cse351_hw1_iqbal_fardin_114866279

February 28, 2025

0.1 Question 1:

0.1.1 a. Airline Flight Schedules

- **Availability: Publicly available (partially restricted in some cases)**
- **Justification:**
  - Many airlines provide **real-time flight schedules** through **their websites and APIs** such as:
    - FlightAware API (Real-time flight tracking)
    - OpenSky Network (Open-source flight tracking)
    - FAA Flight Data (Federal Aviation Administration)
  - **Publicly available data includes:**
    - Departure and arrival times
    - Flight numbers and airlines
    - Gate assignments and estimated delays
  - **Restricted data includes:**
    - Passenger manifests and bookings (restricted for privacy)
    - Real-time air traffic control communication (restricted for security)
    - Operational logistics (available only to airline staff or premium users)

0.1.2 b. University Admission Statistics

- **Availability: Partially public, mostly restricted**
- **Justification:**
  - General admission statistics (e.g., acceptance rates, SAT scores, student demographics) are **publicly available** on:
    - Common Data Set Initiative (Detailed admission reports)
    - National Center for Education Statistics (NCES) (Federal education data)
    - Individual university websites often publish their own reports
  - **Restricted data includes:**
    - Individual applicant details (protected under FERPA)
    - Admission committee notes, essays, and recommendations (confidential)
    - Financial aid decisions (restricted to the applicant)

0.1.3 c. Crime Statistics in Your City

- **Availability: Publicly available (some details restricted for privacy and security)**

- **Justification:**
  - Many law enforcement agencies provide open access to crime data for public awareness and research. Some official sources include:
    * FBI Crime Data Explorer (Nationwide crime reports)
    * NYPD Crime Map (Crime data for New York City)
    * Bureau of Justice Statistics (BJS) (Criminal justice system data)
  - **Publicly available data includes:**
    * Number and type of crimes reported
    * Crime rates per neighborhood or district
    * Trends in violent vs. non-violent crimes
  - **Restricted data includes:**
    * Personal information of victims and suspects (protected by privacy laws)
    * Ongoing investigations (restricted to law enforcement agencies)
    * Juvenile crime records (protected under privacy regulations)

0.1.4 d. Movie Box Office Revenue

- **Availability: Publicly available (some financial data restricted)**
- **Justification:**
  - Box office revenue data is widely available through industry sources:
    * Box Office Mojo (Detailed box office earnings)
    * The Numbers (Box office and financial insights)
    * IMDB Box Office Data (Basic box office reports)
  - **Publicly available data includes:**
    * Daily and weekend box office earnings
    * Gross revenue by region (domestic/international)
    * Movie rankings based on earnings
  - **Restricted data includes:**
    * Profit and loss breakdowns (hidden by studios for competitive reasons)
    * Revenue from streaming and licensing deals (not publicly disclosed)
    * Marketing and distribution costs (often estimates, not official figures)

0.1.5 e. Daily Weather Temperature in Your City

- **Availability: Publicly available (some proprietary models restricted)**
- **Justification:**
  - Most weather data is **freely available** through government agencies and open APIs:
    * National Weather Service (NWS) (Official US weather data)
    * NOAA Climate Data (Historical climate and temperature records)
    * OpenWeatherMap API (Real-time weather API)
    * The Weather Channel (Global weather reports)
  - **Publicly available data includes:**
    * Current temperature, humidity, and wind speed
    * Severe weather warnings (hurricanes, tornadoes, storms)
    * Historical weather trends for research purposes

- **Restricted data includes:**
  - Proprietary weather forecasting models (owned by private companies like AccuWeather)
  - Military or classified weather data (used for defense operations)
  - Specialized climate analysis that requires paid access

0.1.6 Summary Table

Data Source	Availability	References
Airline Flight Schedules	Public (basic data), Restricted (detailed)	FlightAware, OpenSky
University Admission Statistics	Public (general data), Restricted (personal data)	Common Data Set, NCES
Crime Statistics	Public (general data), Restricted (sensitive data)	FBI Crime Explorer, NYPD Crime Map
Movie Box Office Revenue	Public (gross revenue), Restricted (detailed finances)	Box Office Mojo, The Numbers
Daily Weather Data	Public (forecasts), Restricted (proprietary models)	NWS, NOAA

0.2 Question 2:

0.3 1. UFC Fight Statistics Dataset

Description: This dataset contains detailed statistics for UFC fights, including information on fighters, match results, strikes landed, takedown success rates, and fight duration. It can be used to analyze trends in fighter performance, strategy effectiveness, and fight outcomes.

- **Source:** UFC Stats
- **Alternative:** Ultimate UFC Dataset on Kaggle
- **GitHub Scraper:** UFC Stats Scraper

Potential Analyses: 1. **Fighter Performance Over Time** - Track how a fighter's striking accuracy, takedown success, and defense evolve over their career. - Identify peak performance years and possible decline phases.

