1
147	6.5	3.0	5.2	2.0	virginica	2	1
148	6.2	3.4	5.4	2.3	virginica	2	1
149	5.9	3.0	5.1	1.8	virginica	2	1

150 rows × 7 columns

In [8]:

```
iris["gt_petal_w"].value_counts()
```

Out[8]:

```
1    100
0     50
Name: gt_petal_w, dtype: int64
```

데이터 프레임 apply 함수 적용

- petal_length * petal_width 값을 갖는 컬럼 생성

In [9]:

```
# axis = 1 : 컬럼 방향 적용
iris['petal_lw'] = iris.apply(lambda x :
                    x['petal_length'] * x['petal_width'], axis=1)
iris
```

Out[9]:

	sepal_length	sepal_width	petal_length	petal_width	species	species_num	gt_petal_w	p
0	5.1	3.5	1.4	0.2	setosa	0	0	
1	4.9	3.0	1.4	0.2	setosa	0	0	
2	4.7	3.2	1.3	0.2	setosa	0	0	
3	4.6	3.1	1.5	0.2	setosa	0	0	
4	5.0	3.6	1.4	0.2	setosa	0	0	
...	...	...	...	...	...	...	...	
145	6.7	3.0	5.2	2.3	virginica	2	1	
146	6.3	2.5	5.0	1.9	virginica	2	1	
147	6.5	3.0	5.2	2.0	virginica	2	1	
148	6.2	3.4	5.4	2.3	virginica	2	1	
149	5.9	3.0	5.1	1.8	virginica	2	1	

150 rows × 8 columns

03. applymap() - 데이터프레임 전체에 데이터 셀 적용

- DataFrame.applymap()

전체 데이터의 log값을 적용하여 확인해 보자.

In [10]:

```
import numpy as np
```

In [11]:

```
# 값이 int형인지 알아봅니다.
print( isinstance(1, int) )

# 값이 str인지 알아봅니다.
print( isinstance("hello", str))

# 값이 float인지 알아봅니다.
print( isinstance(10.5, float) )
print( isinstance(10, float) )
```

```
True
True
True
False
```

In [12]:

```
iris.applymap(lambda v : np.log(v) if isinstance(v, float) else v)
```

Out[12]:

	sepal_length	sepal_width	petal_length	petal_width	species	species_num	gt_petal_w	
0	1.629241	1.252763	0.336472	-1.609438	setosa	0	0	-1
1	1.589235	1.098612	0.336472	-1.609438	setosa	0	0	-1
2	1.547563	1.163151	0.262364	-1.609438	setosa	0	0	-1
3	1.526056	1.131402	0.405465	-1.609438	setosa	0	0	-1
4	1.609438	1.280934	0.336472	-1.609438	setosa	0	0	-1
...	...	...	...	...	...	...	...	
145	1.902108	1.098612	1.648659	0.832909	virginica	2	1	2
146	1.840550	0.916291	1.609438	0.641854	virginica	2	1	2
147	1.871802	1.098612	1.648659	0.693147	virginica	2	1	2
148	1.824549	1.223775	1.686399	0.832909	virginica	2	1	2
149	1.774952	1.098612	1.629241	0.587787	virginica	2	1	2

150 rows × 8 columns

04. groupby() - 그룹별 통계 확인

- df.groupby("") : 지정된 컬럼의 값으로 그룹화시킵니다.
  - df.groupby("species").mean()
  - df.groupby("species").sum()
  - df.groupby("species").count()
  - df.groupby("species").median()

In [13]:

```python
iris.groupby('species')
```

Out[13]:

```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x0000018B29742760>
```

In [14]:

```python
iris.groupby('species').mean()
```

Out[14]:

	sepal_length	sepal_width	petal_length	petal_width	species_num	gt_petal_w	petal
species							
setosa	5.006	3.428	1.462	0.246	0	0	0.3
versicolor	5.936	2.770	4.260	1.326	1	1	5.7
virginica	6.588	2.974	5.552	2.026	2	1	11.2

In [15]:

```python
iris.groupby('species').sum()
```

Out[15]:

	sepal_length	sepal_width	petal_length	petal_width	species_num	gt_petal_w	petal
species							
setosa	250.3	171.4	73.1	12.3	0	0	18
versicolor	296.8	138.5	213.0	66.3	50	50	286
virginica	329.4	148.7	277.6	101.3	100	50	564

In [16]:

```
# petal_length로 묶어, 'species'값의 중복제외한 값을 확인
iris.groupby('petal_length')['species'].unique()
```

Out[16]:

```
petal_length
1.0                    [setosa]
1.1                    [setosa]
1.2                    [setosa]
1.3                    [setosa]
1.4                    [setosa]
1.5                    [setosa]
1.6                    [setosa]
1.7                    [setosa]
1.9                    [setosa]
3.0                [versicolor]
3.3                [versicolor]
3.5                [versicolor]
3.6                [versicolor]
3.7                [versicolor]
3.8                [versicolor]
3.9                [versicolor]
4.0                [versicolor]
4.1                [versicolor]
4.2                [versicolor]
4.3                [versicolor]
4.4                [versicolor]
4.5     [versicolor, virginica]
4.6                [versicolor]
4.7                [versicolor]
4.8     [versicolor, virginica]
4.9     [versicolor, virginica]
5.0     [versicolor, virginica]
5.1     [versicolor, virginica]
5.2                 [virginica]
5.3                 [virginica]
5.4                 [virginica]
5.5                 [virginica]
5.6                 [virginica]
5.7                 [virginica]
5.8                 [virginica]
5.9                 [virginica]
6.0                 [virginica]
6.1                 [virginica]
6.3                 [virginica]
6.4                 [virginica]
6.6                 [virginica]
6.7                 [virginica]
6.9                 [virginica]
Name: species, dtype: object
```

In [17]:

```
# to_frame() 함수를 통해 frame로 변환
iris.groupby('petal_length')['species'].unique().to_frame()
```

Out[17]:

	species
petal_length	
1.0	[setosa]
1.1	[setosa]
1.2	[setosa]
1.3	[setosa]
1.4	[setosa]
1.5	[setosa]
1.6	[setosa]
1.7	[setosa]
1.9	[setosa]
3.0	[versicolor]
3.3	[versicolor]
3.5	[versicolor]
3.6	[versicolor]
3.7	[versicolor]
3.8	[versicolor]
3.9	[versicolor]
4.0	[versicolor]
4.1	[versicolor]
4.2	[versicolor]
4.3	[versicolor]
4.4	[versicolor]
4.5	[versicolor, virginica]
4.6	[versicolor]
4.7	[versicolor]
4.8	[versicolor, virginica]
4.9	[versicolor, virginica]
5.0	[versicolor, virginica]
5.1	[versicolor, virginica]
5.2	[virginica]
5.3	[virginica]
5.4	[virginica]
5.5	[virginica]
5.6	[virginica]

	species
petal_length	
5.7	[virginica]
5.8	[virginica]
5.9	[virginica]
6.0	[virginica]
6.1	[virginica]
6.3	[virginica]
6.4	[virginica]
6.6	[virginica]
6.7	[virginica]
6.9	[virginica]