Ananya Dahal

2408840

Week-1

1) Case-1: Healthcare Policy Evaluation.

Task:
Which method would provide more actionable insights? What key variables would you prioritize?

→ Selection Bias.

→ Observational as researchers are not manipulating variables. They just collect & analyze existing information or opinions.

→ Survey better as more actionable insights than just secondary sources.

→ key variables :- Accessibility, Awareness, trust, status (socioeconomic)

2) Case-2: Corporate Diversity Audit.

Task:
How would you balance data completeness with legal / ethical constraints?

→ Secondary source (Observational) (database)

→ ~~B~~ Automatic bias.

→ Leagal restrictions prevent new data collection in some cant record cover all 20 offices over 5 yrs, ensuring completeness.

limitation :-
missing demographic data in restricted countries.

× Balancing completeness & ethics :-
use existing HR data where legally allowed.
Anonymize & aggregate data.
Report gaps transparently & avoid inferring gender where not permitted.

3) Case-3 :- Urban Planning Challenge.

Task:-
What blended approach could validate finding while minimizing bias?

→ Observational.
→ Secondary source. as :-
objective usage data captures real behavior at scale.
enables before & after trend & time-of-day/route.
→ Blended Approach to validate & minimize bias:-
use smart card data for effect size, add small purposive focus groups to interpret barriers.
→ key limitations :-
card data may miss shared / swiped exceptions
qualitative inputs not generalizable.

4)
→
→

4) Case-4:- Education Reform.

→ Observational.
→ Method:- Secondary source : std record + provincial labor stats
reasons :- ten-year coverage enables robust trend & cohort analyses.
labor stats are official, comparable; surveys have low response bias.
→ Addressing Temporal Mismatch (education vs labor lag):
Align cohorts to labor entry, use nowcasting / forecasting for jobs. Compare moving averages & report sensitivity ranges. Segment by STEM subfields reduce aggregation bias.
→ limitations:-
Forecast / nowcast uncertainty; proxies for intent may not capture true preferences.

5) Case-5:- GitHub Productivity Study.
→ Observational
→ Secondary (commit logs + project metadata)
public commits are objective, timestamped, and scalable across teams. metadata enables normalization & controls.
→ Blended Approach to Validate & Minimize Bias:-
Compute multiple productivity proxies (commits, [illegible]

normalize by repo size, activity windows &
team composition.

→ Limitation
Commits ≠ true productivity.

HCAI5DS02_AnanyaDahal_2408840_Week-1.ipynb

File Edit View Insert Runtime Tools Help

Commands + Code + Text Run all

Share Gemini

RAM Disk

Ananya Dahal

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]

```python
# 1. Sports Performance (Heights)
heights = np.array([5.1, 4.8, 5.0, 4.9, 5.3, 4.7, 4.9, 4.8, 5.2, 5.0])
mean_height = np.mean(heights)
sd_height = np.std(heights, ddof=1)
cv_height = (sd_height / mean_height) * 100

print(f"Sports Heights\nMean: {mean_height:.2f} ft, SD: {sd_height:.2f} ft, CV: {cv_height:.2f}%")
```

```
Sports Heights
Mean: 4.97 ft, SD: 0.19 ft, CV: 3.80%
```

[16]

```python
# 2. Impact of Outlier (3.8 ft)
outlier_heights = np.append(heights, 3.8)
mean_outlier = np.mean(outlier_heights)
sd_outlier = np.std(outlier_heights, ddof=1)
cv_outlier = (sd_outlier / mean_outlier) * 100

print(f"\nWith Outlier\nMean: {mean_outlier:.2f} ft, SD: {sd_outlier:.2f} ft, CV: {cv_outlier:.2f}%")
```

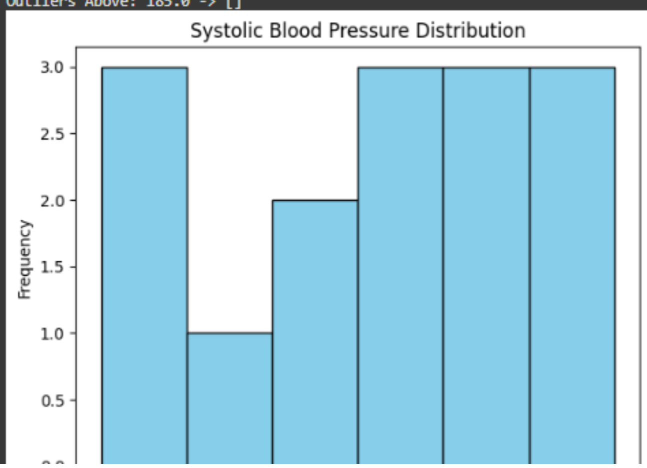
```

With Outlier
Mean: 4.86 ft, SD: 0.40 ft, CV: 8.14%
```

