강원지역혁신플랫폼

기계학습

Machine Learning

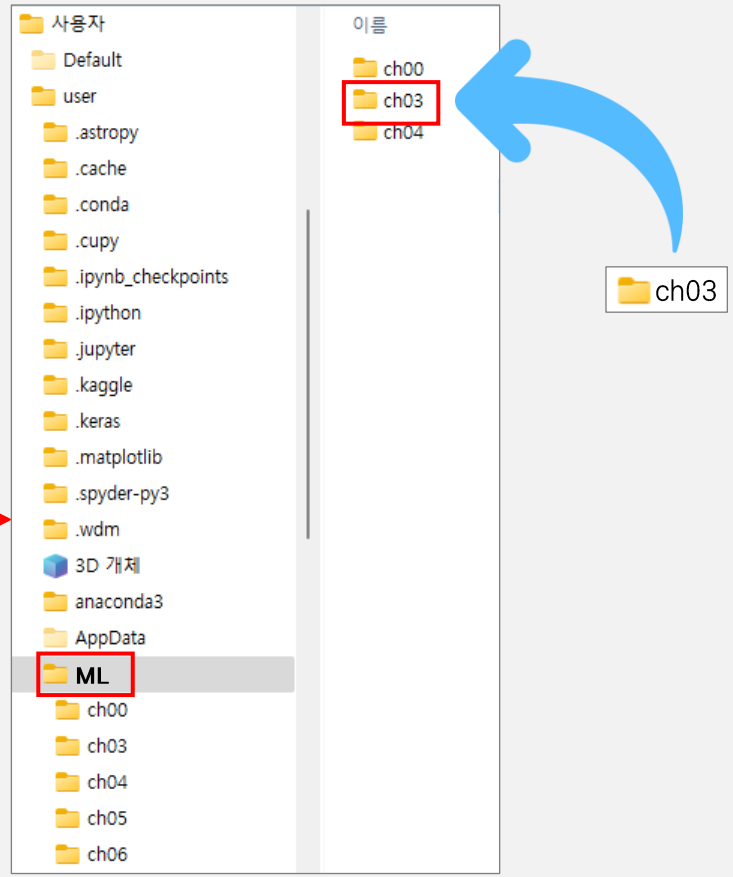
결측값 처리 개념

학습목표

- 결측값 처리 개념을 이해하고 구현할 수 있습니다.