2. **Winning Strategies by Weight Class**
   - Compare statistics between different weight classes to identify the most common winning strategies (e.g., striking-heavy vs. grappling-heavy).
   - Determine if certain attributes (height, reach, fight style) correlate with higher win rates.
3. **Predicting Fight Outcomes**
   - Use machine learning models to predict the winner of an upcoming fight based on historical data.
   - Factors like age, previous fight history, reach advantage, and striking accuracy can be analyzed to improve predictions.

0.4 2. Chess Game Data

Description: This dataset contains millions of chess games played at various levels, from casual online games to world championship matches. It includes details like player ratings, moves played, opening strategies, and game outcomes.

- **Source:** Lichess Game Database
- **Alternative:** FICS Chess Database
- **PGN Database:** Chess.com PGN Downloads

Potential Analyses: 1. **Opening Strategy Effectiveness** - Analyze win rates for various chess openings (e.g., Sicilian Defense vs. King's Indian). - Identify which openings lead to the highest success rates for White and Black at different rating levels.

2. **Player Rating Progression**
    - Track how a player's Elo rating changes over time and identify factors contributing to rapid improvement or stagnation.
    - Compare rating progressions of titled players (e.g., Grandmasters) vs. casual players.
3. **AI vs. Human Playstyles**
    - Compare human gameplay trends with AI engines like Stockfish.
    - Determine how AI-influenced strategies (e.g., AlphaZero-inspired moves) have changed human play over time.

0.5 3. NASA Exoplanet Data

Description: This dataset provides detailed information on exoplanets discovered beyond our solar system, including planet size, orbital period, distance from their host star, and potential habitability factors.

- **Source:** NASA Exoplanet Archive
- **Alternative:** Exoplanet Data from Kaggle
- **JWST:** [https://webbtelescope.org/contents/articles/webbs-impact-on-exoplanet-research]

Potential Analyses: 1. **Habitability Score Prediction** - Use exoplanet parameters (e.g., temperature, atmospheric conditions, distance from the star) to assess the likelihood of supporting life. - Identify the most Earth-like exoplanets discovered.

2. **Exoplanet Size Distribution**
    - Analyze the frequency of different exoplanet sizes (e.g., Earth-sized, Jupiter-sized).
    - Determine whether certain types of stars are more likely to host larger or smaller planets.
3. **Orbital Patterns and Star Types**
    - Study the relationship between a planet's orbital characteristics (e.g., eccentricity, distance) and its host star's properties (e.g., mass, temperature).
    - Identify trends that could guide future exoplanet searches.

0.6 Question 3:

0.6.1 Objective

The goal of this experiment is to determine whether my friends prefer the taste of **Regular Coke** or **Diet Coke** through a controlled blind taste test.

0.6.2 Study Design

1. **Participants:**
   - A sample of **at least 20 participants** from my friend group.
   - Participants should have **no dietary restrictions** that would prevent them from consuming soda.
2. **Experimental Setup:**
   - Each participant will be given two cups labeled **A** and **B**.
   - One cup will contain **Regular Coke**, and the other will contain **Diet Coke**.
   - The order of the drinks will be **randomized** to prevent bias.
3. **Blind Taste Test:**
   - Participants will **not** be informed which cup contains which soda.
   - They will take a sip from each cup and **choose which one they prefer**.
   - They will also be asked to **guess which one is Regular Coke** and which one is Diet Coke.
4. **Data Collection:**
   - **Preferred Drink:** Record whether each participant preferred the drink in **Cup A or Cup B**.
   - **Correct Identification:** Track how many participants correctly identified which drink was Regular Coke and which was Diet Coke.
   - **Additional Notes:** Participants can provide comments on taste differences.
5. **Analysis:**
   - **Count the number of votes** for Regular Coke vs. Diet Coke.
   - **Perform a statistical test** (e.g., **chi-square test**) to determine if there is a significant preference for one over the other.
   - **Analyze identification accuracy** to see if people can correctly distinguish between the two.
6. **Conclusion:**
   - If significantly more participants choose one drink over the other, we conclude that **one is generally preferred**.
   - If the results are close to 50-50, it may suggest that **there is no strong preference**.
   - If most participants struggle to correctly identify the drinks, it could indicate that **the taste difference is not obvious**.

0.6.3 Considerations for Fairness

- **Randomization:** To eliminate order bias, half of the participants receive Regular Coke in Cup A, and the other half receive Diet Coke in Cup A.
- **Controlled Conditions:** Ensure that both drinks are **served at the same temperature** and in identical cups to avoid visual bias.
- **Avoid Brand Influence:** Participants should not see the cans/bottles to prevent preconceived preferences.