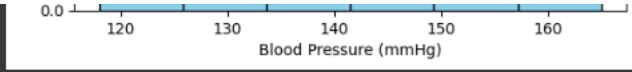
[4]

```python
# 2. Blood Pressure Readings
bp = np.array([118, 122, 125, 130, 135, 138, 142, 144, 146, 150, 152, 155, 160, 162, 165])
mean_bp = np.mean(bp)
median_bp = np.median(bp)
sd_bp = np.std(bp, ddof=1)
q1_bp = np.percentile(bp, 25)
q3_bp = np.percentile(bp, 75)
iqr_bp = q3_bp - q1_bp
upper_fence = q3_bp + 1.5 * iqr_bp
outliers_bp = bp[bp > upper_fence]
```

```python
# 2. Blood Pressure Readings
bp = np.array([118, 122, 125, 130, 135, 138, 142, 144, 146, 150, 152, 155, 160, 162, 165])
mean_bp = np.mean(bp)
median_bp = np.median(bp)
sd_bp = np.std(bp, ddof=1)
q1_bp = np.percentile(bp, 25)
q3_bp = np.percentile(bp, 75)
iqr_bp = q3_bp - q1_bp
upper_fence = q3_bp + 1.5 * iqr_bp
outliers_bp = bp[bp > upper_fence]

print(f"Blood Pressure\nMean: {mean_bp:.2f} mmHg, Median: {median_bp:.2f} mmHg, SD: {sd_bp:.2f} mmHg, IQR: {iqr_bp}")
print(f"Outliers Above: {upper_fence} -> {outliers_bp}")

plt.hist(bp, bins=6, color='skyblue', edgecolor='black')
plt.title('Systolic Blood Pressure Distribution')
plt.xlabel('Blood Pressure (mmHg)')
plt.ylabel('Frequency')
plt.show()
```

```
Blood Pressure
Mean: 142.93 mmHg, Median: 144.00 mmHg, SD: 14.80 mmHg, IQR: 21.0
Outliers Above: 185.0 -> []
```





Systolic Blood Pressure Analysis

The mean and median are very close, indicating a fairly symmetric distribution.

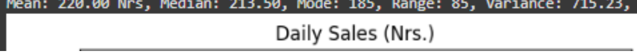
The close values of mean (142.93) and median (144) suggest minimal skewness. From the data spread (ranging from 118 to 165), the distribution likely appears slightly right-skewed, but overall, reasonably symmetric.

No patients had systolic BP above 185 mmHg, so no high outliers were detected.

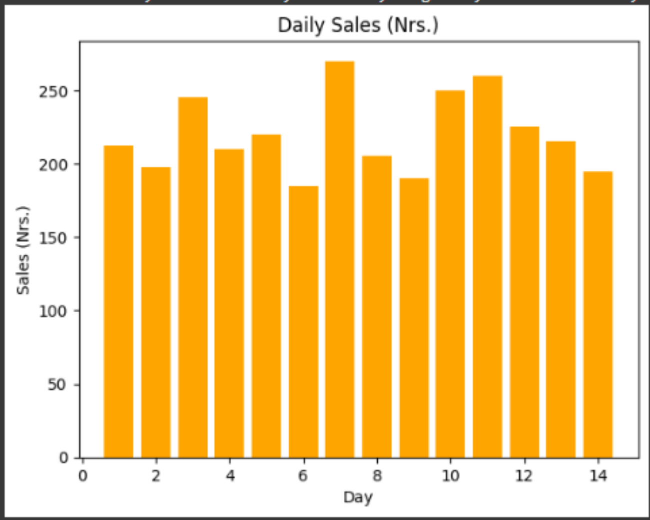
The mean of 143 mmHg is slightly above the ideal systolic BP (120 mmHg). The absence of outliers indicates the group is consistent, but the overall BP levels suggest many patients may have pre-hypertension or mild hypertension. The spread (SD of 148 mmHg) is moderate, indicating typical biological variability. Based on the shape and spread, the group appears relatively homogeneous, but rarely have elevated BP compared to healthy guidelines.

```
# 3. Retail Sales
sales = np.array([212, 198, 245, 210, 220, 185, 270, 205, 190, 250, 260, 225, 215, 195])

mean_sales = np.mean(sales)
median_sales = np.median(sales)
mode_sales = pd.Series(sales).mode()[0]
range_sales = np.ptp(sales)
variance_sales = np.var(sales, ddof=1)
sd_sales = np.std(sales, ddof=1)

print(f"Retail Sales\nMean: {mean_sales:.2f} Nrs, Median: {median_sales:.2f}, Mode: {mode_sales}, "
      f"Range: {range_sales}, Variance: {variance_sales:.2f}, SD: {sd_sales:.2f}")

plt.bar(range(1, 15), sales, color='orange')
plt.title('Daily Sales (Nrs.)')
plt.xlabel('Day')
plt.ylabel('Sales (Nrs.)')
plt.show()
```

```
Retail Sales
Mean: 220.00 Nrs, Median: 213.50, Mode: 185, Range: 85, Variance: 715.23, SD: 26.74
```