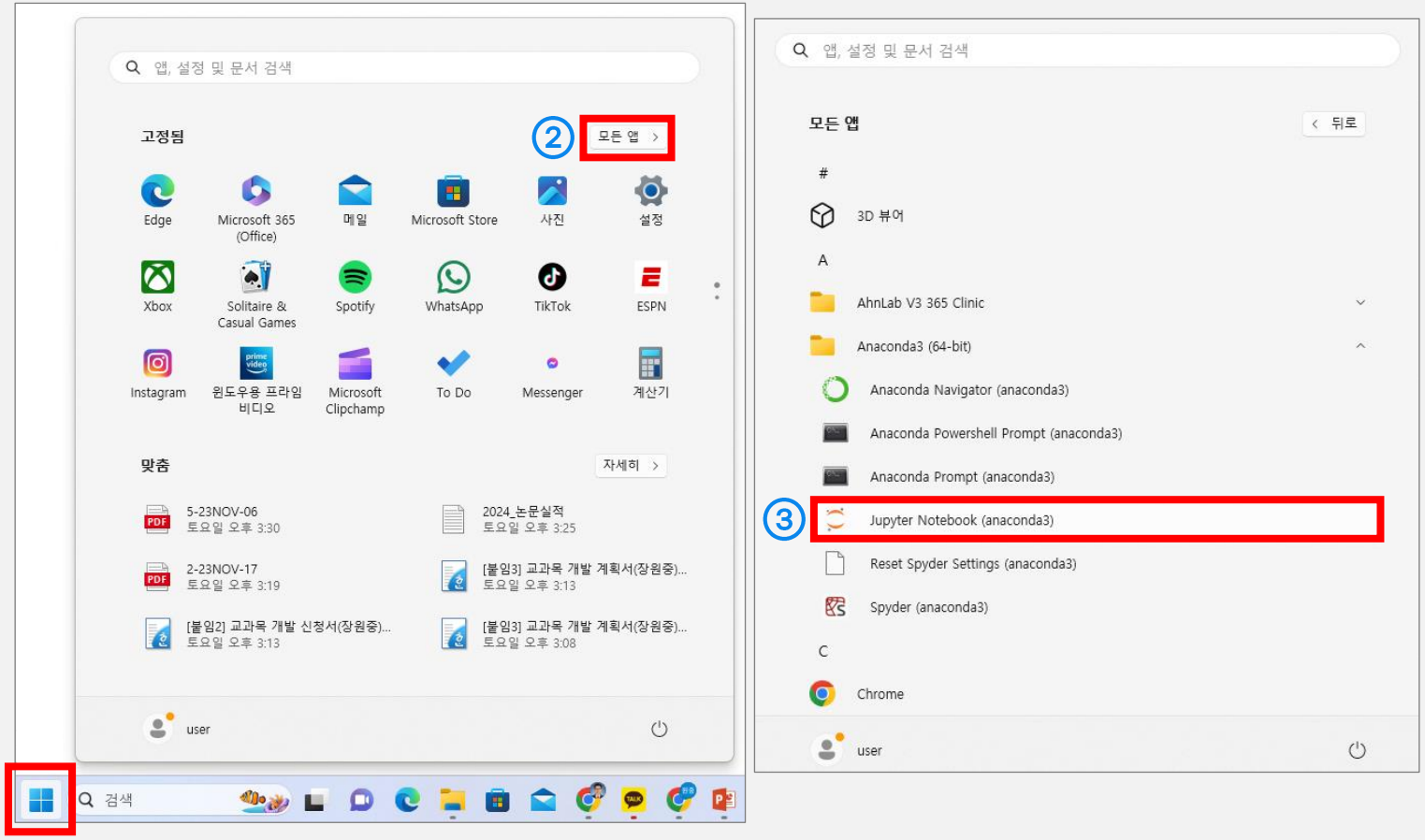
01 | 3주차 실습코드 복사하기

- (권장) 아래와 같은 경로에 실행 소스가 존재하면 환경 구축 완료
  - 3주차 실습코드 다운로드 → 압축해제 → ch03 폴더를 ML 하위 폴더로 복사
  - c:₩Users₩user>ML> 컴퓨터이름 또는 사용자계정