0.7 Question 4: The Role of Logarithms in Data Analysis

Logarithms are essential mathematical tools used in data analysis for **scaling, transformation, and interpretation** of numerical data. Below are three major use cases:

0.7.1 1. Data Transformation and Normalization

- Many real-world datasets exhibit **skewed distributions** (e.g., income levels, population sizes, stock prices).
- Applying a **log transformation** converts highly skewed data into a more **normally distributed** form, making statistical analysis more reliable.
- **Example:** Converting exponential data (e.g., **COVID-19 case growth**) into a linear scale for easier interpretation.

0.7.2 2. Handling Multiplicative Relationships

- Logarithms allow **multiplicative relationships** to be converted into **additive** ones.
- This is particularly useful in **machine learning and regression models**, where relationships between variables are often **not linear**.
- **Example:** In economics, the **Cobb-Douglas production function** models output as a product of labor and capital, which can be transformed using logs to estimate the contributions of each factor.

0.7.3 3. Measuring Growth Rates and Percent Changes

- Logarithmic scales are commonly used to measure **relative changes** and **growth rates**.
- **Example:** The **logarithmic return** in finance measures the percentage change in stock prices:

$$r = \ln\left(\frac{P_t}{P_{t-1}}\right)$$

where P_t and P_{t-1} are the stock prices at different times.

- Log scales are also used in the **Richter scale (earthquakes)** and **decibels (sound intensity)** to handle large variations in data.

0.7.4 Conclusion

Logarithms play a vital role in **data science, statistics, and real-world modeling** by simplifying complex relationships, improving data interpretation, and making exponential trends more understandable.

0.8 **Question 5:*

Correlation and causation are two fundamental concepts in statistics and data analysis. **Correlation** refers to a statistical relationship between two variables, meaning that when one variable changes, the other tends to change as well. However, correlation does **not** imply that one variable directly causes the other to change. **Causation**, on the other hand, indicates a **cause-and-effect relationship**, where changes in one variable **directly lead** to changes in another.

A classic example of correlation without causation is the relationship between **ice cream sales and drowning incidents**. Data may show that as ice cream sales increase, drowning incidents

also rise. However, this does not mean that eating ice cream causes drowning. Instead, a third factor—**hot weather**—is influencing both variables. The heat increases ice cream consumption while also encouraging more people to swim, leading to a higher risk of drowning. This is an example of a **spurious correlation**, where a hidden factor affects both observed variables.

To establish causation, **controlled experiments** or advanced statistical techniques such as **randomized controlled trials (RCTs)** and **longitudinal studies** are required. Observing a correlation alone is **not sufficient**—researchers must eliminate **confounding variables** and demonstrate a **direct cause-and-effect link**. Without proper experimental design and rigorous statistical analysis, assuming causation from correlation can lead to misleading conclusions.

In summary, while correlation can help identify relationships between variables, it does not prove causation. Establishing a true causal link requires careful study, eliminating external factors, and applying proper experimental methods.

0.9 Question 6: Statistical Analysis of Study Hours and Exam Scores

Given the dataset of **hours studied per week (X)** and **exam scores (Y)**, we perform statistical calculations to analyze the relationship between these variables.

0.9.1 a. Mean of (X) and (Y)

The mean represents the **average value** of the dataset.

- **Mean of (X) (Hours of Study per Week): 8.125**
- **Mean of (Y) (Exam Score): 73.9**

0.9.2 b. Standard Deviations of (X) and (Y)

The standard deviation measures the **spread** of the data.

- **Standard Deviation of (X): 3.57**
- **Standard Deviation of (Y): 13.33**

0.9.3 c. Covariance between (X) and (Y)

Covariance measures how two variables **vary together**.

- **Covariance: 47.30**

A **positive covariance** indicates that as **study hours increase, exam scores tend to increase as well**.

0.9.4 d. Pearson Correlation Coefficient ((r))

Pearson's correlation coefficient measures the **strength and direction** of a linear relationship.

- **Pearson Correlation Coefficient ((r)) = 0.994**
- **Interpretation:**
  - The value is **very close to 1**, indicating an **extremely strong positive linear relationship**.
  - This means that **as study hours increase, exam scores tend to increase in a highly predictable manner**.

