In [2]:
```
import numpy as np
import pandas as pd
import matplotlib as mpl
import matplotlib.pyplot as plt
import mglearn
%matplotlib inline
import seaborn as sns
import platform
from matplotlib import font_manager , rc

if platform.system() == 'Darwin':
  rc('font' , family = 'AppleGothic')
elif platform.system() == 'Windows':
  path = 'C:/Windows/Fonts/malgun.ttf'
  font_name = font_manager.FontProperties(fname = path).get_name()
  rc('font' , family = font_name)
else:
  print('모름')
plt.rcParams['axes.unicode_minus'] = False
import warnings
warnings.filterwarnings('ignore')
```
executed in 2.63s, finished 12:23:30 2023-11-03

In [9]:
```
retail = pd.read_csv('OnlineRetail.csv' , encoding = 'ISO-8859-1')
```
executed in 429ms, finished 12:28:42 2023-11-03

In [10]:
```
retail.columns
```
executed in 15ms, finished 12:28:45 2023-11-03

Out[10]:
```
Index(['InvoiceNo', 'StockCode', 'Description', 'Quantity', 'InvoiceDate',
       'UnitPrice', 'CustomerID', 'Country'],
      dtype='object')
```

feature list

- InvoiceNo : 주문번호
- StockCode : 상품코드
- Description : 상품설명
- Quantity : 주문수량
- InvoiceDate : 주문날짜
- UnitPrice : 상품가격
- CustomerID : 고객아이디
- Country : 나라

In [11]:
```
retail.info()
```
executed in 237ms, finished 12:31:36 2023-11-03

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 541909 entries, 0 to 541908
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype  
---  ------       --------------   -----  
 0   InvoiceNo    541909 non-null  object 
 1   StockCode    541909 non-null  object 
 2   Description  540455 non-null  object 
 3   Quantity     541909 non-null  int64  
 4   InvoiceDate  541909 non-null  object 
 5   UnitPrice    541909 non-null  float64
 6   CustomerID   406829 non-null  float64
 7   Country      541909 non-null  object 
dtypes: float64(2), int64(1), object(5)
memory usage: 33.1+ MB
```

In [12]:
```
retail.describe().T
```
executed in 51ms, finished 12:32:44 2023-11-03

Out[12]:

	count	mean	std	min	25%	50%	75%	max
Quantity	541909.0	9.552250	218.081158	-80995.00	1.00	3.00	10.00	80995.0
UnitPrice	541909.0	4.611114	96.759853	-11062.06	1.25	2.08	4.13	38970.0
CustomerID	406829.0	15287.690570	1713.600303	12346.00	13953.00	15152.00	16791.00	18287.0

In [19]:
```
retail.dropna(inplace = True)
```
executed in 251ms, finished 12:35:48 2023-11-03

In [26]:
```
retail = retail[retail['Quantity'] > 0]
```
executed in 35ms, finished 12:38:09 2023-11-03

In [28]:
```
retail = retail[retail['UnitPrice'] > 0]
```
executed in 34ms, finished 12:38:25 2023-11-03

In [31]:
```
retail.info()
```
executed in 170ms, finished 12:38:48 2023-11-03

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 397884 entries, 0 to 541908
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype  
---  ------       --------------   -----  
 0   InvoiceNo    397884 non-null  object 
 1   StockCode    397884 non-null  object 
 2   Description  397884 non-null  object 
 3   Quantity     397884 non-null  int64  
 4   InvoiceDate  397884 non-null  object 
 5   UnitPrice    397884 non-null  float64
 6   CustomerID   397884 non-null  float64
 7   Country      397884 non-null  object 
dtypes: float64(2), int64(1), object(5)
memory usage: 27.3+ MB
```

In [32]:
```
retail
```
executed in 29ms, finished 12:40:50 2023-11-03

Out[32]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	01-12-2010 08:26	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	01-12-2010 08:26	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	01-12-2010 08:26	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	01-12-2010 08:26	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	01-12-2010 08:26	3.39	17850.0	United Kingdom
...	...	...	...	...	...	...	...	...
541904	581587	22613	PACK OF 20 SPACEBOY NAPKINS	12	09-12-2011 12:50	0.85	12680.0	France
541905	581587	22899	CHILDREN'S APRON DOLLY GIRL	6	09-12-2011 12:50	2.10	12680.0	France
541906	581587	23254	CHILDRENS CUTLERY DOLLY GIRL	4	09-12-2011 12:50	4.15	12680.0	France
541907	581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	4	09-12-2011 12:50	4.15	12680.0	France
541908	581587	22138	BAKING SET 9 PIECE RETROSPOT	3	09-12-2011 12:50	4.95	12680.0	France