Retail Sales
Mean: 220.00 Nrs, Median: 213.50, Mode: 185, Range: 85, Variance: 715.23, SD: 26.74



Sales Data

The mean and median are close, suggesting minimal skewness, but the presence of higher values like 6270, 6260 inflates the mean slightly.

Visual inspection should reveal Highest sales = 1270, Lowest sales = 1185. Annotating these points makes high/lows clear for decision-makers.

Range (R85) and SD (26.88) indicate moderate variability. The data suggests sales fluctuate but aren't wildly inconsistent. The distribution shows daily ups and downs typical of retail, but no extreme volatility.

If Sunday sales are consistently 20% lower, those days pull down the overall mean, potentially making sales appear weaker than reality for other sizes. It introduces systematic variation. Average calculations may underrepresent weekday performance. Performance reports should segment Sunday separators to avoid misleading conclusions. Adjusted Approach: Consider calculating separate means for weekdays and Sundays or use median for reporting to reduce Sunday distortion.

[9]
```python
#dropout risk assesment:
drop_cv = (0.6/2.1)*100
retain_cv = (0.5/3.1*100)

print(f"Dropout Risk \n Dropout CV:{drop_cv:.2f}%, Retained CV: {retain_cv:.2f}%")
```

```
Dropout Risk
 Dropout CV:28.57%, Retained CV: 16.13%
```

CV measures relative variability. Higher CV = more spread compared to the mean.

The Dropout group has a significantly higher CV (28.57% vs 16.13%), meaning their GPAs vary more relative to their average. Retained students have more consistent academic performance.

With missing data, the calculated mean and SD may be unreliable, especially for a small group of 30 students.

If missing GPAs are low, the mean could decrease, SD may increase, exaggerating variability.

If missing GPAs are high, the mean could increase, SD may decrease. Incomplete data leads to potential bias, making conclusions less certain.

Mean & SD assume symmetric, normal distributions, which may not be true for GPAs.

They do not reveal skewness (leaning toward low or high GPAs), presence of outliers, or distribution shape. For dropout analysis, understanding the full GPA spread, not just average/SD, is critical.

Boxplots display:

Median GPA, IQR (middle 50% spread), outliers, if present.

They help compare GPA distribution for Dropouts vs Retained, revealing consistency within groups, skewness, or unusual data points.

Boxplot would likely show: Wider spread for Dropouts, more compact GPA cluster for Retained students.

[10]
```python
#gender pay gap
iqr_male = 110-90
iqr_female = 105-85

print(f"Gender pay gap \n Male iqr: {iqr_male}, female iqr: {iqr_female}")
```

```
Gender pay gap
 Male iqr: 20, female iqr: 20
```

Mean vs Median Interpretation — Are Outliers Likely?

Males: Mean = $105k, Median =$98k Females: Mean = $92k, Median =$90k

In both groups, the mean exceeds the median, more so for males. This suggests a right-skewed distribution, meaning some individuals earn significantly more than the typical employee. Outliers (high salaries) are likely present, especially among males, exaggerating the mean.

Both groups have the same IQR, meaning the middle 50% of salaries has similar spread. Despite similar IQR, the higher male mean and SD suggest greater upper-end variability (outliers or very high earners).

Due to outliers and skewness: The median better reflects the typical salary for both groups. The mean is inflated by high earners, giving a misleading sense of average pay.

A boxplot is ideal to reveal:

Median differences, spread of salaries, outliers, and distribution shape.

Adding a violin plot or strip plot overlay provides further insight into salary clustering and gaps.

Simpson's Paradox occurs when overall trends reverse within subgroups. Example: Females may earn less overall, but within high-paying departments (e.g., engineering), they could earn similarly or more than males. Without analyzing department breakdowns, conclusions about the gender pay gap can be misleading. Need to control for: Job roles, departments, experience levels.

[11]

```
#fitness tracker accuracy
cv_a = (310/8050)*100
cv_b=(800/8250)*100
print(f"fitness tracker \n brand A cv: {cv_a:.2f}%, brand B cv: {cv_b:.2f}%")
```

```
fitness tracker
 brand A cv: 3.85%, brand B cv: 9.70%
```

Lower CV indicates more consistent performance. Brand A provides more consistent step counts, with less relative variability compared to Brand B.

