프로젝트 기반 빅데이터 서비스 솔루션 개발 전문과정

교과목명 : 분석라이브러리 활용

- 평가일 : 22.1.21
- 성명 : 오주완
- 점수 : 55

Q1. 표준정규분포 기반의 2행 3열 배열을 랜덤하게 생성하여 크기, 자료형, 차원을 출력하세요.

In [93]:

```
import numpy as np
ar = np.random.randn(2,3)
print(ar,ar.shape,ar.ndim)
```

```
[[-0.1639004   0.94480808 -0.2751556 ]
 [-0.60275229  0.2087622  -0.12993939]] (2, 3) 2
```

Q2. arange(), reshape() 이용 1차원 2차원 3차원 배열을 아래와 같이 생성하세요.

[0 1 2 3 4 5 6 7 8 9]

[[0 1 2 3 4]
[5 6 7 8 9]]

[[[0 1 2 3 4]
[5 6 7 8 9]]]

In [18]:

```
ar = np.array([0,1,2,3,4,5,6,7,8,9])
print(ar,'₩n')
ar1 = np.array([0,1,2,3,4,5,6,7,8,9]).reshape(2,-1)
print(ar1,'₩n')
ar2 = np.array([[[0,1,2,3,4],
                 [5,6,7,8,9]]])
print(ar2)
```

```
[0 1 2 3 4 5 6 7 8 9]

[[0 1 2 3 4]
 [5 6 7 8 9]]

[[[0 1 2 3 4]
  [5 6 7 8 9]]]
```

Q3. 1 ~ 100 까지 배열에서 3과 7의 공배수인 것만을 출력하세요.

In [12]:

```
ar = np.arange(1,101)
ar1 = ar[ar%21==0]
print(ar1)
```

```
[21 42 63 84]
```

Q4. 아래 3차원 배열을 생성하여 출력한 후 1차원으로 변환하여 출력하세요.(reshape() 사용)

[[[0 1 2 3 4]
[5 6 7 8 9]]

[[10 11 12 13 14]
[15 16 17 18 19]]

[[20 21 22 23 24]
[25 26 27 28 29]]]

In [105]:

```
ar = np.arange(30).reshape(3,2,-1)
ar1 = np.array([0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29])
ar1
```

Out[105]:

```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29])
```

Q5. array2d에서 인덱스를 이용해서 값을 선택하고 리스트로 아래와 같이 출력하세요.

arr2d = np.arange(1,10).reshape(3,3)

[3, 6]

[[1, 2],
[4, 5]]

[[1, 2, 3]
[4, 5, 6]]

In [108]:

```
arr2d = np.arange(1,10).reshape(3,3)
print(arr2d,'₩n')
print(arr2d[0:2,2:3])
print(arr2d[0:2,0:2])
print(arr2d[0:2,:])
```

```
[[1 2 3]
 [4 5 6]
 [7 8 9]]

[[3]
 [6]]
[[1 2]
 [4 5]]
[[1 2 3]
 [4 5 6]]
```

Q6. zeros_like, ones_like, full_like 함수 사용 예를 작성하세요.

Q7. 10 ~ 20 사이의 정수 난수로 10행 5열 2차원 배열을 생성하고 저장한 후 다시 불러내서 출력하세요.

In [116]:

```
import pandas as pd
ar = np.ones(10,5)
```

```
---------------------------------------------------------------------------
_
TypeError                                 Traceback (most recent call last)
~₩AppData₩Local₩Temp/ipykernel_1676/2894732184.py in <module>
      1 import pandas as pd
----> 2 ar = np.zeros(10.5)

TypeError: 'float' object cannot be interpreted as an integer
```

Q8. df = sns.load_dataset('titanic')로 불러와서 다음 작업을 수행한 후 출력하세요.

- 전체 칼럼중 'survived'외에 모든 칼럼을 포함한 df_x를 산출한 후 dataset/df_x.pkl로 저장한다.
- df_x.pkl을 데이터프레임 df_x 이름으로 불러온 후 앞 5개 행을 출력한다.