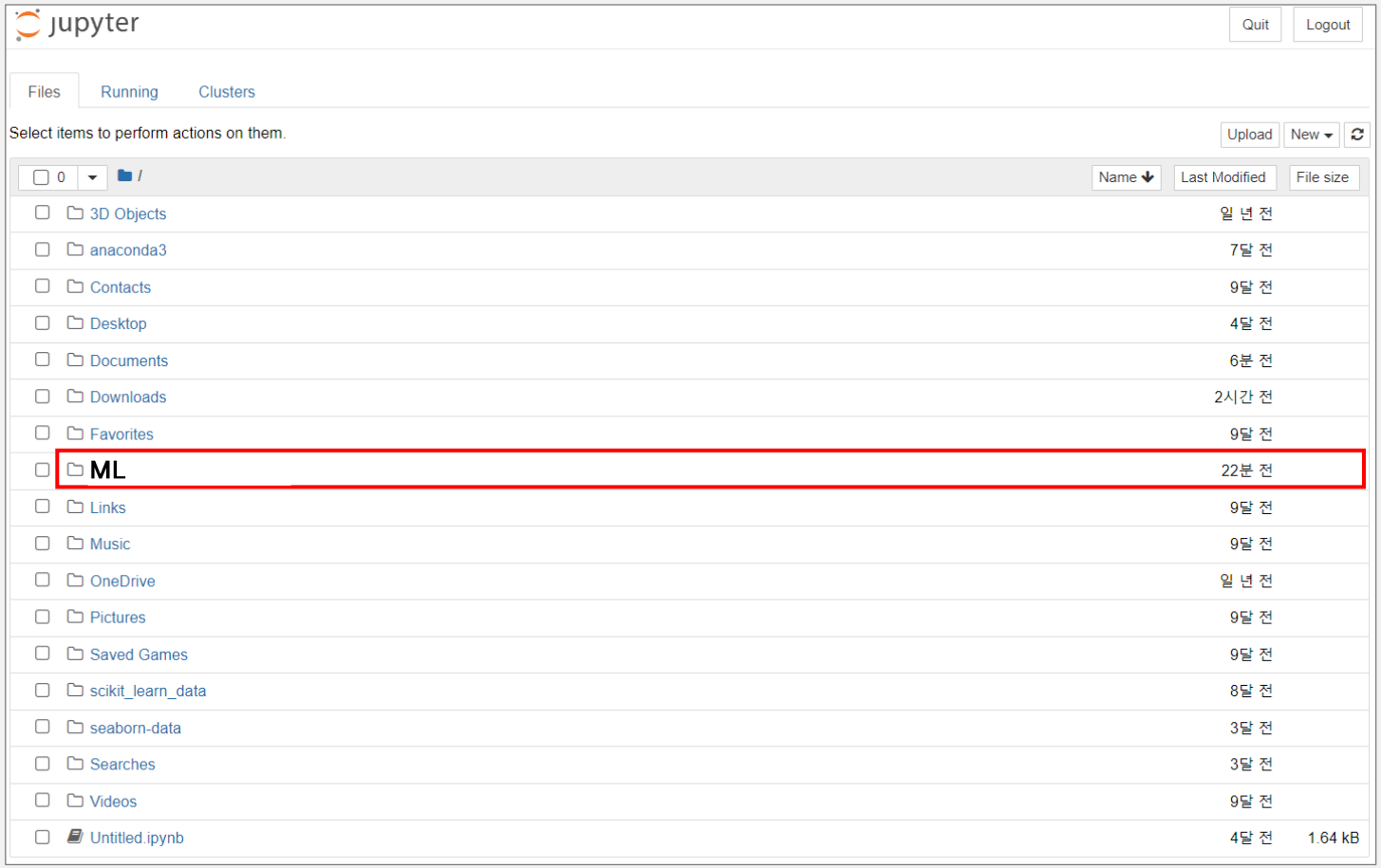
02 | Jupyter Notebook 실행하기

✦ ①시작 메뉴 클릭 > ②모든 앱 버튼 클릭 > ③Anaconda3(64-bit)
> “Jupyter Notebook (anaconda)” 메뉴 클릭하기



