데이터 가공 및 시각화

- 1. 전처리
  - 결측값 처리: 단순대치, 평균 대치, 단순확률 대치 (Hot-deck, nearest neighbor), 다중 대치, knnImputation, centralimputation
  - 클래스불균형: 업샘플링 (SMOTE, Boaderline SMOTE, Adasyn), 다운샘플링
  - 이상값 처리: 극단값 절단, 조정
  - 변수 변환, 스케일링: 수치형 변수 변환(로그변환, 제곱근변환, 지수변환, 제곱변환, Box-cox 변환, 표준화, 정규화), 범주형 변수 변환(범주형 변수 인코딩, 대규모 범주형 변수처리), 날짜 및 변수 변환, 피쳐스케일링
  - 원핫인코딩(더미변수), 컬럼 트랜스퍼, 구간분할, 이산화, 피쳐선택
- 1. **표본 추출: 단순랜덤 추출법, 계통추출법, 집락추출법, 층화추출법**
- 1. 데이터 분할: 구축/검정/시험용, 홀드아웃방법, 교차확인방법 (10 fold 교차분석), 부트스트랩
- 1. 그래프 그리기:
  - 산점도, 막대그래프, 선그래프, 히트맵, 서브플롯, 트리맵, 도넛차트, 버블차트, 히스토그램, 체르노프 페이스, 스타차트, 다차원척도법, 평행좌표계
  - 도식화와 시각화

파이썬으로 표본 추출 (Sampling) 하기

단순랜덤추출법 (simple random sampling)

- 각 샘플에 번호를 부여하여 임의의 n개를 추출하는 방법으로 각 샘플은 선택될 확률이 동일하다.
- 추출한 element를 다시 집어 넣어 추출하면 복원 추출, 다시 집어넣지 않고 추출하면 비복원 추출이다.
- pandas의 DataFrame으로 만든 데이터프레임 객체인 df에 .sample() 함수를 써서 전체 데이터 중 일부 샘플을 랜덤으로 추출할 수 있다.
- df.sample(n=2, replace=False)의 식으로 사용하는데, n은 추출할 샘플의 개수이고 replace는 샘플을 복원해서 뽑을지 여부이다.
- replace=False로 파라미터를 설정하면, 샘플을 뽑을 때마다 뽑은 샘플을 복원하지 않고 연속적으로 샘플을 뽑는다.

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [3]:
```
df = pd.read_csv('/Users/benny/Desktop/datascience/heart.csv')
df_dummy = pd.get_dummies(df)
X = df_dummy.drop('HeartDisease', axis=1)
y = df_dummy['HeartDisease']
from sklearn.model_selection import train_test_split
X_train_clf, X_test_clf, y_train_clf, y_test_clf = train_test_split(X, y, str
print(X_train_clf.shape, X_test_clf.shape)
```

```
(734, 20) (184, 20)
```

In [4]:
```
X_train_sampling = X_train_clf.reset_index(drop=True)
y_train_sampling = y_train_clf.reset_index(drop=True)
```

In [24]:
```
# 샘플 2개를 추출
X_train_sampling.sample(n=2, replace=False) # replace=False 비복원추출
```

Out[24]:

	index	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	Sex_F	Sex_M	ChestP
119	786	69	140	254	0	146	2.0	0	1	
589	385	61	150	0	0	105	0.0	0	1	

2 rows × 21 columns

In [25]:
```
# 전체 데이터의 10%를 추출
X_train_sampling.sample(frac=0.1)
```

Out[25]:

	index	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	Sex_F	Sex_M	ChestP
716	881	44	120	263	0	173	0.0	0	1	
620	741	62	120	267	0	99	1.8	0	1	
286	449	55	0	0	0	155	1.5	0	1	
46	509	58	110	198	0	110	0.0	0	1	
602	367	68	135	0	0	120	0.0	0	1	
...	...	...	...	...	...	...	...	...	...	...
206	473	60	141	316	1	122	1.7	0	1	
468	905	67	152	212	0	150	0.8	0	1	
336	812	54	110	214	0	158	1.6	1	0	
375	543	70	170	192	0	129	3.0	0	1	
339	284	42	140	358	0	170	0.0	0	1	

73 rows × 21 columns

In [26]:
```
# 또한 샘플링을 할 때 파라미터 weights에 특정 열의 값을 입력하면, 해당 열의 값이 크면 클수록 표본
# n과 frac은 둘 중 하나를 선택해서 파라미터로 입력해야 하지만 weights는 두 경우 다 적용 가능하다

X_train_sampling.sample(frac=0.1, weights='Cholesterol')
```

Out[26]:

	index	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	Sex_F	Sex_M	ChestP
267	138	54	140	166	0	118	0.0	0	1	
698	900	58	114	318	0	140	4.4	0	1	
314	537	74	150	258	1	130	4.0	0	1	
146	796	56	134	409	0	150	1.9	1	0	
432	144	56	120	279	0	150	1.0	1	0	

	index	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	Sex_F	Sex_M	ChestP
...	...	...	...	...	...	...	...	...	...	
16	24	40	130	215	0	138	0.0	0	1	
576	145	39	110	273	0	132	0.0	0	1	
708	186	58	130	251	0	110	0.0	0	1	
336	812	54	110	214	0	158	1.6	1	0	
450	225	50	145	264	0	150	0.0	0	1	

73 rows × 21 columns

In [27]:
```
# 일반적으로 한 행이 하나의 데이터 포인트라고 볼 수 있기 때문에 행 방향으로(axis=0) 데이터를 랜덤하
# 하지만 필요 시 열 방향(axis=1)으로도 데이터를 랜덤하게 추출할 수 있다. 그러면 여러가지 열들 중에

X_train_sampling.sample(n=2, axis=1)
```