397884 rows × 8 columns

In [40]:
```
retail['Price'] = retail['Quantity'] * retail['UnitPrice']
```
executed in 18ms, finished 12:44:46 2023-11-03

In [41]:
```
retail.drop(['Quantity','UnitPrice'] , axis = 1 , inplace = True)
```
executed in 40ms, finished 12:45:03 2023-11-03

In [42]:
```
retail
```
executed in 16ms, finished 12:45:06 2023-11-03

Out[42]:

	InvoiceNo	StockCode	Description	InvoiceDate	CustomerID	Country	Price
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	01-12-2010 08:26	17850.0	United Kingdom	15.30
1	536365	71053	WHITE METAL LANTERN	01-12-2010 08:26	17850.0	United Kingdom	20.34
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	01-12-2010 08:26	17850.0	United Kingdom	22.00
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	01-12-2010 08:26	17850.0	United Kingdom	20.34
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	01-12-2010 08:26	17850.0	United Kingdom	20.34
...	...	...	...	...	...	...	...
541904	581587	22613	PACK OF 20 SPACEBOY NAPKINS	09-12-2011 12:50	12680.0	France	10.20
541905	581587	22899	CHILDREN'S APRON DOLLY GIRL	09-12-2011 12:50	12680.0	France	12.60
541906	581587	23254	CHILDRENS CUTLERY DOLLY GIRL	09-12-2011 12:50	12680.0	France	16.60
541907	581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	09-12-2011 12:50	12680.0	France	16.60
541908	581587	22138	BAKING SET 9 PIECE RETROSPOT	09-12-2011 12:50	12680.0	France	14.85

397884 rows × 7 columns

In [45]:
```
from datetime import datetime
retail.InvoiceDate = retail.InvoiceDate.apply(lambda x : datetime.strptime(x, '%d-%m-%Y %H:%M'))
retail['year'] = retail.InvoiceDate.apply(lambda x : x.year)
retail['month'] = retail.InvoiceDate.apply(lambda x : x.month)
retail['day'] = retail.InvoiceDate.apply(lambda x : x.day)
retail['hour'] = retail.InvoiceDate.apply(lambda x: x.hour)
```
executed in 5.18s, finished 14:04:26 2023-11-03

In [47]:
```
retail.drop('InvoiceDate' , inplace = True , axis = 1)
```
executed in 45ms, finished 14:05:04 2023-11-03

In [48]:
```
retail.head()
```
executed in 27ms, finished 14:05:09 2023-11-03

Out[48]:

	InvoiceNo	StockCode	Description	CustomerID	Country	Price	year	month	day	hour
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	17850.0	United Kingdom	15.30	2010	12	1	8
1	536365	71053	WHITE METAL LANTERN	17850.0	United Kingdom	20.34	2010	12	1	8
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	17850.0	United Kingdom	22.00	2010	12	1	8
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	17850.0	United Kingdom	20.34	2010	12	1	8
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	17850.0	United Kingdom	20.34	2010	12	1	8

In [158]:
```
a = retail.groupby(['CustomerID','year','month','day'])['Price'].sum()
```
executed in 61ms, finished 15:02:53 2023-11-03

In [159]:
```
a = pd.DataFrame(a)
```
executed in 14ms, finished 15:02:55 2023-11-03

In [160]:
```
total = a.reset_index()
```
executed in 22ms, finished 15:02:57 2023-11-03

In [161]:
```
total['count'] = 1
```
executed in 14ms, finished 15:02:59 2023-11-03

In [162]:
```
visit = pd.DataFrame(total.groupby('CustomerID')['count'].sum()).reset_index()
```
executed in 11ms, finished 15:03:01 2023-11-03

In [172]:
```
visit
```
executed in 21ms, finished 15:03:55 2023-11-03

Out[172]:

	CustomerID	count
0	12346.0	1
1	12347.0	7
2	12348.0	4
3	12349.0	1
4	12350.0	1
...	...	...
4333	18280.0	1
4334	18281.0	1
4335	18282.0	2
4336	18283.0	14
4337	18287.0	3

4338 rows × 2 columns

In [164]:
```
top20 = visit.sort_values(by = 'count' , ascending = False).head(20)
```
executed in 20ms, finished 15:03:05 2023-11-03