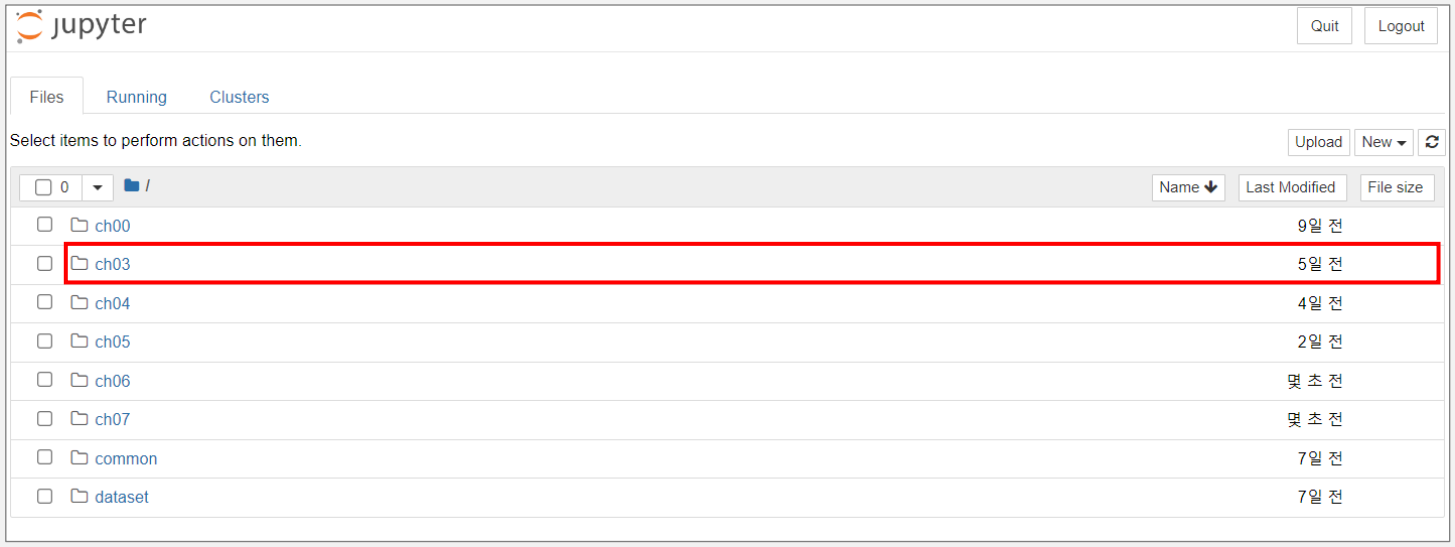
03 | ML 폴더

ML 폴더를 클릭하기



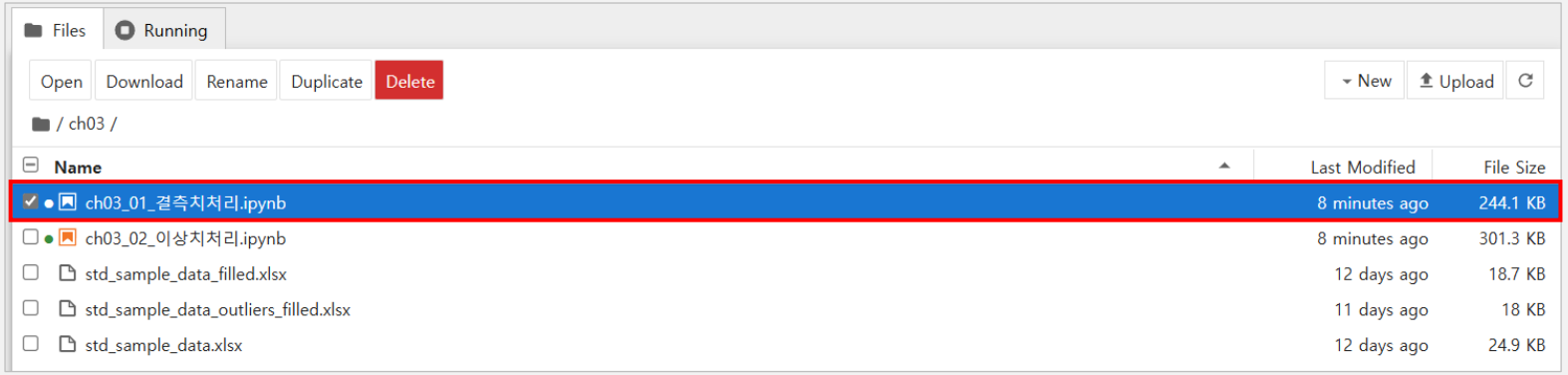
04 | ch03 폴더

ch03 폴더 클릭하기



05 | ch03_01_결측치처리.ipynb

ch03_01_결측치처리.ipynb 파일 클릭하기



06 | 결측값이란?

결측값(Missing Value) 처리

- 결측값(또는 결측치)은 데이터에 값이 없는 것을 뜻함
  - 결측치를 줄여서 NA(Not Available)로 표현하기도 함
  - 다른 언어에서는 Null 이란 표현을 많이 사용함
  - 수의 연산이 불가능 경우는 NaN(Not a Number)으로 처리됨

07 | 결측치 발생 원인

✦ 결측치 발생 원인

- 데이터를 입력하는 도중 **실수로 값**을 **입력하지 못하였**을 때
- 해당 항목에 **적절한 값**이 **없어서 입력되지 못하였**을 때
- **전산 상의 오류**가 발생하여 입력되지 못하였을 때
- 설문조사에서 참가자 중 일부가 **답변하기 어렵거나 곤란한 질문**에 **응답하지 않았**을 때
- 특정 대상을 장기간에 걸쳐서 조사하는 **종단연구**에서 **사망**, **연락두절** 등의 상태가 발생했을 때 등

08 | 결측치의 문제

✦ 결측치는 데이터를 분석하는 데에 있어서 매우 방해가 됨

› 결측치는 다음과 같은 문제를 야기함

- 결측치를 다 제거할 경우, 막대한 데이터 손실을 불러일으킬 수 있음
- 결측치를 잘못 대체할 경우, 데이터에서 편향(bias)이 생길 수 있음
- 결측치를 처리하는 데에 있어 분석가의 견해가 가장 많이 반영됨
  - ❖ 이 때문에 잘못된 경우, 분석결과가 매우 틀어질 수도 있음
- 결측치를 자세히 처리하기 위해서는 시간이 많이 투자되어야 함
  - ❖ 데이터에 기반한 결측치 처리가 진행되어야 분석을 정확하게 진행할 수 있음

09 | 결측값 처리 방법

✦ 결측값 처리 방법에는 여러 가지가 있으며, 각 방법은 상황에 따라 다르게 적용될 수 있음

1. 제거 방법(Deletion Method)
2. 대체 방법(Imputation Method)
3. 모델 기반 방법(Model-Based Method)

10 | 제거 방법: 완전 제거

1 제거 방법(Deletion Method)

› 완전 제거(Listwise Deletion)

- 결측값이 포함된 **행 전체**를 삭제하는 방법
- 데이터 손실이 크지만 **분석이 간단해짐**
- **데이터 표본의 숫자가 적은 경우** 표본의 축소로 인한 **검정력 감소**

	Id	이름	나이	몸무게
Listwise deletion	1	홍길동	27	67
	2	김홍익	34	NA
	3	NA	76	87
	4	이실장	53	65
	5	차도남	NA	71
	6	NA	46	92
	7	최슬기	37	45

완전 제거 방법의 예시

10 | 제거 방법: 부분 제거

1 제거 방법(Deletion Method)

› 부분 제거(Pairwise Deletion)

- 분석에 **필요한 변수**들에서만 **결측값이 있는 행을 제외**하고, 나머지는 사용함
- 데이터 손실을 줄일 수 있음

	Id	이름	나이	몸무게
Pairwise deletion	1	홍길동	27	67
	2	김홍익	34	NA
	3	NA	76	87
	4	이실장	53	65
	5	차도남	NA	71
	6	NA	46	92
	7	최슬기	37	45

부분 제거 방법의 예시

11 | 대체 방법

2 대체 방법(Imputation Method)

› 평균/중앙값/최빈값 대체

- 결측값을 해당 **변수**의 **평균**, **중앙**값 또는 **최빈값**으로 **대체**하는 방법임
- 간단하지만 **분산**이 **감소할 수 있음**

› K-최근접 이웃 대체(K-Nearest Neighbors Imputation)

- **비슷한 값**을 **가진 다른 관측값**을 기반으로 **결측값**을 **대체**하는 방법임
- **데이터의 구조**를 **잘 반영**할 수 있음

11 | 대체 방법

2 대체 방법(Imputation Method)

› 회귀 대체(Regression Imputation)