In [151]:

```
import seaborn as sns
df = sns.load_dataset('titanic')
df_x = df.drop('survived', axis=1, inplace=True)
df_x
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
~WAppDataWLocalWTemp/ipykernel_1676/2778763357.py in <module>
      1 import seaborn as sns
      2 df = sns.load_dataset('titanic')
----> 3 df_x = df.drop('[survived]', axis=1, inplace=True)
      4 df_x

C:WswWAnaconda3WenvsWcakd5WlibWsite-packagesWpandasWutilW_decorators.py in
wrapper(*args, **kwargs)
    309                     stacklevel=stacklevel,
    310                 )
--> 311             return func(*args, **kwargs)
    312
    313         return wrapper

C:WswWAnaconda3WenvsWcakd5WlibWsite-packagesWpandasWcoreWframe.py in drop(s
elf, labels, axis, index, columns, level, inplace, errors)
   4904                 weight  1.0     0.8
   4905         """
-> 4906         return super().drop(
   4907             labels=labels,
   4908             axis=axis,

C:WswWAnaconda3WenvsWcakd5WlibWsite-packagesWpandasWcoreWgeneric.py in drop
(self, labels, axis, index, columns, level, inplace, errors)
   4148         for axis, labels in axes.items():
   4149             if labels is not None:
-> 4150                 obj = obj._drop_axis(labels, axis, level=level, errors=e
rrors)
   4151
   4152         if inplace:

C:WswWAnaconda3WenvsWcakd5WlibWsite-packagesWpandasWcoreWgeneric.py in _dro
p_axis(self, labels, axis, level, errors)
   4183                 new_axis = axis.drop(labels, level=level, errors=errors)
   4184             else:
-> 4185                 new_axis = axis.drop(labels, errors=errors)
   4186             result = self.reindex(**{axis_name: new_axis})
   4187

C:WswWAnaconda3WenvsWcakd5WlibWsite-packagesWpandasWcoreWindexesWbase.py i
n drop(self, labels, errors)
   6015         if mask.any():
   6016             if errors != "ignore":
-> 6017                 raise KeyError(f"{labels[mask]} not found in axis")
   6018             indexer = indexer[~mask]
   6019         return self.delete(indexer)

KeyError: "['[survived]'] not found in axis"
```

Q9. df = sns.load_dataset('titanic')로 불러와서 deck 열에서 NaN 갯수를 계산하세요.

In [137]:

```
df = sns.load_dataset('titanic')
df1 = df.groupby('deck')
df1

def sumnan(df1):
    return df1.isnull().sum()

df2 = df.copy()
print(df2.pipe(sumnan),'₩n')
```

```
survived         0
pclass           0
sex              0
age            177
sibsp            0
parch            0
fare             0
embarked         2
class            0
who              0
adult_male       0
deck           688
embark_town      2
alive            0
alone            0
dtype: int64
```

Q10. Q9의 df에서 각 칼럼별 null 개수와 df 전체의 null 개수를 구하세요.

아래 tdf 데이터프레임에서 Q11 ~ Q12 작업을 수행하세요.

In [83]:

```
import seaborn as sns
df = sns.load_dataset('titanic')
tdf = df[['survived','sex','age','class']]
tdf.head()
```

Out[83]:

	survived	sex	age	class
0	0	male	22.0	Third
1	1	female	38.0	First
2	1	female	26.0	Third
3	1	female	35.0	First
4	0	male	35.0	Third

Q11. age를 7개 카테고리로 구분하는 새로운 칼럼 'cat_age'를 생성하여 출력하세요. 단, 카테고리 구분을 수행하는 사용자 함수를 만들고 그 함수를 age 칼럼에 매핑하여 결과를 tdf1에 저장하고 출력하세요.

[카테고리]
age <= 5: cat = 'Baby'
age <= 12: cat = 'Child'
age <= 18: cat = 'Teenager'
age <= 25: cat = 'Student'
age <= 60: cat = 'Adult'
age > 60 : cat = 'Elderly'

In [129]:

```
def cat_age(age):
    cat=''
    if age <= 5: cat = 'Baby'
    elif age <= 12: cat = 'Child'
    elif age <= 18: cat = 'Teenager'
    elif age <= 25: cat = 'student'
    elif age <= 60: cat = 'Adult'
    else:
        cat = 'Elderly'
    return cat
tdf['age_cat'] = tdf.age.apply(lambda x: cat_age(x))
tdf[['age','age_cat']].head()
```

Out[129]:

	age	age_cat
0	22.0	student
1	38.0	Adult
2	26.0	Adult
3	35.0	Adult
4	35.0	Adult

Q12. tdf1의 sex, class 칼럼을 '_'으로 연결한 'sc'칼럼을 추가한 후 아래와 같이 출력하세요.