[3]:
```python
import numpy as np

# Data: Hours of Study (X) and Exam Scores (Y)
X = np.array([5, 8, 7, 9, 11, 4.5, 10, 3, 12, 6, 7, 10, 2, 13, 8, 14, 5.5, 3.5,␣
 ↪12.5, 11.5])
Y = np.array([62, 74, 69, 76, 85, 60, 80, 55, 90, 65, 70, 82, 50, 92, 78, 95,␣
 ↪63, 58, 88, 86])


# Function to compute statistics
def compute_statistics(X, Y, dataset_name="Dataset"):
    mean_X = np.mean(X)
    mean_Y = np.mean(Y)
    std_X = np.std(X, ddof=1)  # Sample standard deviation
    std_Y = np.std(Y, ddof=1)  # Sample standard deviation
    cov_XY = np.cov(X, Y, ddof=1)[0, 1]
    pearson_r = np.corrcoef(X, Y)[0, 1]
    correlation_strength = "strong" if abs(pearson_r) > 0.7 else "moderate" if␣
 ↪abs(pearson_r) > 0.4 else "weak"
    correlation_direction = "positive" if pearson_r > 0 else "negative"

    print("=" * 60)
    print(f"{dataset_name:^60}")
    print("=" * 60)
    print(f"Mean of X: {mean_X:.2f}")
    print(f"Mean of Y: {mean_Y:.2f}")
    print("-" * 60)
    print(f"Standard Deviation of X: {std_X:.2f}")
    print(f"Standard Deviation of Y: {std_Y:.2f}")
    print("-" * 60)
    print(f"Covariance between X and Y: {cov_XY:.2f}")
    print("-" * 60)
    print(f"Pearson Correlation Coefficient (r): {pearson_r:.3f}")
    print(f"Correlation Strength: {correlation_strength.capitalize()}")
    print(f"Correlation Direction: {correlation_direction.capitalize()}")
    print("=" * 60)


# Run the function for Study Hours vs Exam Scores
compute_statistics(X, Y, dataset_name="Study Hours vs Exam Scores")
```

```
============================================================
                Study Hours vs Exam Scores
============================================================
Mean of X: 8.12
Mean of Y: 73.90
------------------------------------------------------------
```

```
Standard Deviation of X: 3.57
Standard Deviation of Y: 13.33
------------------------------------------------------------
Covariance between X and Y: 47.30
------------------------------------------------------------
Pearson Correlation Coefficient (r): 0.994
Correlation Strength: Strong
Correlation Direction: Positive
============================================================
```

0.10 Question 7: Spearman Rank Correlation Analysis

Given the dataset of **study hours (X) and exam scores (Y)**, we calculate the **Spearman Rank Correlation Coefficient ( )** to measure the strength and direction of the monotonic relationship between these variables.

0.10.1 a. Ranks for (X) and (Y)

- **Ranks for (X)**: [5.0, 10.5, 8.5, 12.0, 15.0, 4.0, 13.5, 2.0, 17.0, 7.0, 8.5, 13.5, 1.0, 19.0, 10.5, 20.0, 6.0, 3.0, 18.0, 16.0]
- **Ranks for (Y)**: [5.0, 10.0, 8.0, 11.0, 15.0, 4.0, 13.0, 2.0, 18.0, 7.0, 9.0, 14.0, 1.0, 19.0, 12.0, 20.0, 6.0, 3.0, 17.0, 16.0]

0.10.2 b. Differences ((d)) Between Ranks and Squared Differences ((d^2))

- **Differences ((d)):** [0.0, 0.5, 0.5, 1.0, 0.0, 0.0, 0.5, 0.0, -1.0, 0.0, -0.5, -0.5, 0.0, 0.0, -1.5, 0.0, 0.0, 0.0, 1.0, 0.0]
- **Squared Differences ((d^2)):** [0.00, 0.25, 0.25, 1.00, 0.00, 0.00, 0.25, 0.00, 1.00, 0.00, 0.25, 0.25, 0.00, 0.00, 2.25, 0.00, 0.00, 0.00, 1.00, 0.00]

0.10.3 c. Sum of Squared Differences ((d^2))

- **Sum of (d^2): 6.5**

0.10.4 d. Spearman Rank Correlation Coefficient (( ))

- **Spearman Rank Correlation Coefficient (( )) = 0.995**
- **Interpretation:**
  - Since **(= 0.995)** is **very close to 1**, this indicates an **extremely strong positive monotonic relationship**.
  - This means that **as study hours increase, exam scores consistently increase in a predictable pattern**.

