<[파이썬 데이터 탐험의 시작] 제3장: 데이터의 숨은 그림 찾기, 히스토그램과 도수분포!>

선생님: (여러 사람의 키가 뒤죽박죽 적힌 종이를 보여주며) 얘들아, 안녕! 우리 반 학생 20명의 몸무게를 이렇게 쭉 적어놓기만 하면, 어떤 몸무게의 친구들이 가장 많은지, 데이터가 어떻게 퍼져있는지 한눈에 알기 어렵겠지?

코딩새싹 🌱: 네, 너무 어지러워요! 하나하나 다 읽어봐야 하잖아요!

선생님: 맞아! 그래서 데이터 탐정들은 데이터를 구간별로 나눠서 각 구간에 몇 개씩 데이터가 있는지 세어보는 **도수분포표**를 만들고, 그걸 막대그래프 모양으로 그린 **히스토그램**을 사용한단다. 데이터의 숨겨진 모양과 패턴을 찾아내는 마법 같은 그림이지!

<1. 똑똑한 자동 분류기, `np.histogram()`!>

선생님: 지난번에 우리가 `if-elif-else`를 길게 써서 도수분포표를 만들었지? 넘파이에는 그 모든 과정을 한 방에 처리해주는 `np.histogram()`이라는 아주 똑똑한 자동 분류기가 있단다!

```
import numpy as np
import pandas as pd

# 20명의 무작위 몸무게 데이터 생성 (20~54kg 사이)
weights = np.random.randint(20, 55, 20)
print("--- 무작위 몸무게 데이터 (20명) ---")
print(weights)

# 데이터를 나눌 '구간(bins)'을 정해주자! 5kg 단위로!
bins = np.arange(20, 56, 5) # 20, 25, 30, 35, 40, 45, 50, 55 (마지막 숫자 포함되게 56으로)
print("\n데이터를 나눌 구간:", bins)

# np.histogram 자동 분류기 작동!
hist, bins_result = np.histogram(weights, bins) # 데이터와 구간을 넣어주면,
print("\n각 구간에 속한 데이터의 개수 (도수):", hist) # 각 구간의 데이터 개수(hist)와
print("실제 사용된 구간 정보:", bins_result) # 실제 사용된 구간(bins_result)을 알려줘!
```

궁금이 💡: 와! `np.histogram()` 함수 하나에 데이터랑 구간만 넣어주니까, `for`문이랑 `if`문 없이도 각 구간에 몇 명이 있는지 바로 알려주네요! `hist`가 바로 도수(frequency)인 거죠?

선생님: 정확해! 이 `hist`를 가지고 우리는 더 다양한 정보를 얻을 수 있단다.

<2. 전체 중 얼마나? 상대도수와 누적도수>

선생님: "어떤 구간에 3명이 있다"는 정보도 좋지만, "전체의 15%가 이 구간에 있다"고 하면 비교하기가 더 쉽겠지? 이게 바로 **상대도수**야!

```
# 상대도수 = 각 구간의 도수 / 전체 데이터 개수
total_number = len(weights) # 전체 학생 수
```

```
relative_freq = hist / total_number
print("\n--- 각 구간의 상대도수 ---")
print(relative_freq)
```

선생님: 그럼 "35kg 미만인 학생은 총 몇 명이지?" 처럼, 특정 구간까지의 도수를 차곡차곡 더해서 보는 건 **누적도수**라고 한단다.

```
# 누적도수 구하기 (저금통 패턴 기억나지?)
accum_weight = []
previous_sum = 0 # 누적 합계를 저장할 변수

for count in hist: # 각 구간의 도수를 하나씩 꺼내서
  previous_sum += count # 이전 합계에 더하고,
  accum_weight.append(previous_sum) # 누적된 값을 리스트에 추가!

print("\n--- 각 구간까지의 누적도수 ---")
print(accum_weight)
```

코딩새싹 🌱: `hist` 배열을 `for`문으로 돌면서 더하니까 누적된 학생 수가 착착 계산되네요!

<3. 데이터로 그림 그리기! `plt.hist()` 와 `plt.bar()`>

선생님: 자, 이제 이 도수분포표를 Matplotlib 화가에게 넘겨서 멋진 히스토그램으로 그려달라고 해보자!