#male_third

Q13. join() 메소드는 두 데이터프레임의 행 인덱스를 기준으로 결합한다. 2개의 주식데이터를 가져와서 join() 메소드로 아래와 같이 결합한 후 다음 사항을 수행하세요.

- df1과 df2의 교집합만 출력되도록 결합하여 df3에 저장하고 출력
- df3에서 중복된 칼럼을 삭제한 후 블린 인덱싱을 이용하여 eps가 3000 보다 적거나 stock_name이 이마트인 데이터를 선택하여 데이터프레임을 생성하고 df4 이름으로 저장 및 출력하세요.(단, '<' 와 '==' 를 반드시 사용해야 함)

In [134]:

```
df1 = pd.read_excel('./dataset/stock price.xlsx', index_col='id')
df2 = pd.read_excel('./dataset/stock valuation.xlsx', index_col='id')
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
~WAppDataWLocalWTemp/ipykernel_1676/3366064677.py in <module>
----> 1 df1 = pd.read_excel('./dataset/stock price.xlsx', index_col='id')
      2 df2 = pd.read_excel('./dataset/stock valuation.xlsx', index_col='id'
)

C:WswWAnaconda3WenvsWcakd5WlibWsite-packagesWpandasWutilW_decorators.py
 in wrapper(*args, **kwargs)
    309                     stacklevel=stacklevel,
    310                 )
--> 311             return func(*args, **kwargs)
    312
    313         return wrapper

C:WswWAnaconda3WenvsWcakd5WlibWsite-packagesWpandasWioWexcelW_base.py in
read_excel(io, sheet_name, header, names, index_col, usecols, squeeze, dt
ype, engine, converters, true_values, false_values, skiprows, nrows, na
_values, keep_default_na, na_filter, verbose, parse_dates, date_parser,
thousands, comment, skipfooter, convert_float, mangle_dupe_cols, storag
e_options)
    362     if not isinstance(io, ExcelFile):
    363         should_close = True
--> 364         io = ExcelFile(io, storage_options=storage_options, engine=engi
ne)
    365     elif engine and engine != io.engine:
    366         raise ValueError(

C:WswWAnaconda3WenvsWcakd5WlibWsite-packagesWpandasWioWexcelW_base.py in
__init__(self, path_or_buffer, engine, storage_options)
   1189                 ext = "xls"
   1190             else:
-> 1191                 ext = inspect_excel_format(
   1192                     content_or_path=path_or_buffer, storage_options=stora
ge_options
   1193                 )

C:WswWAnaconda3WenvsWcakd5WlibWsite-packagesWpandasWioWexcelW_base.py in
inspect_excel_format(content_or_path, storage_options)
   1068         content_or_path = BytesIO(content_or_path)
   1069
-> 1070     with get_handle(
   1071         content_or_path, "rb", storage_options=storage_options, is_text=
False
   1072     ) as handle:

C:WswWAnaconda3WenvsWcakd5WlibWsite-packagesWpandasWioWcommon.py in get_h
andle(path_or_buf, mode, encoding, compression, memory_map, is_text, err
ors, storage_options)
    709         else:
    710             # Binary mode
--> 711             handle = open(handle, ioargs.mode)
    712         handles.append(handle)
    713
```

```
FileNotFoundError: [Errno 2] No such file or directory: './dataset/stock price.
xlsx'
```

Q14. 배열 a에 대하여 3차원 자리에 2차원을 2차원 자리에 1차원을 1차원 자리에 3차원을 넣어서 변환하여 출력하세요

In [73]:

```
a = np.arange(6).reshape(1,2,3)
print(a,a.shape,'₩n')
```

```
[[[0 1 2]
  [3 4 5]]] (1, 2, 3)
```

In [133]:

```
a1 = np.transpose(a,(1,2,0))
print(a1,a1.shape)
```

```
[[[0]
  [1]
  [2]]

 [[3]
  [4]
  [5]]] (2, 3, 1)
```

Q15. 'mpg'를 'kpl' 로 환산하여 새로운 열을 생성하고 반올림하여 소수점 아래 둘째 자리까지 처음 5개행을 출력하세요.