A mean higher than median suggests right-skewed data, where some users recorded very high step counts, inflating the average. Tracker B's step count distribution likely has outliers or a long right tail (very high step counts). Typical users see lower daily steps (8100), but the average is pulled up by a few high results.

Both IQR and SD for Brand B are higher, meaning more variability in user step counts, greater spread in both the middle 50% (IQR) and entire dataset (SD). Tracker B's readings are less stable, with frequent high deviations.

"More optimistic" implies it reports inflated step counts compared to reality. To statistically evaluate: Compare readings from both trackers for the same users, if possible. Use a paired-t-test or Bland-Altman plot to assess systematic bias. If B consistently overestimates steps, the claim is supported.

Fitness Tracker Step Count Accuracy Report Overview: Purpose: Compare consistency & accuracy of Brand A vs Brand B. Summary Statistics: Mean, Median, SD, IQR, CV for both brands. Visuals: Boxplots and histograms of daily steps. Density plots showing variability. Key Insights: Consistency levels based on CV, identification of skewness or outliers, evidence of inflated step counts ("optimistic reporting"). Recommendation: Best tracker for users prioritizing accuracy and consistency, note on potential overestimation trends.

[12]
```
#vaccine response

disp_under40 = (140/840)*100
disp_over60 = (85/620)*100

print(f"Vaccine Response \n Under40 cv: {disp_under40:.2f}%, over 60 cv: {disp_over60:.2f}%")
```

```
Vaccine Response
 Under40 cv: 16.67%, over 60 cv: 13.71%
```

The Under 40 group has greater relative variability in antibody levels. Even though their average antibody levels are higher, results are more spread out, indicating inconsistent immune responses across younger participants.

Recommended Visuals: Overlapping Histograms: Shows both groups' antibody distributions side by side. Expect a higher peak for the Over 60 group clustered near lower antibody levels. Boxplots: Clearly compare medians, spreads, and potential outliers. Under 40 likely shows wider IQR and higher max values. These visuals reveal: The younger group has both higher peak responses and greater variability. Older group responses are lower but more consistent.

Mean and SD could hide distinct subgroups within either age bracket: For example: Some Under 40 participants might have very high responses, others much lower. The mean may suggest moderate immunity, but reality reflects a polarized response. In such cases: Histograms or density plots reveal bimodal or skewed patterns. Median and IQR become more reliable for typical response interpretation.

The five-number summary includes: Minimum, Q1 (25th percentile), Median, Q3 (75th percentile), Maximum. This helps: Detect individuals with unusually low or high antibody levels. Monitor for extreme low responders (who may be vulnerable). Identify exceptionally high responders, informing further study.

Descriptive Analysis Supports:

Younger participants: Higher but inconsistent responses; dosage may be sufficient, but variability should be investigated.

Older participants: Lower and more consistent responses; potential need for: Booster doses, adjusted dosage levels, closer monitoring for low responders. Ethical Considerations:

Ensure equitable protection across age groups.

Transparent communication of varying effectiveness.

Adjust protocols to optimize public health outcomes without discrimination.

[13]
```python
#E-Commerce performance

np.random.seed(0)
old_users=np.random.normal(120,15,30)
new_users = np.random.normal(100,30,30)
cv_old=(np.std(old_users)/np.mean(old_users))*100
cv_new=(np.std(new_users)/np.mean(new_users))*100

print(f"E-Commerce Performance \n Old users cv: {cv_old:.2f}%, new users cv: {cv_new:.2f}%")
```

```
E-Commerce Performance
 Old users cv: 12.81%, new users cv: 29.53%
```

The higher CV for new users indicates: Much greater variability in daily cart values among new users. Old users show more consistent spending patterns, suggesting established habits or loyalty.

New users likely have:

Occasional high cart values (right-skewed distribution).

Many lower spending days. This could be due to:

First-time promotions.

Some users making bulk purchases, others testing the platform. Skewness implication:

The mean for new users may be misleading, inflated by a few large spenders.

Median or IQR provides a better picture of typical new user spending.

A campaign boosting spending selectively:

Increases overall mean cart value.

Increases variability (SD and CV rise).

Potentially increases skewness as large transactions occur on campaign days. Implication:

Without separating campaign effects, overall statistics misrepresent organic user behavior. Suggested approach:

Analyze pre-campaign vs post-campaign data separately.

Report median values alongside mean to counter distortion.

A. Segmented Analysis: Compare spending patterns for:

First-time buyers.

Repeat buyers. Helps tailor promotions or loyalty programs to different user types. B. Boxplots with Outlier Detection: Visualize:

Median cart values.

Median cart values.

IQR.