In [165]:
```
aa = visit.sort_values(by = 'count' , ascending = False)['count'].value_counts()
```
executed in 14ms, finished 15:03:06 2023-11-03

In [166]:
```
aa.values
```
executed in 13ms, finished 15:03:08 2023-11-03

Out[166]:
```
array([1548,  874,  501,  389,  227,  184,  132,   86,   67,   48,   42,
         36,   28,   26,   24,   19,   17,   11,   10,    7,    6,    5,
          5,    5,    4,    3,    3,    3,    2,    2,    2,    2,    2,
          1,    1,    1,    1,    1,    1,    1,    1,    1,    1,    1,
          1,    1,    1,    1,    1,    1,    1], dtype=int64)
```

In [169]:
```
aa
```
executed in 14ms, finished 15:03:25 2023-11-03

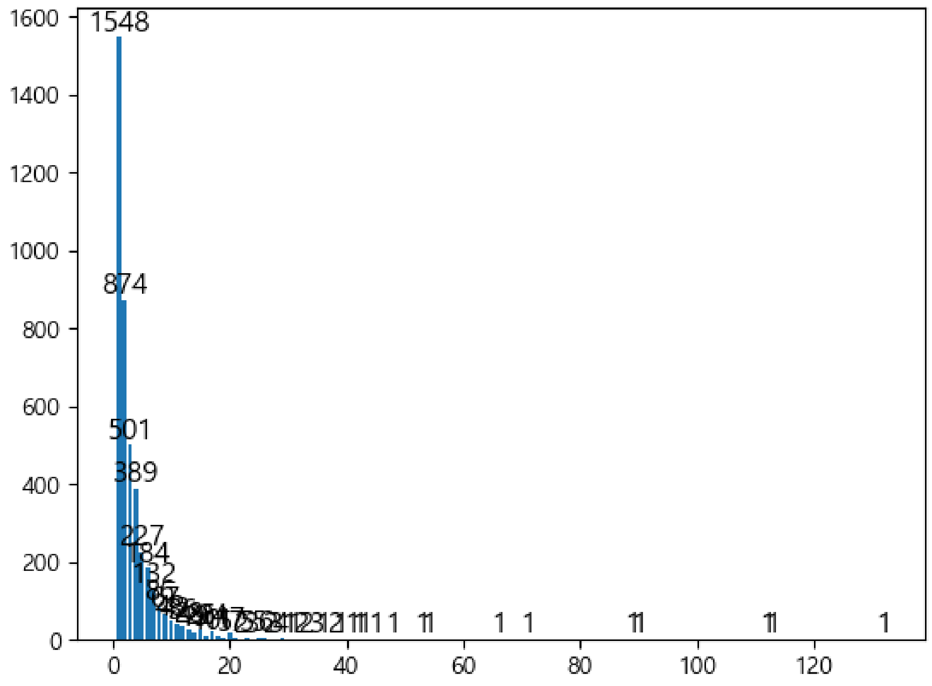
Out[169]:
```
1      1548
2       874
3       501
4       389
5       227
6       184
7       132
8        86
9        67
10       48
11       42
12       36
13       28
15       26
17       24
14       19
20       17
18       11
16       10
21        7
26        6
19        5
25        5
23        5
29        4
28        3
35        3
24        3
38        2
33        2
32        2
22        2
27        2
48        1
90        1
112       1
89        1
71        1
66        1
54        1
53        1
30        1
45        1
43        1
42        1
41        1
39        1
36        1
31        1
113       1
132       1
Name: count, dtype: int64
```

In [167]:
```
aa.keys()
```
executed in 7ms, finished 15:03:10 2023-11-03

Out[167]:
```
Int64Index([  1,   2,   3,   4,   5,   6,   7,   8,   9,  10,  11,  12,  13,
             15,  17,  14,  20,  18,  16,  21,  26,  19,  25,  23,  29,  28,
             35,  24,  38,  33,  32,  22,  27,  48,  90, 112,  89,  71,  66,
             54,  53,  30,  45,  43,  42,  41,  39,  36,  31, 113, 132],
           dtype='int64')
```

In [168]:
```
bar = plt.bar(x = aa.keys() , height = aa.values)


for rect in bar:
    height = rect.get_height()
    plt.text(rect.get_x()+rect.get_width()/2.0,height,  height, ha = 'center',va='bottom',size=12)
```
executed in 231ms, finished 15:03:11 2023-11-03