Out[27]:

	FastingBS	index
0	0	14
1	0	550
2	0	620
3	0	93
4	1	347
...	...	...
729	0	879
730	0	910
731	1	826
732	0	169
733	0	656

734 rows × 2 columns

계통추출법 (systematic sampling)

- 단순랜덤추출법의 변형된 방식으로서 일단, 번호를 부여한 샘플을 나열한다.
- 총 N(30)개의 모집단에서 n(5)개의 샘플을 추출하기 위해서 N/n으로 구간을 나눈다.
- 이 경우 각 구간에 들어있는 샘플의 수는 K(6)가 된다.
- K(6)개의 샘플이 들어 있는 첫 구간에서 임의로 샘플을 하나 선택하고, K(6)개씩 띄어서 각 구간에서 하나씩 샘플을 추출하는 방법이다.

In [17]:
```
def sys_sampling(data, n): # 모집단 데이터프레임과 거기서 추출할 샘플 개수 입력
    N = len(data) # 모집단 관측치 수
    K = N//n # 구간 내 샘플 수
    index = data[:K].sample(1).index # 첫 구간에서 임의로 선택한 샘플 1개의 인덱스
    intoin = index-0 # 샘플 간 간격
    # index개씩 띄어서 각 구간에서 하나씩 샘플을 추출
```

```
    sys_df = pd.DataFrame()
    while len(sys_df) < n:
        sys_df = sys_df.append(data.loc[index, :])
        index += K
    return(sys_df)
```

In [29]:
```
sys_sampling(X_train_sampling, 10)
```

Out[29]:

	index	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	Sex_F	Sex_M	ChestP
4	347	48	115	0	1	128	0.0	0	1	
77	554	53	155	175	1	160	0.3	0	1	
150	485	63	139	217	1	128	1.2	0	1	
223	586	53	124	243	0	122	2.0	0	1	
296	46	37	120	223	0	168	0.0	0	1	
369	29	51	125	188	0	145	0.0	0	1	
442	678	60	150	240	0	171	0.9	1	0	
515	267	34	98	220	0	150	0.0	0	1	
588	68	52	160	246	0	82	4.0	0	1	
661	324	46	100	0	1	133	-2.6	0	1	

10 rows × 21 columns

층화추출법 (stratified random sampling)

- 이질적인 원소들로 구성된 모집단에서 각 계층을 고루 대표할 수 있도록 표본을 추출하는 방법이다.
- 유사한 원소끼리 몇개의 층(stratum)으로 나누어 각 층에서 랜덤 추출하는 방법이다.
- 비례층화추출법과 불비례층화추출법이 있다. 층 내 요소들은 유사하지만, 층과 층의 요소들은 상이하다.
- Scikit Learn의 StratifiedShuffleSplit : 각 층의 비율을 고려해 무작위로 train/test set을 분할하는 인덱스를 반환한다. 비례층화추출법에 해당한다.
- 여기서 파라미터들을 살펴보면 n_splits는 데이터를 섞는(shuffling) 횟수이다.
- test_size는 train set와 test set의 분할 비율을 의미한다.
- 파라미터를 test_size로 해도 되고 train_size로 해도 되는데 둘 중 하나만 적으면 된다.
- 원본 데이터셋의 각 층의 비율과, 층화추출법을 통해 샘플링 된 후 분할된 train/test set의 데이터 층의 비율이 동일한 것을 확인할 수 있다.

In [23]:
```
from sklearn.model_selection import StratifiedShuffleSplit
# n_splits: 다시 섞기 및 분할 반복 횟수
sss = StratifiedShuffleSplit(n_splits=1, test_size=0.2, random_state=1993)

for train_index, test_index in sss.split(X_train_sampling, y_train_sampling):
    X_train, X_test = X_train_sampling.loc[train_index], X_train_sampling.loc
    y_train, y_test = y_train_sampling.loc[train_index], y_train_sampling.loc
```

In [24]:
```
print(y_train_sampling.value_counts()/len(y_train_sampling))
print(y_train.value_counts()/len(y_train))
print(y_test.value_counts()/len(y_test))
```

```
1    0.553134
0    0.446866
Name: HeartDisease, dtype: float64
1    0.553663
0    0.446337
Name: HeartDisease, dtype: float64
1    0.55102
0    0.44898
Name: HeartDisease, dtype: float64
```

In [22]:
```
from sklearn.model_selection import train_test_split
X_train2, X_test2, y_train2, y_test2 = train_test_split(X_train_sampling, y_t
                                                        test_size=0.5, shuffle

print(y_train_sampling.value_counts()/len(y_train_sampling))
print(y_train2.value_counts()/len(y_train2))
print(y_test2.value_counts()/len(y_test2))
```

```
1    0.553134
0    0.446866
Name: HeartDisease, dtype: float64
1    0.553134
0    0.446866
Name: HeartDisease, dtype: float64
1    0.553134
0    0.446866
Name: HeartDisease, dtype: float64
```

집락추출법 (cluster random sampling)

- 군집을 구분하고 군집별로 단순랜덤 추출법을 수행한 후, 모든 자료를 활용하거나 샘플링하는 방법이다.
- 지역표본추출과 다단계표본추출이 이에 해당한다.
- 이 경우, 군집 내 요소들은 상이하지만, 군집과 군집은 비교적 유사한 특성을 띈다.
- 위의 층화추출법에서 사용한 함수들을 사용해서 층 대신 집락 데이터를 적용하면 된다.