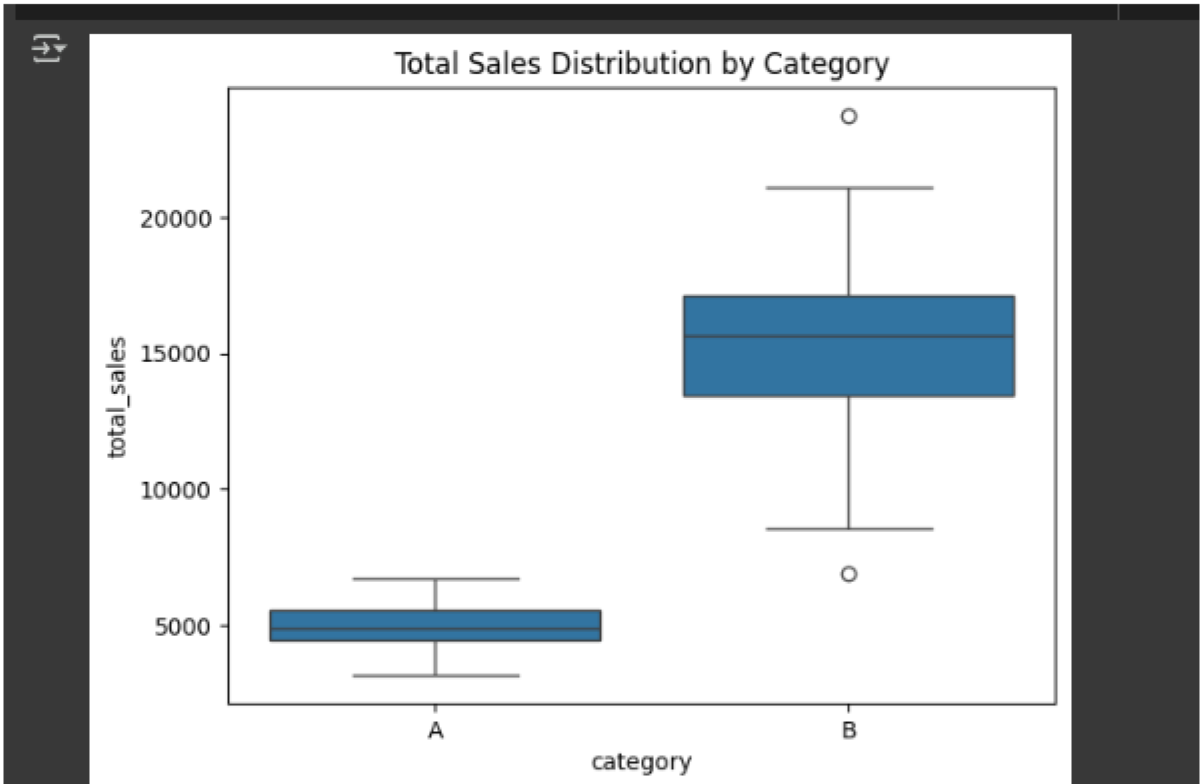
High-spending outliers. Quickly reveals:

Consistency differences between groups.

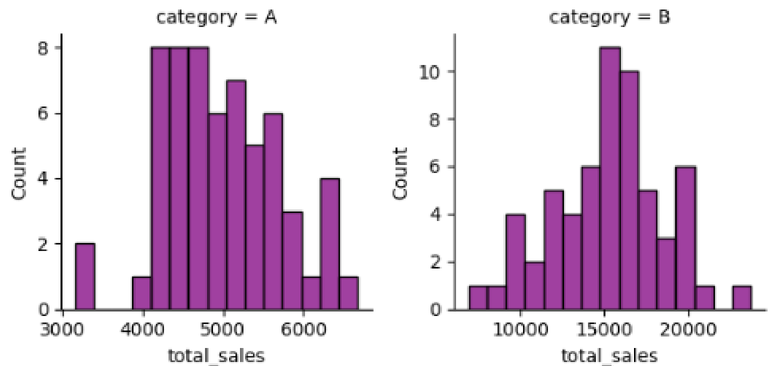
Opportunities to engage big spenders or address spending gaps.