[4]:
```
from scipy.stats import rankdata, spearmanr

# Compute ranks for X and Y
ranks_X = rankdata(X)
ranks_Y = rankdata(Y)
```

```python
# Compute differences (d) and squared differences (d^2)
d = ranks_X - ranks_Y
d_squared = d ** 2

# Sum of squared differences (Σ d^2)
sum_d_squared = np.sum(d_squared)

# Calculate Spearman's Rank Correlation Coefficient ( )
spearman_rho, _ = spearmanr(X, Y)

# Interpretation
spearman_strength = "strong" if abs(spearman_rho) > 0.7 else "moderate" if␣
 ↪abs(spearman_rho) > 0.4 else "weak"
spearman_direction = "positive" if spearman_rho > 0 else "negative"

# Print results
print("=" * 70)
print(f"{'Spearman Rank Correlation Analysis':^70}")
print("=" * 70)
print(f"Ranks for X: {ranks_X}")
print(f"Ranks for Y: {ranks_Y}")
print("-" * 70)
print(f"Differences (d) between ranks: {d}")
print(f"Squared Differences (d²): {d_squared}")
print(f"Sum of Squared Differences (Σ d²): {sum_d_squared}")
print("-" * 70)
print(f"Spearman Rank Correlation Coefficient ( ): {spearman_rho:.3f}")
print(f"Correlation Strength: {spearman_strength.capitalize()}")
print(f"Correlation Direction: {spearman_direction.capitalize()}")
print("=" * 70)
```

```
======================================================================
                  Spearman Rank Correlation Analysis
======================================================================
Ranks for X: [ 5.  10.5  8.5 12.  15.   4.  13.5  2.  17.   7.   8.5 13.5  1.
19.
 10.5 20.   6.   3.  18.  16. ]
Ranks for Y: [ 5. 10.  8. 11. 15.  4. 13.  2. 18.  7.  9. 14.  1. 19. 12. 20.
6.  3.
 17. 16.]
----------------------------------------------------------------------
Differences (d) between ranks: [ 0.   0.5  0.5  1.   0.   0.   0.5  0.  -1.   0.
-0.5 -0.5  0.   0.
 -1.5  0.   0.   0.   1.   0. ]
Squared Differences (d²): [0.   0.25 0.25 1.   0.   0.   0.25 0.   1.   0.
0.25 0.25 0.   0.
 2.25 0.   0.   0.   1.   0.  ]
Sum of Squared Differences (Σ d²): 6.5
```

```
----------------------------------------------------------------------
Spearman Rank Correlation Coefficient ( ): 0.995
Correlation Strength: Strong
Correlation Direction: Positive
======================================================================
```

0.11 Question 8:

0.11.1 Key Findings from Question 6 and Question 7

- **Pearson Correlation Coefficient ((r)): 0.994**
- **Spearman Rank Correlation Coefficient (( )): 0.995**
- Both values are **very close to 1**, indicating a **strong positive correlation** in both measures.

0.11.2 Similarities Between Pearson's and Spearman's Correlation

1. **Strong Positive Relationship**
   - Both **Pearson's ((r)) and Spearman's (( )) values are near 1**, confirming that as study hours increase, exam scores also increase.
   - This suggests a **strong association** between the variables in both linear and monotonic terms.
2. **Direction of Relationship**
   - Both coefficients are **positive**, meaning the relationship between study hours and exam scores is **positively correlated**—students who study more tend to score higher.

0.11.3 Differences Between Pearson's and Spearman's Correlation

1. **Type of Relationship Measured**
   - **Pearson's Correlation ((r))** measures the **strength of a linear relationship**.
   - **Spearman's Correlation (( ))** measures the **strength of a monotonic relationship**, which applies even if the relationship is non-linear.
2. **Handling of Data Distribution and Outliers**
   - **Pearson's correlation is sensitive to outliers**, meaning extreme values can impact the correlation coefficient.
   - **Spearman's correlation is rank-based**, meaning it is less affected by outliers and measures how well the data maintains a consistent increasing or decreasing trend.
3. **Applicability to Different Data Types**
   - **Pearson's correlation** requires **normally distributed, continuous data** and assumes a **linear relationship**.
   - **Spearman's correlation** can be used for **ordinal, non-linear, or non-normally distributed data** because it ranks values instead of using exact numbers.

0.11.4 Conclusion

- In this dataset, **both Pearson's and Spearman's correlation values are extremely high**, meaning **study hours and exam scores have a very strong positive relationship**.
- The **small difference between (r) and ( )** suggests that the relationship between study hours and exam scores is **both linear and monotonic**, meaning students who study more tend to perform better in a predictable, proportional manner.

- If the data had **outliers or a non-linear relationship**, **Spearman's ( )** would be the preferred measure.

0.11.5 Final Interpretation

Since **(r = 0.994) and (rho = 0.995)**, we can confidently conclude that the relationship between **study hours and exam scores is highly correlated in both linear and rank-based measures**. There is no significant difference between the two correlation coefficients in this dataset, but **Spearman's correlation would be more reliable in cases with non-linear trends or outliers**.

0.12 Question 9: Comparison of Means and Standard Deviations

In each pair of distributions, we compare which set has a **greater mean ( )** and **greater standard deviation ( )** without explicitly calculating them.