- 결측값을 **예측**하기 위해 **회귀 모델**을 **사용**함
- **상관관계**를 **잘 반영**할 수 있지만, 모델링 과정이 필요함

› 다중 대체(Multiple Imputation)

- 여러 번의 대체를 통해 **다양한 가능한 값**들로 **결측값**을 **대체**하고, 이를 종합하여 분석함
- **가장 정교한 방법** 중 하나임

12 | 모델 기반 방법

3 모델 기반 방법(Model-Based Method)

› 기대값 최대화 알고리즘(Expectation-Maximization algorithm, EM algorithm)

- 반복적인 방법으로, 결측값을 추정하고 모델을 적합시키는 과정을 반복하여 결측값을 처리함

› 마코프 연쇄 몬테 칼로(Markov Chain Monte Carlo, MCMC) 방법

- 결측값을 다루기 위해 베이시안 접근법을 사용하는 방법
- 통계학 분야에서 전체 데이터 크기에 비해
  결측치의 상대적 빈도가 적은 소규모 데이터의 결측치 처리에 사용됨

13 | 샘플 데이터 집합으로 결측치 처리하기

샘플 데이터 집합으로 결측치 처리하기

대학생 샘플 데이터 집합 읽어옴

› 아래의 표는 대학생 샘플 데이터의 설명임

NO	속성명	속성 설명
1	성명	학생 이름
2	학년	학년(1=1학년, 2=2학년, 3=3학년, 4=4학년)
3	키(cm)	키(cm)
4	몸무게(kg)	몸무게(kg)
5	취미	취미

› 대학생 샘플 데이터 집합은 위의 표와 같이 5개의 속성과 500개의 관측치로 구성됨

13 | 샘플 데이터 집합으로 결측치 처리하기

샘플 데이터 집합으로 결측치 처리하기

- 다음은 **대학생 샘플** 데이터 집합을 읽어오는 코드이다.
  - 대학생 샘플 데이터 집합의 **형상 (500, 5)인 것**을 알 수 있음

```
# 데이터 읽어오기
sample_data = pd.read_excel(os.getcwd()+'/std_sample_data.xlsx')

# 데이터의 형상 # - shape 속성 : 데이터의 (행, 열) 크기를 확인
print(sample_data.shape)   # (500, 5)
```

13 | 샘플 데이터 집합으로 결측치 처리하기

샘플 데이터 집합으로 결측치 처리하기

- 다음은 대학생 샘플 데이터 집합의 전반적인 정보를 확인하는 코드이다.
  - 아래와 같이 데이터 집합의 전반적인 정보를 확인할 수 있음

```
sample_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 5 columns) :
 #   Column    Non-Null Count   Dtype
---  ------    --------------   -----
 0   성명        500 non-null     object
 1   학년        500 non-null     int64
 2   키(cm)      496 non-null     float64
 3   몸무게(kg)  497 non-null     float64
 4   취미        500 non-null     object
Dtypes: float64(2), int64(1), object(2)
Memory usage: 19.7+ KB
```

13 | 샘플 데이터 집합으로 결측치 처리하기

샘플 데이터 집합으로 결측치 처리하기

- 다음은 대학생 샘플 데이터 집합의 **결측치 탐색**을 수행하는 코드이다.
  - 아래 결과와 같이 **2개 속성**(키, 몸무게)에서 **결측치**가 **존재하는 것**을 알 수 있음

```
sample_data.isnull().sum()
```

```
성명          0
학년          0
키(cm)        4
몸무게(kg)    3
취미          0
dtype: int64
```

13 | 샘플 데이터 집합으로 결측치 처리하기

샘플 데이터 집합으로 결측치 처리하기

- 다음은 대학생 샘플 데이터를 출력하는 코드이다.
  - 아래와 같이 숫자가 아닌 속성(성명, 학년, 취미)이 있는 것을 알 수 있음
    - 숫자 속성은 키(cm), 몸무게(kg)인 것을 알 수 있음

sample_data

	성명	학년	키(cm)	몸무게(kg)	취미
0	이서연	3	150.8	62.00000	탁구
1	장지민	3	181.1	84.30000	탁구
2	윤하윤	1	161.3	70.60000	골프
3	임수현	1	158.6	79.80000	달리기
4	정수현	3	182.9	80.40000	골프
...	...	...	...	...	...
495	장민수	1	184.5	59.20000	등산
496	최하윤	4	179.5	57.00000	축구
497	윤현우	2	188.7	63.80000	테니스
498	임지민	4	158.9	115.36752	탁구
499	김예은	3	179.9	84.60000	달리기