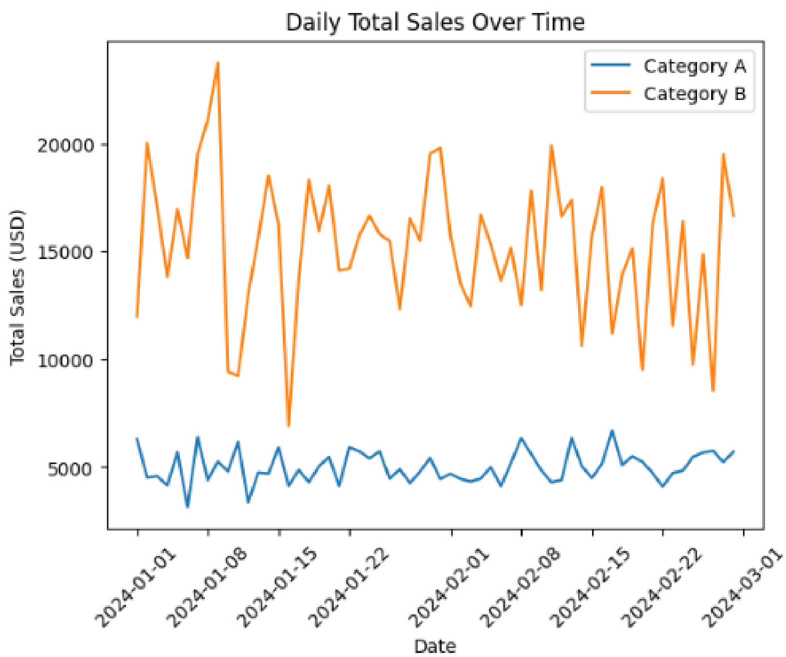
```
# 4.2 Case 1 - Daily Sales Trends for Two Product Categories
import seaborn as sns

# Simulated Dataset for 60 Days
dates = pd.date_range(start="2024-01-01", periods=60)
np.random.seed(1)

category_a_sales = np.random.normal(5000, 800, 60)
category_b_sales = np.random.normal(15000, 4000, 60)
avg_basket_a = np.random.normal(50, 5, 60)
avg_basket_b = np.random.normal(300, 30, 60)
transactions_a = category_a_sales / avg_basket_a
transactions_b = category_b_sales / avg_basket_b

df_sales = pd.DataFrame({
    "date": np.concatenate([dates, dates]),
    "category": ["A"]*60 + ["B"]*60,
    "total_sales": np.concatenate([category_a_sales, category_b_sales]),
    "avg_basket": np.concatenate([avg_basket_a, avg_basket_b]),
    "num_transactions": np.concatenate([transactions_a, transactions_b])
})

# Boxplot for Total Sales by Category
sns.boxplot(data=df_sales, x="category", y="total_sales")
plt.title("Total Sales Distribution by Category")
plt.show()

# Histogram for Total Sales
g = sns.FacetGrid(df_sales, col="category", sharex=False, sharey=False)
g.map_dataframe(sns.histplot, x="total_sales", bins=15, color='purple')
plt.show()

# Time Series Plot
for cat in ["A", "B"]:
    subset = df_sales[df_sales["category"] == cat]
    plt.plot(subset["date"], subset["total_sales"], label=f"Category {cat}")
plt.title("Daily Total Sales Over Time")
```

```python
# Histogram for Total Sales
g = sns.FacetGrid(df_sales, col="category", sharex=False, sharey=False)
g.map_dataframe(sns.histplot, x="total_sales", bins=15, color='purple')
plt.show()

# Time Series Plot
for cat in ["A", "B"]:
    subset = df_sales[df_sales["category"] == cat]
    plt.plot(subset["date"], subset["total_sales"], label=f"Category {cat}")
plt.title("Daily Total Sales Over Time")
plt.xlabel("Date")
plt.ylabel("Total Sales (USD)")
plt.legend()
plt.xticks(rotation=45)
plt.show()

# Coefficient of Variation (CV) Calculation
cv_summary = df_sales.groupby("category").agg({
    "total_sales": lambda x: (np.std(x, ddof=1) / np.mean(x)) * 100,
    "avg_basket": lambda x: (np.std(x, ddof=1) / np.mean(x)) * 100
}).rename(columns={"total_sales": "CV_Total_Sales", "avg_basket": "CV_Avg_Basket"})

print("\n--- Coefficient of Variation Summary ---\n", cv_summary)

# 7-Day Moving Average Plot
for cat in ["A", "B"]:
    subset = df_sales[df_sales["category"] == cat].copy()
    subset.sort_values("date", inplace=True)
    subset["MA_7"] = subset["total_sales"].rolling(window=7).mean()
    plt.plot(subset["date"], subset["MA_7"], label=f"{cat} 7-Day MA")
plt.title("7-Day Moving Average of Total Sales")
plt.xlabel("Date")
plt.ylabel("Total Sales (USD)")
plt.legend()
plt.xticks(rotation=45)
plt.show()
```