In [187]:
```
total = pd.merge(visit , pd.DataFrame(total.groupby('CustomerID')['Price'].sum()).reset_index() , on
```
executed in 11ms, finished 15:07:56 2023-11-03

In [260]:
```
total2 = total[['count','Price']]
```
executed in 15ms, finished 15:31:40 2023-11-03

In [261]:
```
total2['Price'] = np.log1p(total2['Price'])
```
executed in 8ms, finished 15:31:47 2023-11-03

In [262]:
```
from sklearn.cluster import KMeans
```
executed in 6ms, finished 15:31:48 2023-11-03

In [263]:
```
model = KMeans(n_clusters = 3 , init = 'k-means++' , max_iter = 300 , random_state = 0)
model.fit(total2)
```
executed in 219ms, finished 15:31:48 2023-11-03

Out[263]:
```
▾              KMeans
KMeans(n_clusters=3, random_state=0)
```

In [264]:
```
model.labels_
```
executed in 12ms, finished 15:31:50 2023-11-03

Out[264]:
```
array([0, 0, 0, ..., 0, 1, 0])
```

In [265]:
```
total2['grade'] = model.labels_
```
executed in 5ms, finished 15:31:50 2023-11-03

In [266]:
```
total2['CustomerID'] = total['CustomerID']
```
executed in 9ms, finished 15:31:52 2023-11-03

In [267]:
```
total2
```
executed in 20ms, finished 15:31:52 2023-11-03

Out[267]:

	count	Price	grade	CustomerID
0	1	11.253955	0	12346.0
1	7	8.368925	0	12347.0
2	4	7.494564	0	12348.0
3	1	7.472245	0	12349.0
4	1	5.815324	0	12350.0
...	...	...	...	...
4333	1	5.201806	0	18280.0
4334	1	4.404522	0	18281.0
4335	2	5.187665	0	18282.0
4336	14	7.647729	1	18283.0
4337	3	7.516586	0	18287.0

4338 rows × 4 columns

In [274]:
```
total2['Price'] = np.exp(total2['Price']) - 1
```
executed in 7ms, finished 15:33:09 2023-11-03

In [231]:
```
total2[total2['grade'] == 0].sort_values(by = 'Price')
```
executed in 19ms, finished 15:14:58 2023-11-03

Out[231]:

	count	Price	grade	CustomerID
3217	1	3.75	0	16738.0
1793	1	6.20	0	14792.0
3014	2	6.90	0	16454.0
4098	1	12.75	0	17956.0
3323	1	13.30	0	16878.0
...	...	...	...	...
330	6	21429.39	0	12753.0
2011	1	39916.50	0	15098.0
2502	2	44534.30	0	15749.0
0	1	77183.60	0	12346.0
3008	2	168472.50	0	16446.0

3941 rows × 4 columns

In [232]:
```
total2[total2['grade'] == 1].sort_values(by = 'Price')
```
executed in 18ms, finished 15:14:58 2023-11-03

Out[232]:

	count	Price	grade	CustomerID
4014	9	901.20	1	17848.0
3987	9	1199.01	1	17800.0
853	10	1215.82	1	13491.0
3264	9	1222.71	1	16794.0
4102	35	1296.44	1	17961.0
...	...	...	...	...
1333	43	117379.63	1	14156.0
55	16	124914.53	1	12415.0
3728	27	194550.79	1	17450.0
4201	26	259657.30	1	18102.0
1689	45	280206.02	1	14646.0

388 rows × 4 columns

In [233]:
```
total2[total2['grade'] == 2].sort_values(by = 'Price')
```
executed in 17ms, finished 15:14:58 2023-11-03

Out[233]:

	count	Price	grade	CustomerID
1602	54	8508.82	2	14527.0
481	71	11189.91	2	12971.0
1661	89	12156.65	2	14606.0
326	113	33719.73	2	12748.0
1069	53	37153.85	2	13798.0
4010	112	40991.57	2	17841.0
562	66	58825.83	2	13089.0
2176	90	60767.90	2	15311.0
1879	132	143825.06	2	14911.0

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [275]:
```
mapping = {0: '일반고객', 1: '우수고객', 2: '최우수고객'}
total2['grade'] = total2['grade'].map(mapping)
```
executed in 14ms, finished 15:33:12 2023-11-03

In [235]:
```
total2[total2['grade'] == '일반고객'].sort_values(by = 'Price')
```
executed in 28ms, finished 15:15:15 2023-11-03