방법 1: 가장 쉬운 방법, `plt.hist()`

`plt.hist()`는 원본 데이터와 구간만 주면 알아서 도수를 계산하고 그래프까지 그려주는 만능 마법사야!

```
import matplotlib.pyplot as plt

plt.figure(figsize=(8, 6)) # 도화지 크기 설정
# weights 데이터를 bins 구간에 맞춰 히스토그램으로 그려줘!
plt.hist(weights, bins, rwidth=0.8, color="green", alpha=0.5, edgecolor="black")
plt.title("Histogram of Weights", fontsize=16)
plt.xlabel("Weight (kg)", fontsize=12) # x축 이름
plt.xticks(bins, fontsize=12) # x축 눈금을 우리 구간에 맞춰서!
plt.yticks(fontsize=12)
plt.grid(axis='y', alpha=0.5) # y축 방향으로만 격자 표시
plt.show()
```

궁금이 💡: `rwidth=0.8`은 막대의 너비를 살짝 줄여서 보기 좋게 하는 건가요? `edgecolor`는 막대 테두리 색이고요! 정말 `plt.hist()` 하나로 다 되네요!

방법 2: 원리를 이해하는 방법, `plt.bar()`

히스토그램의 원리를 더 깊이 이해하기 위해, `plt.bar()`(막대그래프)로 직접 그려볼 수도 있어. 막대그래프는 "어디에(x 위치), 얼마나 높은(height 높이)" 막대를 그릴지 알려줘야 해.

```
# 높이(height)는 우리가 계산한 도수(hist)를 쓰면 되지.
# 그럼 x 위치는? 각 구간의 '가운데' 값(계급값)을 쓰는 게 좋겠지?
bin_centers = (bins[:-1] + bins[1:]) / 2 # (구간시작+구간끝)/2 를 모든 구간에 대해 한방에!
print("\n각 구간의 가운데 값 (계급값):", bin_centers)

plt.figure(figsize=(8, 6))
plt.bar(bin_centers, hist, width=4, color='skyblue', edgecolor='darkblue') # x위치, 높이, 너비를 직접 지
plt.title("Histogram with plt.bar()", fontsize=16)
plt.xticks(bins)
plt.show()
```

코딩새싹 🌱: plt.hist()가 훨씬 편하긴 한데, plt.bar()로 직접 그려보니까 히스토그램이 '각 구간의 가운데 값에 도수만큼의 높이를 가진 막대'라는 게 확실히 이해돼요!

<4. 히스토그램으로 세상 분석하기!>

선생님: 이 히스토그램(도수분포표)만 있으면, 원래 데이터가 없어도 많은 것을 추측하고 계산할 수 있어!

[미션 1] 도수분포표로 그룹의 평균 몸무게 추정하기!

"우리는 각 구간에 몇 명이 있는지는 알지만, 그 학생들의 정확한 몸무게는 몰라. 그래서 '이 구간에 있는 학생들은 모두 구간의 가운데 값(계급값)의 몸무게를 가질 거야!'라고 가정하고 평균을 계산하는 거야."

```
# 평균 ≈ (계급값 * 도수)의 총합 / 전체 도수
estimated_total_weight = sum(hist * bin_centers) # 넘파이의 강력한 벡터 연산!
estimated_average = estimated_total_weight / len(weights)

print(f"\n도수분포표로 추정한 평균 몸무게: {estimated_average:.2f} kg")
print(f"실제 데이터의 평균 몸무게: {weights.mean():.2f} kg") # 실제 평균과 비교!
```

궁금이 💡: 와! 추정한 평균이랑 실제 평균이 꽤 비슷하네요! hist * bin_centers 이렇게 넘파이 배열끼리 곱하니까 각 구간의 (계급값*도수)가 한 번에 계산됐어요!

[미션 2] "몸무게가 50kg 이상인 학생은 전체의 몇 %인가?"

```
# bins: [20 25 30 35 40 45 50 55]
# hist: [ n1 n2 n3 n4 n5 n6 n7]
# 50kg 이상인 구간은 '50~55' 구간 하나뿐. 이 구간은 hist에서 인덱스 6에 해당.
# (만약 45kg 이상이라면 hist[5:] 처럼 슬라이싱을 썼겠지?)
# 더 넓은 범위의 예시로 수정: 40kg 이상인 학생은 몇 %인가?
# 40~45, 45~50, 50~55 구간에 해당. hist의 인덱스 4, 5, 6
over_40_count = sum(hist[4:])
percentage = (over_40_count / len(weights)) * 100
print(f"\n몸무게가 40kg 이상인 학생의 비율: {percentage:.1f} %")
```

선생님: (깔끔하게 정리된 히스토그램을 보여주며) 자, 오늘 우리는 뒤죽박죽이던 데이터 속에서 np.his-

togram으로 질서를 찾고, plt.hist로 숨겨진 그림을 그려냈어! 그리고 그 그림을 통해 데이터의 평균을 추정하고, 특정 그룹의 비율까지 알아냈지. 이것이 바로 데이터 분석의 첫걸음이자 가장 중요한 과정이란다! 오늘 탐험도 정말 훌륭했어, 나의 꼬마 데이터 탐정들! 🕵☒♀☒🎉