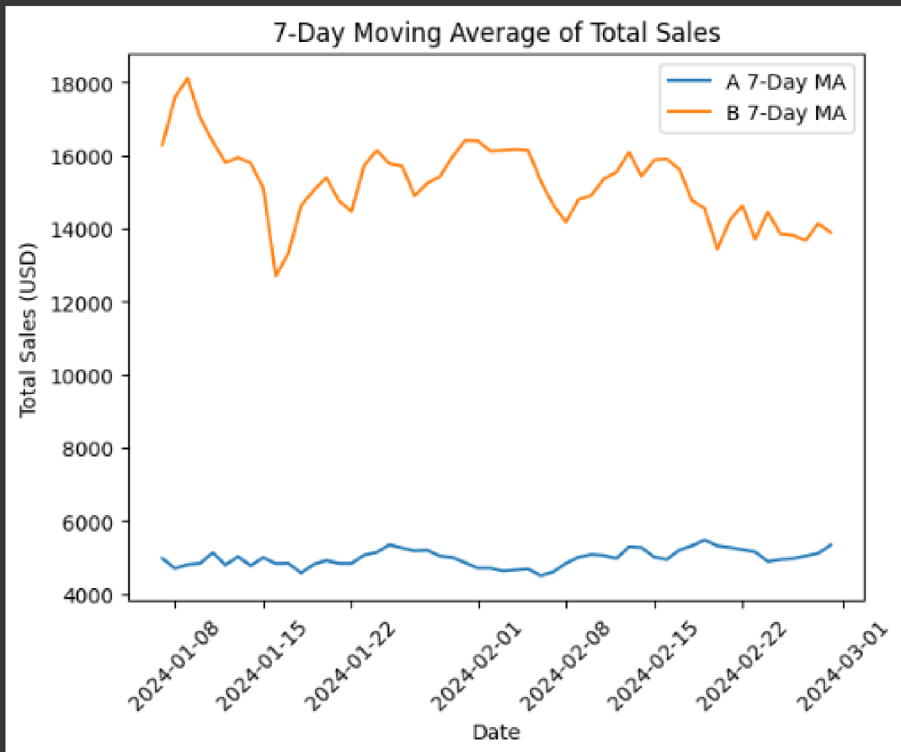
category = A category = B






```
--- Coefficient of Variation Summary ---
```

```
--- Coefficient of Variation Summary ---
          CV_Total_Sales  CV_Avg_Basket
category
A              14.928464       9.693477
B              21.931046      10.491739
```

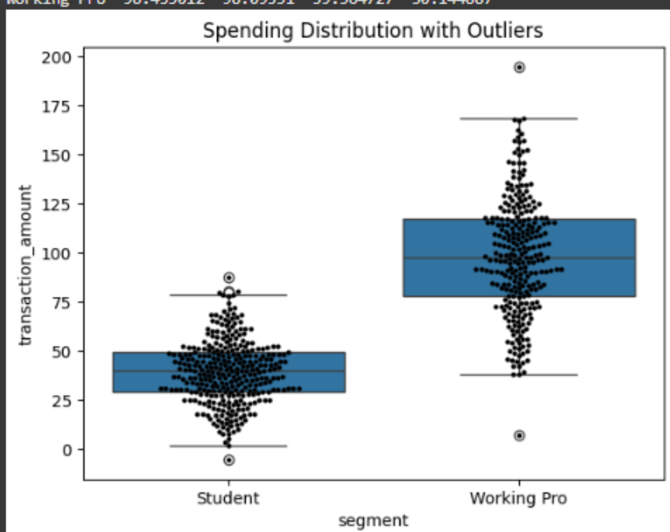


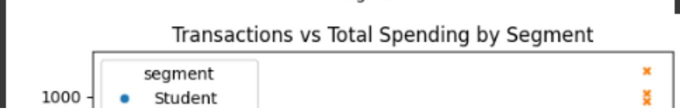

```
# 4.2 Case 2 - Customer Spending by Segment
# Simulated Dataset for Transactions
```