Out[235]:

	count	Price	grade	CustomerID
3217	1	3.75	일반고객	16738.0
1793	1	6.20	일반고객	14792.0
3014	2	6.90	일반고객	16454.0
4098	1	12.75	일반고객	17956.0
3323	1	13.30	일반고객	16878.0
...	...	...	...	...
330	6	21429.39	일반고객	12753.0
2011	1	39916.50	일반고객	15098.0
2502	2	44534.30	일반고객	15749.0
0	1	77183.60	일반고객	12346.0
3008	2	168472.50	일반고객	16446.0

3941 rows × 4 columns

In [236]:
```
total2[total2['grade'] == '우수고객'].sort_values(by = 'Price')
```
executed in 24ms, finished 15:15:18 2023-11-03

Out[236]:

	count	Price	grade	CustomerID
4014	9	901.20	우수고객	17848.0
3987	9	1199.01	우수고객	17800.0
853	10	1215.82	우수고객	13491.0
3264	9	1222.71	우수고객	16794.0
4102	35	1296.44	우수고객	17961.0
...	...	...	...	...
1333	43	117379.63	우수고객	14156.0
55	16	124914.53	우수고객	12415.0
3728	27	194550.79	우수고객	17450.0
4201	26	259657.30	우수고객	18102.0
1689	45	280206.02	우수고객	14646.0

388 rows × 4 columns

In [237]:
```
total2[total2['grade'] == '최우수고객']
```
executed in 12ms, finished 15:15:29 2023-11-03

Out[237]:

	count	Price	grade	CustomerID
326	113	33719.73	최우수고객	12748.0
481	71	11189.91	최우수고객	12971.0
562	66	58825.83	최우수고객	13089.0
1069	53	37153.85	최우수고객	13798.0
1602	54	8508.82	최우수고객	14527.0
1661	89	12156.65	최우수고객	14606.0
1879	132	143825.06	최우수고객	14911.0
2176	90	60767.90	최우수고객	15311.0
4010	112	40991.57	최우수고객	17841.0

In [239]:
```
total2.columns
```
executed in 12ms, finished 15:18:39 2023-11-03

Out[239]:
```
Index(['count', 'Price', 'grade', 'CustomerID'], dtype='object')
```

In [272]:
```
total2 = total2[['grade','CustomerID','count','Price']]
```
executed in 11ms, finished 15:32:45 2023-11-03

In [268]:
```
total2
```
executed in 17ms, finished 15:32:11 2023-11-03

Out[268]:

	count	Price	grade	CustomerID
0	1	11.253955	0	12346.0
1	7	8.368925	0	12347.0
2	4	7.494564	0	12348.0
3	1	7.472245	0	12349.0
4	1	5.815324	0	12350.0
...	...	...	...	...
4333	1	5.201806	0	18280.0
4334	1	4.404522	0	18281.0
4335	2	5.187665	0	18282.0
4336	14	7.647729	1	18283.0
4337	3	7.516586	0	18287.0

4338 rows × 4 columns

In [259]:
```
```
executed in 20ms, finished 15:31:23 2023-11-03

Out[259]:
```
array([77183.6 ,  4310.  ,  1797.24, ...,   178.05,  2094.88,  1837.28])
```

In [243]:
```
from sklearn.metrics import silhouette_samples , silhouette_score
```
executed in 12ms, finished 15:29:02 2023-11-03

In [271]:
```
score_samples = silhouette_samples(total2[['Price','count']] , total2['grade'])
```
executed in 258ms, finished 15:32:37 2023-11-03

```
In [273]: total2['실루엣계수'] = score_samples
```
executed in 14ms, finished 15:32:59 2023-11-03

```
In [277]: total2['실루엣계수'].mean()
```
executed in 18ms, finished 15:33:35 2023-11-03

```
Out[277]: 0.7504617090297463
```

```
In [278]: total2.loc[total2['grade'] == '일반고객' , '실루엣계수'].mean()
```
executed in 10ms, finished 16:27:38 2023-11-03

```
Out[278]: 0.7843206855933095
```

```
In [279]: total2.loc[total2['grade'] == '우수고객' , '실루엣계수'].mean()
```
executed in 9ms, finished 16:27:46 2023-11-03

```
Out[279]: 0.4132199232645537
```

```
In [280]: total2.loc[total2['grade'] == '최우수고객' , '실루엣계수'].mean()
```
executed in 15ms, finished 16:27:49 2023-11-03

```
Out[280]: 0.4628601801288574
```

```
In [ ]:
```