500 rows x 5 columns

13 | 샘플 데이터 집합으로 결측치 처리하기

샘플 데이터 집합으로 결측치 처리하기

- 다음은 대학생 샘플 데이터 집합에서 **결측치가 포함된 행만 필터링**하는 코드이다.
  - 아래 결과와 같이 **7개 행**에서 **결측치가 존재하는 것**을 알 수 있음

```
# 결측치가 포함된 행만 필터링
missing_data_rows = sample_data[sample_data.isnull().any(axis=1)]

# 결측치가 포함된 행 출력
print(missing_data_rows)
#       성명  학년 키(cm)  몸무게(kg)  취미
#107  조지우   3   NaN     63.4      골프
#141  김지우   3   185.6   NaN       달리기
#192  이준호   3   178.6   NaN       달리기
#365  최지우   2   NaN     57.7      등산
#395  정하윤   4   NaN     68.0      골프
#450  김예은   3   NaN     56.3      수영
#454  장수현   3   181.0   NaN       등산
```

13 | 샘플 데이터 집합으로 결측치 처리하기

샘플 데이터 집합으로 결측치 처리하기

- 다음은 대학생 샘플 데이터 집합에서 **결측치**를 **중앙값으로 대체**하는 코드이다.
  - 아래 결과와 같이 **결측치**가 **중앙값**으로 **대체된 것**을 알 수 있음

```
# 결측치를 중앙값으로 대체
sample_data_filled = sample_data.fillna(sample_data.median(numeric_only=True))

# 결측치가 포함된 행만 필터링
missing_data_rows = sample_data_filled[sample_data_filled.isnull().any(axis=1)]

# 결측치가 포함된 행 출력
print(missing_data_rows)
```

```
Empty DataFrame
Columns: [성명, 학년, 키(cm), 몸무게(kg), 취미]
Index: [ ]
```

13 | 샘플 데이터 집합으로 결측치 처리하기

샘플 데이터 집합으로 결측치 처리하기

- 다음은 대학생 샘플 데이터 집합에서 **결측치를 중앙값**으로 **대체한 데이터**를 **새로운 엑셀 파일**로 **저장**하는 코드이다.
  - 아래 결과와 같이 **새로운 엑셀 파일로 저장된 것**을 알 수 있음

```
# 중앙값으로 대체한 데이터를 새로운 엑셀 파일로 저장
filled_file_path = os.getcwd()+'/std_sample_data_filled.xlsx'
sample_data_filled.to_excel(filled_file_path, index=False)

print(f"Filled data saved to {filled_file_path}")
```

```
Filled data saved to D:\Pywork\CKU202401SA\ch06/std_sample_data_filled.xlsx
```

13 | 샘플 데이터 집합으로 결측치 처리하기

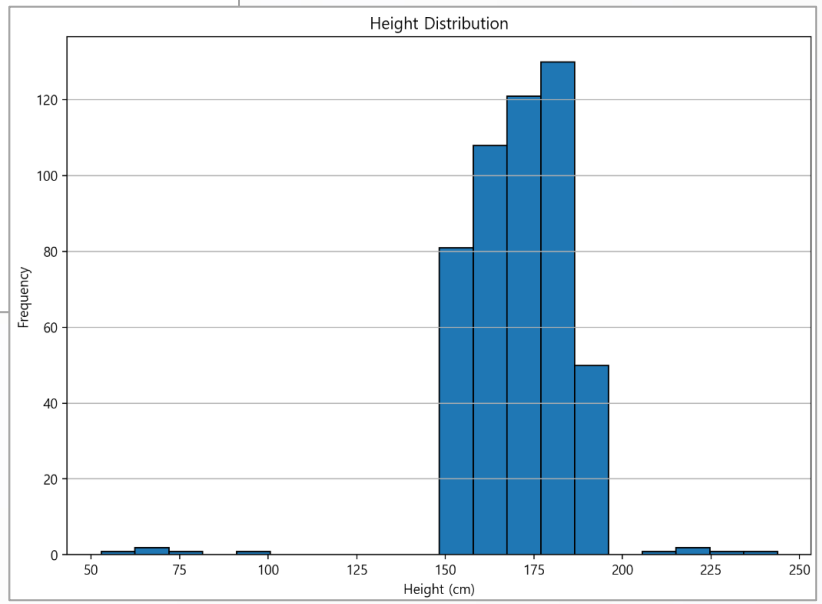
샘플 데이터 집합으로 결측치 처리하기

✦ 다음은 대학생 샘플 데이터 집합에서 키 속성으로 히스토그램을 그리는 코드이다.

› 아래 결과와 키가 아주 작은 사람과 아주 큰 사람이 존재하는 것을 알 수 있음

```
# 히스토그램 그리기
plt.figure(figsize=(10, 7))
plt.hist(sample_data_filled['키(cm)'], bins=20, edgecolor='black')
plt.title('Height Distribution')
plt.xlabel('Height (cm)')
plt.ylabel('Frequency')
plt.grid(axis='y')

# 그래프 보여주기
plt.show()
```