In [16]:

```
# read_csv() 함수로 df 생성
import pandas as pd
auto_df = pd.read_csv('./dataset/auto-mpg.csv')
# 열 이름을 지정
auto_df.columns = ['mpg','cylinders','displacement','horsepower','weight',
                   'acceleration','model year','origin','name']
print(auto_df.head(3))
```

```
    mpg  cylinders  displacement horsepower  weight  acceleration  model year  ₩
0  18.0          8         307.0        130    3504          12.0          70
1  15.0          8         350.0        165    3693          11.5          70
2  18.0          8         318.0        150    3436          11.0          70

   origin                       name
0       1  chevrolet chevelle malibu
1       1          buick skylark 320
2       1         plymouth satellite
```

Q16. './dataset/stock-data.csv'를 데이터프레임으로 불러와서 datetime64 자료형으로 변환한 후에 년, 월, 일로 분리하고 year를 인덱스로 셋팅하여 출력하세요.

In [135]:

```
df = pd.read_csv('./dataset/stock-data.csv')
df
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_1676/208673092.py in <module>
----> 1 df = pd.read_csv('./dataset/stock-data.csv')
      2 df

C:\sw\Anaconda3\envs\cakd5\lib\site-packages\pandas\util\_decorators.py in
wrapper(*args, **kwargs)
    309                     stacklevel=stacklevel,
    310                 )
--> 311             return func(*args, **kwargs)
    312
    313         return wrapper

C:\sw\Anaconda3\envs\cakd5\lib\site-packages\pandas\io\parsers\readers.py
 in read_csv(filepath_or_buffer, sep, delimiter, header, names, index_col,
 usecols, squeeze, prefix, mangle_dupe_cols, dtype, engine, converters, tr
ue_values, false_values, skipinitialspace, skiprows, skipfooter, nrows, na
_values, keep_default_na, na_filter, verbose, skip_blank_lines, parse_date
s, infer_datetime_format, keep_date_col, date_parser, dayfirst, cache_date
s, iterator, chunksize, compression, thousands, decimal, lineterminator, q
uotechar, quoting, doublequote, escapechar, comment, encoding, encoding_er
rors, dialect, error_bad_lines, warn_bad_lines, on_bad_lines, delim_whites
pace, low_memory, memory_map, float_precision, storage_options)
    584     kwds.update(kwds_defaults)
    585
--> 586     return _read(filepath_or_buffer, kwds)
    587
    588

C:\sw\Anaconda3\envs\cakd5\lib\site-packages\pandas\io\parsers\readers.py
 in _read(filepath_or_buffer, kwds)
    480
    481     # Create the parser.
--> 482     parser = TextFileReader(filepath_or_buffer, **kwds)
    483
    484     if chunksize or iterator:

C:\sw\Anaconda3\envs\cakd5\lib\site-packages\pandas\io\parsers\readers.py
 in __init__(self, f, engine, **kwds)
    809             self.options["has_index_names"] = kwds["has_index_names"]
    810
--> 811         self._engine = self._make_engine(self.engine)
    812
    813     def close(self):

C:\sw\Anaconda3\envs\cakd5\lib\site-packages\pandas\io\parsers\readers.py
 in _make_engine(self, engine)
   1038             )
   1039         # error: Too many arguments for "ParserBase"
-> 1040         return mapping[engine](self.f, **self.options)  # type: ignore
[call-arg]
   1041
   1042     def _failover_to_python(self):
```

```
C:₩sw₩Anaconda3₩envs₩cakd5₩lib₩site-packages₩pandas₩io₩parsers₩c_parser_wr
apper.py in __init__(self, src, **kwds)
     49
     50         # open handles
---> 51           self._open_handles(src, kwds)
     52         assert self.handles is not None
     53

C:₩sw₩Anaconda3₩envs₩cakd5₩lib₩site-packages₩pandas₩io₩parsers₩base_parse
r.py in _open_handles(self, src, kwds)
    220         Let the readers open IOHandles after they are done with their potent
ial raises.
    221         """
--> 222           self.handles = get_handle(
    223             src,
    224             "r",

C:₩sw₩Anaconda3₩envs₩cakd5₩lib₩site-packages₩pandas₩io₩common.py in get_han
dle(path_or_buf, mode, encoding, compression, memory_map, is_text, errors,
storage_options)
    700         if ioargs.encoding and "b" not in ioargs.mode:
    701             # Encoding
--> 702             handle = open(
    703                 handle,
    704                 ioargs.mode,

FileNotFoundError: [Errno 2] No such file or directory: './dataset/stock-data.csv'
```