0.12.1 (a)

Distributions:

- **i.** 3, 5, 5, 5, 8, 11, 11, 11, 13
- **ii.** 3, 5, 5, 5, 8, 11, 11, 11, 20

Comparison:

- **Greater Mean: ii**
  - The values in **ii** are identical to **i**, except the last value is **20 instead of 13**.
  - Since the largest number is **greater**, the **mean increases**.
- **Greater Standard Deviation: ii**
  - The spread of values is **wider** in **ii** due to the larger extreme value (**20 vs. 13**), increasing .

0.12.2 (b)

Distributions:

- **i.** -20, 0, 0, 0, 15, 25, 30, 30
- **ii.** -40, 0, 0, 0, 15, 25, 30, 30

Comparison:

- **Greater Mean: i**
  - Both distributions are identical except that **ii** has **-40 instead of -20**.
  - Since **-40 is smaller than -20**, the **mean of ii is lower**, making **i have the greater mean**.
- **Greater Standard Deviation: ii**

– The **larger negative value (-40 vs. -20)** in **ii** creates a **greater spread**, increasing .

0.12.3 (c)

Distributions:

- **i.** 0, 2, 4, 6, 8, 10
- **ii.** 20, 22, 24, 26, 28, 30

Comparison:

- **Greater Mean: ii**
  - The values in **ii** are simply **20 units larger** than those in **i**, shifting the **entire distribution upward**, resulting in a **higher mean**.
- **Greater Standard Deviation: Neither; they are the same**
  - Both distributions have **identical spacing between numbers**.
  - **Shifting** all numbers **by a constant amount (20)** does **not change** the standard deviation.

0.12.4 (d)

Distributions:

- **i.** 100, 200, 300, 400, 500
- **ii.** 0, 50, 300, 550, 600

Comparison:

- **Greater Mean: Neither; both have the same mean**
  - The sum of values in both distributions is the same, so their **means are equal**.
- **Greater Standard Deviation: ii**
  - The values in **ii** are more **spread out** (especially **0 and 600** vs. **100 and 500** in i).
  - **A wider spread increases standard deviation ( ).**

0.12.5 Final Summary Table

Pair	Greater Mean ( )	Greater Standard Deviation ( )
(a)	ii	ii
(b)	i	ii
(c)	ii	Same
(d)	Same	ii

0.12.6 Conclusion

- **Mean ( ) is affected by shifts in values**, while **standard deviation ( ) is affected by spread.**
- Adding a **larger extreme value** increases both mean and standard deviation.
- **Shifting all values by a constant does not change standard deviation.**

[5]:
```
import numpy as np

# Define the distributions
distributions = {
    "(a) i": np.array([3, 5, 5, 5, 8, 11, 11, 11, 13]),
    "(a) ii": np.array([3, 5, 5, 5, 8, 11, 11, 11, 20]),
    "(b) i": np.array([-20, 0, 0, 0, 15, 25, 30, 30]),
    "(b) ii": np.array([-40, 0, 0, 0, 15, 25, 30, 30]),
    "(c) i": np.array([0, 2, 4, 6, 8, 10]),
    "(c) ii": np.array([20, 22, 24, 26, 28, 30]),
    "(d) i": np.array([100, 200, 300, 400, 500]),
    "(d) ii": np.array([0, 50, 300, 550, 600])
}

# Compute mean and standard deviation for each distribution
results = {}
for label, data in distributions.items():
    mean = np.mean(data)
    std_dev = np.std(data, ddof=1)  # Sample standard deviation
    results[label] = {"Mean": mean, "Std Dev": std_dev}

# Compare means and standard deviations for each pair
comparison_results = []
pairs = [("(a) i", "(a) ii"), ("(b) i", "(b) ii"), ("(c) i", "(c) ii"), ("(d)␣
 ↪i", "(d) ii")]

for pair in pairs:
    mean_winner = pair[0] if results[pair[0]]["Mean"] >␣
 ↪results[pair[1]]["Mean"] else pair[1] if results[pair[0]]["Mean"] <␣
 ↪results[pair[1]]["Mean"] else "Same"
    std_dev_winner = pair[0] if results[pair[0]]["Std Dev"] >␣
 ↪results[pair[1]]["Std Dev"] else pair[1] if results[pair[0]]["Std Dev"] <␣
 ↪results[pair[1]]["Std Dev"] else "Same"
    comparison_results.append((pair[0], pair[1], mean_winner, std_dev_winner))

# Print results in a readable format
print("=" * 50)
print(f"{'Comparison of Means and Standard Deviations':^50}")
print("=" * 50)

for pair in comparison_results:
```

```
    print(f"Pair: {pair[0]} vs {pair[1]}")
    print(f"  - Greater Mean: {pair[2]}")
    print(f"  - Greater Standard Deviation: {pair[3]}")
    print("-" * 50)
```

```
==================================================
   Comparison of Means and Standard Deviations
==================================================
Pair: (a) i vs (a) ii
  - Greater Mean: (a) ii
  - Greater Standard Deviation: (a) ii
--------------------------------------------------
Pair: (b) i vs (b) ii
  - Greater Mean: (b) i
  - Greater Standard Deviation: (b) ii
--------------------------------------------------
Pair: (c) i vs (c) ii
  - Greater Mean: (c) ii
  - Greater Standard Deviation: Same
--------------------------------------------------
Pair: (d) i vs (d) ii
  - Greater Mean: Same
  - Greater Standard Deviation: (d) ii
--------------------------------------------------
```