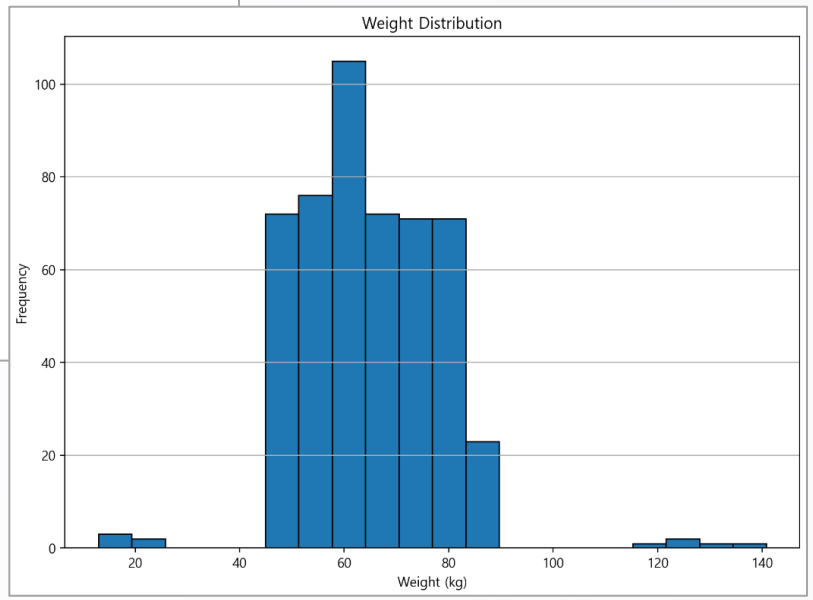
13 | 샘플 데이터 집합으로 결측치 처리하기

샘플 데이터 집합으로 결측치 처리하기

- 다음은 대학생 샘플 데이터 집합에서 몸무게 속성으로 히스토그램을 그리는 코드이다.
  - 아래 결과와 몸무게가 아주 작은 사람과 아주 큰 사람이 존재하는 것을 알 수 있음

```
# 히스토그램 그리기
plt.figure(figsize=(10, 7))
plt.hist(sample_data_filled['몸무게(kg)'], bins=20,
edgecolor='black')
plt.title('Weight Distribution')
plt.xlabel('Weight (kg)')
plt.ylabel('Frequency')
plt.grid(axis='y')

# 그래프 보여주기
plt.show()
```



13 | 샘플 데이터 집합으로 결측치 처리하기

샘플 데이터 집합으로 결측치 처리하기

- 다음은 대학생 샘플 데이터 집합에서 **학년별 평균 키와 몸무게**를 계산하는 코드이다.
  - 아래 결과와 같이 **학년별 평균 키와 몸무게**가 **큰 차이 없는 것**을 알 수 있음

```
# 학년별 평균 키(cm)와 몸무게(kg) 계산
grade_means = sample_data_filled.groupby('학년')[['키(cm)', '몸무게(kg)']].mean()

# 결과 출력
print("Grade-wise Mean Height (cm) and Weight (kg):")
print(grade_means)
```

```
Grade-wise Mean Height (cm) and Weight (kg) :
          키(cm)      몸무게(kg)
학년
1     169.715654   64.896831
2     170.005210   63.009466
3     169.876298   65.329955
4     173.799325   65.195040
```

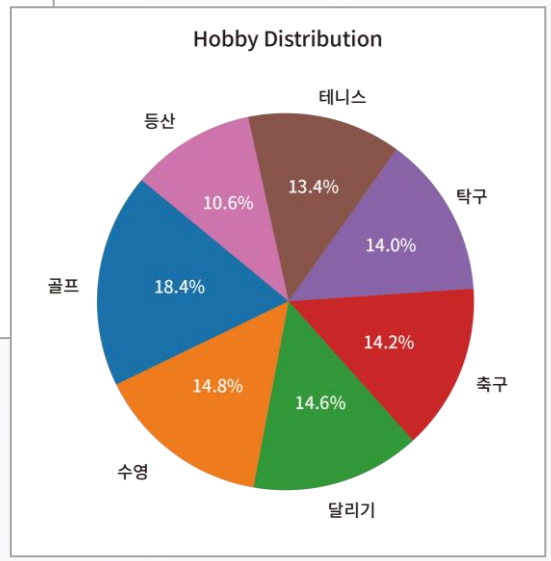
샘플 데이터 집합으로 결측치 처리하기

- 다음은 대학생 샘플 데이터 집합에서 **취미 분포**를 **알 수 있도록 파이차트**를 그리는 코드이다.
  - 아래 결과와 같이 학생들의 **취미**가 **골고루 분포된 것**을 알 수 있음

```
# 취미 분포 계산
hobby_distribution = sample_data_filled['취미'].value_counts()

# 파이차트 그리기
plt.figure(figsize=(10, 7))
plt.pie(hobby_distribution, labels=hobby_distribution.index, autopct='%1.1f%%', startangle=140)
plt.title('Hobby Distribution')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.

# 그래프 보여주기
plt.show()
```



14 | 회귀 대체로 결측치 처리하기

회귀 대체(Regression Imputation)로 결측치 처리하기

- 아래의 데이터를 이용해 결측치를 예측하여 대치해 보자.
  - 결측치를 대치할 회귀식도 구해보자.