Q17. titanic 데이터셋에서 5개 열을 선택해서 class열을 기준으로 그룹화를 수행한 후 아래와 같이 출력하였다. 다음 사항을 출력하세요.

5개 열 : ['age','sex', 'class', 'fare', 'survived']

- 그룹별 평균 출력
- 그룹별 최대값 출력

In [78]:

```
titanic = sns.load_dataset('titanic')
df = titanic.loc[:,['age','sex','class','fare','survived']]
grouped = df.groupby(['class'])
grouped.mean()
grouped.max()
```

Out[78]:

class	age	sex	fare	survived
First	80.0	male	512.3292	1
Second	70.0	male	73.5000	1
Third	74.0	male	69.5500	1

Q18. titanic 데이터셋에서 'Third'그룹만을 선택해서 group3 이름으로 저장하고 통계

요약표를 출력하세요.

In [75]:

```
import seaborn as sns
df = sns.load_dataset('titanic')
df.head()
```

Out[75]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True

In [152]:

```
group3 = grouped.get_group('Third')

group3.describe()
```

Out[152]:

	age	fare	survived
count	355.000000	491.000000	491.000000
mean	25.140620	13.675550	0.242363
std	12.495398	11.778142	0.428949
min	0.420000	0.000000	0.000000
25%	18.000000	7.750000	0.000000
50%	24.000000	8.050000	0.000000
75%	32.000000	15.500000	0.000000
max	74.000000	69.550000	1.000000

Q19. titanic 데이터셋에서 다음 전처리를 수행하세요.

1. df에서 중복 칼럼으로 고려할 수 있는 컬럼들(6개 내외)을 삭제한 후 나머지 컬럼들로 구성되는 데이터프레임을 df1 이름으로 저장 후 출력하세요.
2. df1에서 null값이 50% 이상인 칼럼을 삭제 후 df2 이름으로 저장하고 출력하세요.
3. df2에서 결측값이 있는 age 칼럼에 대해서 평균값으로 대체 처리를 수행하세요.
4. df2에서 결측값이 있는 embarked 칼럼에 대해서 앞행의 값으로 대체 처리를 수행하세요.
5. df2 문자로 되어있는 칼럼들을 레이블 인코딩 수행하여 숫자로 변환 후 df2.info()를 출력하세요

In [25]:

```
df.head()
```

Out[25]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True

Q20. 보스톤 주택가격 데이터를 탐색한 후 가장 중요한 독립변수 2개를 선정하고 그 이유를 시각화하여 설명하세요.

In [32]:

```
from sklearn.datasets import load_boston
# boston 데이타셋 로드
boston = load_boston()

# boston 데이타셋 DataFrame 변환
bostonDF = pd.DataFrame(boston.data , columns = boston.feature_names)

# boston dataset의 target array는 주택 가격임. 이를 PRICE 컬럼으로 DataFrame에 추가함.
bostonDF['PRICE'] = boston.target
print('Boston 데이타셋 크기 :',bostonDF.shape)
bostonDF.head()
```

```
Boston 데이타셋 크기 : (506, 14)
```

Out[32]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LS
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	396.90	4
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	396.90	9
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	392.83	4
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	394.63	2
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	18.7	396.90	5

In [ ]:

```

```