0.13 Question 10:

We are given the probabilities: - P(A) = 0.3 - P(B) = 0.7

0.13.1 (a) Can we compute P(A and B) with only P(A) and P(B)?

No, we **cannot** determine P(A and B) unless we know whether **A and B are independent or dependent**. If the events are **dependent**, we would need additional information, such as P(A|B) or P(B|A), to compute P(A and B).

0.13.2 (b) Assuming A and B are independent:

For **independent** events, we use the following formulas:

1. **P(A and B)** P(A and B) = P(A) × P(B) P(A and B) = 0.3 × 0.7 = 0.21
2. **P(A or B)** P(A or B) = P(A) + P(B) - P(A and B) P(A or B) = 0.3 + 0.7 - 0.21 = 0.79
3. **P(A|B) (Conditional Probability)** P(A|B) = P(A and B)/P(B) P(A|B) = 0.21/0.7 = 0.3

[6]:
```
# Given probabilities
P_A = 0.3  # Probability of event A
P_B = 0.7  # Probability of event B
```

```python
# (a) P(A and B) cannot be determined without knowing dependence

# (b) Assuming A and B are independent:
P_A_and_B = P_A * P_B  # P(A and B) = P(A) * P(B) for independent events
P_A_or_B = P_A + P_B - P_A_and_B  # P(A or B) = P(A) + P(B) - P(A and B)
P_A_given_B = P_A_and_B / P_B  # P(A|B) = P(A and B) / P(B)

# Output results
print(f"P(A and B) = {P_A_and_B:.4f}")
print(f"P(A or B) = {P_A_or_B:.4f}")
print(f"P(A | B) = {P_A_given_B:.4f}")
```

```
P(A and B) = 0.2100
P(A or B) = 0.7900
P(A | B) = 0.3000
```

0.14 Question 11:

Probability and statistics are closely related but serve different purposes. **Probability** is the mathematical framework that deals with predicting the likelihood of different outcomes before any data is collected. It's all about working from a known model to possible results. For example, if you flip a fair coin, probability tells you there's a **50%** chance of landing on heads and a **50%** chance of tails. It's theoretical, dealing with outcomes that **haven't happened yet**.

Statistics, on the other hand, works **backward**—it starts with real-world data and tries to make sense of it. Instead of assuming a known model, statistics **analyzes** data to determine patterns, relationships, and trends. For example, if you flip a coin **100 times** and get heads **55 times**, statistics helps determine if the coin is fair or biased based on the data collected. It's all about making inferences **after observing real events**.

The key difference is that **probability is predictive**, and **statistics is analytical**. Probability goes from **assumptions to outcomes**, while statistics goes from **outcomes to conclusions**. Probability helps us set expectations, while statistics helps us understand and verify reality.

0.15 Question 12: Disease Screening Test Analysis

A disease affects a small percentage of the population, and a screening test is used to detect it. However, like all tests, it is not perfectly accurate—false positives and false negatives occur. The following table shows the test outcomes:

Outcome	Disease Present (D+)	Disease Absent (D-)	Total
Test Positive (T+)	95	105	200
Test Negative (T-)	5	1795	1800
Total	**100**	**1900**	**2000**

We calculate the following probabilities:

0.15.1 (b) P(D+) - The Prior Probability of Having the Disease

P(D+) = Total with disease / Total population = 100/2000 = 0.05 (5%)

0.15.2 (c) P(T+) - The Total Probability of Testing Positive

P(T+) = Total positive tests / Total population = 200/2000 = 0.10 (10%)

0.15.3 (d) P(D+ | T+) - The Probability of Having the Disease Given a Positive Test

P(D+ | T+) = True Positives / Total positive tests = 95/200 = 0.475 (47.5%)

Interpretation: If someone tests positive, there is only a 47.5% chance that they actually have the disease.

0.15.4 (e) P(T+ | D+) - The Probability of Testing Positive Given the Presence of the Disease (Sensitivity)

P(T+ | D+) = True Positives / Total with disease = 95/100 = 0.95 (95%)

Interpretation: The test correctly identifies 95% of individuals who have the disease.