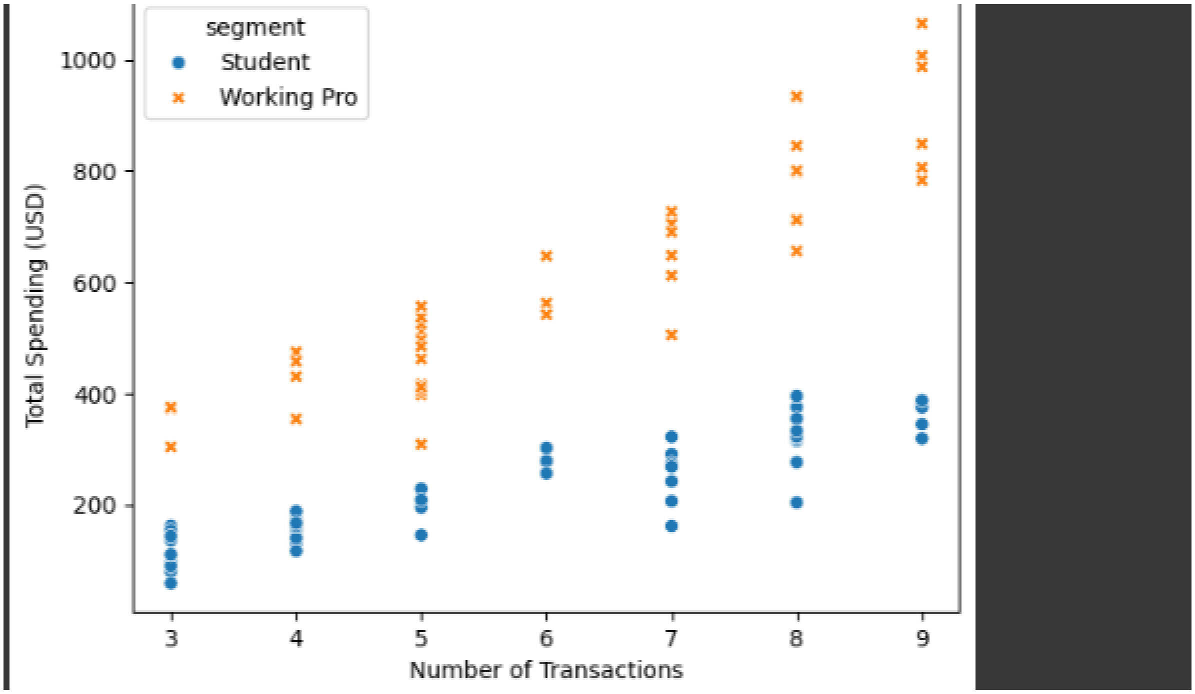
```python
# 4.2 Case 2 - Customer Spending by Segment
# Simulated Dataset for Transactions
np.random.seed(2)

customer_ids = [f"C{i}" for i in range(1, 101)]
segments = np.random.choice(["Student", "Working Pro"], size=100, p=[0.5, 0.5])
transaction_data = []

for cid, seg in zip(customer_ids, segments):
    num_txns = np.random.randint(3, 10)
    if seg == "Student":
        amounts = np.random.normal(40, 15, num_txns)
    else:
        amounts = np.random.normal(100, 30, num_txns)
    for amt in amounts:
        transaction_data.append([cid, seg, amt, pd.Timestamp("2024-01-01") + pd.Timedelta(days=np.random.randint(90))])

df_txns = pd.DataFrame(transaction_data, columns=["customer_id", "segment", "transaction_amount", "date"])

# Violin & Strip Plot
sns.violinplot(data=df_txns, x="segment", y="transaction_amount", inner=None, palette="muted")
sns.stripplot(data=df_txns, x="segment", y="transaction_amount", color='black', size=2, jitter=True)
plt.title("Transaction Amount by Segment")
plt.show()

# KDE Plot
sns.kdeplot(data=df_txns, x="transaction_amount", hue="segment", fill=True)
plt.title("Density of Transaction Amounts")
plt.show()

# Summary Statistics
summary = df_txns.groupby("segment").agg({
    "transaction_amount": ["mean", "median",
        lambda x: np.percentile(x, 75) - np.percentile(x, 25),
        lambda x: (np.std(x, ddof=1) / np.mean(x)) * 100]
})
summary.columns = ["Mean", "Median", "IQR", "CV (%)"]
print("\n--- Spending Summary by Segment ---\n", summary)

# Boxplot with Swarm Overlay
sns.boxplot(data=df_txns, x="segment", y="transaction_amount")
sns.swarmplot(data=df_txns, x="segment", y="transaction_amount", color='black', size=3)
plt.title("Spending Distribution with Outliers")
plt.show()

# Scatter Plot: Number of Transactions vs Total Spending
spending_summary = df_txns.groupby(["customer_id", "segment"]).agg({
```

```python
# Scatter Plot: Number of Transactions vs Total Spending
spending_summary = df_txns.groupby(["customer_id", "segment"]).agg({
    "transaction_amount": ["sum", "count"]
}).reset_index()
spending_summary.columns = ["customer_id", "segment", "total_spending", "num_transactions"]

sns.scatterplot(data=spending_summary, x="num_transactions", y="total_spending",
                hue="segment", style="segment")
plt.title("Transactions vs Total Spending by Segment")
plt.xlabel("Number of Transactions")
plt.ylabel("Total Spending (USD)")
plt.show()
```

```
--- Spending Summary by Segment ---
                  Mean    Median        IQR     CV (%)
segment
Student      39.542992  39.88846  20.014159  40.308091
Working Pro  98.455012  98.09551  39.364727  30.144887
```







Git hub repo link: https://github.com/AnanyaDahal/HCAI5DS02_AnanyaDahal