Y_1	Y_2	Y_3	Y_3(변경 후)
10	15	20	20
12	20	30	30
14	25	40	40
25	35	05	50
30	45	55	55
34	45	60	60
37	49	70	70
40	55	?	?
44	60	?	?
49	68	?	?

14 | 회귀 대체로 결측치 처리하기

회귀 대체(Regression Imputation)로 결측치 처리하기

- 아래와 같이 회귀식 구할 수 있음
  - 회귀식을 이용해 결측치를 아래와 같이 예측됨

절편 계수

$$Y_{3i} = \beta_0 + \beta_1 Y_{1i} + \beta_2 Y_{2i} + \epsilon_i, i = 1, \ldots, 7$$

$$Y_3 = 5.11 + 0.21 Y_{1i} + 1.09 Y_{2i}$$

$$\beta_0 = 5.11, \beta_1 = 0.21, \beta_2 = 1.09$$

$$?_1 = 5.11 + 0.21 \times 40 + 1.09 \times 55 = 73.49$$

$$?_2 = 5.11 + 0.21 \times 44 + 1.09 \times 60 = 79.78$$

$$?_2 = 5.11 + 0.21 \times 44 + 1.09 \times 68 = 89.56$$

14 | 회귀 대체로 결측치 처리하기

회귀 대체(Regression Imputation)로 결측치 처리하기

✦ 다음은 회귀식을 이용해 결측치를 예측하는 코드이다.

› 데이터 생성하기

```
# Create the dataframe
data = { "Y1": [10, 12, 14, 25, 30, 34, 37, 40, 44, 49],
         "Y2": [15, 20, 25, 35, 45, 45, 49, 55, 60, 68],
         "Y3": [20, 30, 40, 50, 55, 60, 70, np.nan, np.nan, np.nan] }
df = pd.DataFrame(data)
```

› 독립변수, 종속변수 생성하기

```
# Perform regression imputation for Y3 using Y1 and Y2
# Drop rows where Y3 is not NaN to train the model
train_data = df.dropna()

# Features and target
X_train = train_data[["Y1", "Y2"]]
y_train = train_data["Y3"]
```

14 | 회귀 대체로 결측치 처리하기

회귀 대체(Regression Imputation)로 결측치 처리하기

회귀 모델 생성 및 학습 하기

```
# Train the linear regression model
model = LinearRegression()
model.fit(X_train, y_train)
```

$$Y_{3i} = \beta_0 + \beta_1 Y_{1i} + \beta_2 Y_{2i} + \epsilon_i \, , i = 1, \ldots, 7$$

$$Y_3 = 5.11 + 0.21 Y_{1i} + 1.09 Y_{2i}$$

$$\beta_0 = 5.11, \beta_1 = 0.21, \beta_2 = 1.09$$

Y_1	Y_2	Y_3	Y_3(변경 후)
10	15	20	20
12	20	30	30
14	25	40	40
25	35	05	50
30	45	55	55
34	45	60	60
37	49	70	70
40	55	?	?
44	60	?	?
49	68	?	?

14 | 회귀 대체로 결측치 처리하기

회귀 대체(Regression Imputation)로 결측치 처리하기

예측값으로 결측값 대체하기

```
# Predict the missing values in Y3
missing_data = df[df["Y3"].isna()]
X_missing = missing_data[["Y1", "Y2"]]
df.loc[df["Y3"].isna(), "Y3"] = model.predict(X_missing)

# Print the imputed dataframe
print(df)
```

	Y1	Y2	Y3
0	10	15	20.000000
1	12	20	30.000000
2	14	25	40.000000
3	25	35	50.000000
4	30	45	55.000000
5	34	45	60.000000
6	37	49	70.000000
7	40	55	73.488741
8	44	60	79.779978
9	49	68	89.558868

회귀식으로 결측값을 예측함

$$?_1 = 5.11 + 0.21 \times 40 + 1.09 \times 55 = 73.49$$

$$?_2 = 5.11 + 0.21 \times 44 + 1.09 \times 60 = 79.78$$

$$?_2 = 5.11 + 0.21 \times 44 + 1.09 \times 68 = 89.56$$

14 | 회귀 대체로 결측치 처리하기

회귀 대체(Regression Imputation)로 결측치 처리하기

회귀 계수(절편과 계수) 출력하기

```
# Extract the coefficients and intercept from the trained model
coefficients = model.coef_
intercept = model.intercept_

# Print the regression equation
regression_equation = f"Y3 = {intercept:.2f} + ({coefficients[0]:.2f} * Y1) +
({coefficients[1]:.2f} * Y2)“
print("Regression Equation:", regression_equation)
```

```
Regression Equation: Y3 = 5.11 + (0.21 * Y1) + (1.09 * Y2)
```

절편 계수 계수