0.15.5 (f) P(T- | D-) - The Probability of Testing Negative Given the Absence of the Disease (Specificity)

P(T- | D-) = True Negatives / Total healthy individuals = 1795/1900 0.9447 (94.47%)

Interpretation: The test correctly identifies approximately 94.47% of healthy individuals as disease-free.

0.15.6 Interpretation of the Test Effectiveness

- **Sensitivity (95%)** indicates the test is very effective at detecting those with the disease.
- **Specificity (94.47%)** shows that the test is also effective at identifying healthy individuals.
- However, the **positive predictive value** P(D+ | T+) is only 47.5%, meaning that more than half of the individuals who test positive might not actually have the disease. This highlights the importance of confirmatory testing before making clinical decisions based solely on this screening test.

[4]:
```
# Given data
total_population = 2000
disease_present = 100
disease_absent = 1900
test_positive = 200
test_negative = 1800
true_positives = 95
false_positives = 105
true_negatives = 1795
false_negatives = 5
```

```python
# (a) P(D+) - Prior probability of having the disease
P_D_plus = disease_present / total_population

# (b) P(T+) - Total probability of testing positive
P_T_plus = test_positive / total_population

# (c) P(D+ | T+) - Probability of having the disease given a positive test␣
 ↪result
P_D_plus_given_T_plus = true_positives / test_positive

# (d) P(T+ | D+) - Sensitivity (True Positive Rate)
P_T_plus_given_D_plus = true_positives / disease_present

# (e) P(T- | D-) - Specificity (True Negative Rate)
P_T_minus_given_D_minus = true_negatives / disease_absent

# Print results
print(f"P(D+) = {P_D_plus:.4f} (5%)")
print(f"P(T+) = {P_T_plus:.4f} (10%)")
print(f"P(D+ | T+) = {P_D_plus_given_T_plus:.4f} (47.5%)")
print(f"P(T+ | D+) = {P_T_plus_given_D_plus:.4f} (95%)")
print(f"P(T- | D-) = {P_T_minus_given_D_minus:.4f} (94.47%)")
```

```
P(D+) = 0.0500 (5%)
P(T+) = 0.1000 (10%)
P(D+ | T+) = 0.4750 (47.5%)
P(T+ | D+) = 0.9500 (95%)
P(T- | D-) = 0.9447 (94.47%)
```

1 Used Resources

AI-generated responses were used for **clarifying formulas, coding structure, and statistical concepts**. Answers were rewritten and verified using additional sources.

1.1 1. Generative AI Assistance (ChatGPT, February 2025)

I used ChatGPT for **coding assistance, mathematical explanations, and Jupyter-related commands**. All responses were rewritten, manually tested, and verified using additional resources.

Question/Topic	Prompt Used	How I Used It
Q2: Publicly Available Datasets	*"What are some unique publicly available datasets for analysis?"*	AI helped identify **niche datasets** (UFC fight stats, Chess data, Exoplanet data), but I manually selected and verified the sources.

Question/Topic	Prompt Used	How I Used It
Q3: Experimental Design (Coke vs. Diet Coke Test)	*"How do you properly design a blind taste test for two sodas?"*	Used AI for **basic structure**, then rewrote with references from experimental design sources.
Q4: Logarithm Use Cases	*"How are logarithms used in real-world data science applications?"*	AI suggested **three broad use cases**, which I refined and confirmed with **Wolfram MathWorld**.
Q5: Correlation vs. Causation	*"Explain correlation vs. causation with examples."*	AI provided general **examples**, but I rewrote them and verified with **Google searches**.
Q6-Q7: Correlation Calculations	*"How do I manually calculate Pearson and Spearman correlation?"*	Used AI for **formula breakdown**, but I manually worked through examples before coding.
Q10: Probability Computations	*"Write Python code to calculate conditional probability P(A B) given P(A) and P(B)."*	
Q12: Sensitivity & Specificity Computations	*"How do you calculate sensitivity, specificity, and Bayes' theorem for a medical test?"*	AI provided formulas, but I **verified all calculations** using Google and statistical resources.
Jupyter Notebook: Export Issues	*"How do I export a Jupyter Notebook to PDF in PyCharm?"*	AI suggested `nbconvert`, but I tested multiple methods including **HTML to PDF conversion**.
PyCharm: Running Jupyter Notebooks	*"How do I set up Jupyter Notebooks in PyCharm and run a `.ipynb` file?"*	AI provided steps, but I also confirmed by setting up PyCharm's **Jupyter support manually**.
Python Script Conversion	*"How do I convert a Jupyter Notebook to a `.py` script?"*	AI suggested `jupyter nbconvert --to script`, which I tested and used for exporting my `.py` file.