ASSIGNMENT 1

16-25 August 2024

SHAIK SAHIL CHANDA
23B0943

PALURI GNANA KOUSHIK REDDY
23B1000

DEVANGAM KISHAN TEJA
23B1061

CS215 Data Analysis and Interpretation

Contents

1 Let's Gamble

First, let us calculate the probability of getting a prime number when a die is rolled.

Prime numbers on the die are $\{2, 3, 5\}$.
Total possible outcomes are $\{1, 2, 3, 4, 5, 6\}$.
Therefore, the probability of getting a prime number on the die is:

$$P(\text{Prime}) = \frac{3}{6} = \frac{1}{2}$$

Now, let us consider two cases: when n is odd and when n is even.

Case 1: n is odd

In the first n trials, there are only two possibilities:

- **Number of wins of A > Number of wins of B:**

  The probability of this happening is $\frac{1}{2}$, since in each roll, the probability of A winning and B winning are equal, $\frac{1}{2}$. If this happens, the probability of A finally winning is 1, since A is already winning by the time he throws the $(n+1)$th die.

- **Number of wins of A < Number of wins of B:**

  The probability of this happening is $\frac{1}{2}$. If this happens, the probability of A finally winning is 0, as the only possible outcomes are B winning or A and B having an equal number of wins by the end of the $(n+1)$th die roll.

So, the probability of A winning when n is odd is:

$$P_A = \left(\frac{1}{2} \times 1\right) + \left(\frac{1}{2} \times 0\right) = \frac{1}{2}$$

Case 2: n is even

In the first n trials, there are three possibilities:

- **Number of wins of A > Number of wins of B:**

  The probability of this happening is $\frac{1}{3}$. If this happens, the probability of A finally winning is 1, since A is already winning by the time he throws the $(n+1)$th die.

- **Number of wins of A < Number of wins of B:**

  The probability of this happening is $\frac{1}{3}$. If this happens, the probability of A finally winning is 0, because by the end of the n trials, B will be leading by at least 2 wins. So, even if A wins in the $(n+1)$th trial, A loses.

- **Number of wins of A = Number of wins of B:**

  The probability of this happening is $\frac{1}{3}$. If this happens, the probability of A finally winning is $\frac{1}{2}$, as A will only win if he wins in the $(n+1)$th trial. By the end of the n trials, each will have an equal number of wins, so to win, A must win the last trial.

So, the probability of A winning when n is even is:

$$P_A = \left(\frac{1}{3} \times 1\right) + \left(\frac{1}{3} \times 0\right) + \left(\frac{1}{3} \times \frac{1}{2}\right) = \frac{1}{3} + \frac{1}{6} = \frac{1}{2}$$

Conclusion

As the cases are mutually exhaustive and the probability for both cases is the same, we can conclude that the probability that A wins is $\frac{1}{2}$.

2 Two Trading Teams

Probability Calculation for Trading Game

Let:

$$P_A = \text{Probability of winning against Team A,}$$
$$P_B = \text{Probability of winning against Team B.}$$

Since Team B is better than Team A, we have:

$$P_A > P_B.$$

Option 1: A-B-A

Consider the possible outcomes that result in a win:

A - B - A:
$W - W - L$
$W - W - W$
$L - W - W$

The probability of winning with Option 1, denoted as P_1, is:

$$\begin{aligned} P_1 &= P_A \cdot P_B \cdot (1 - P_A) + P_A \cdot P_B \cdot P_A + (1 - P_A) \cdot P_B \cdot P_A \\ &= P_A \cdot P_B \cdot (2 - P_A). \end{aligned}$$

Option 2: B-A-B

Now consider the possible outcomes for this sequence:

B - A - B:
$W - W - L$
$W - W - W$
$L - W - W$

The probability of winning with Option 2, denoted as P_2, is:

$$\begin{aligned} P_2 &= P_B \cdot P_A \cdot (1 - P_B) + P_B \cdot P_A \cdot P_B + (1 - P_B) \cdot P_A \cdot P_B \\ &= P_A \cdot P_B \cdot (2 - P_B). \end{aligned}$$

Comparison of Options

Since $P_A > P_B$, we have:

$$\begin{aligned} 2 - P_A &< 2 - P_B, \\ P_A \cdot P_B \cdot (2 - P_A) &< P_A \cdot P_B \cdot (2 - P_B). \end{aligned}$$

Therefore, $P_1 < P_2$, meaning Option 2 (**B-A-B**) provides a higher probability of winning.

3 Random Variables

3.1

Given: Let Q_1, Q_2 be non-negative random variables. Let $P(Q_1 < q_1) \geq 1 - p_1$ and $P(Q_2 < q_2) \geq 1 - p_2$, where q_1, q_2 are non-negative constants.
To prove:

$$P(Q_1 Q_2 < q_1 q_2) \geq 1 - (p_1 + p_2)$$

Proof:

Consider the events E_1, E_2 as

$$E_1 = \{Q_1 \geq q_1\}$$

$$E_2 = \{Q_2 \geq q_2\}$$

The given inequalities can be written as

$$P(Q_1 \geq q_1) \leq p_1 \Longrightarrow P(E_1) \leq p_1 \tag{1}$$

and

$$P(Q_2 \geq q_2) \leq p_2 \Longrightarrow P(E_2) \leq p_2 \tag{2}$$

Now, we are interested in the event $E = \{Q_1 Q_2 \geq q_1 q_2\}$. Because Q_1, Q_2 are

non-negative, $Q_1Q_2 \geq q_1q_2$ guarantees that atleat one of $Q1 \geq q1$ or $Q2 \geq q2$ holds. i.e, The event E is a subset of $E_1 \cup E_2$

$$E \subseteq E_1 \cup E_2$$

Thus,

$$P(E) \leq P(E_1 \cup E_2)$$

$$\Rightarrow P(E) \leq P(E_1) + P(E_2)$$

and from equation (1) and (2) we get,

$$P(E) \leq p_1 + p_2$$

i.e,

$$P(Q_1Q_2 \geq q_1q_2) \leq p_1 + p_2$$

and thus,

$$P(Q_1Q_2 < q_1q_2) \geq 1 - (p_1 + p_2)$$

3.2

To prove: For all i,

$$|x_i - \mu| \leq \sigma\sqrt{n-1}$$

where, μ is the mean and σ is the standard deviation

Proof:

From the definition of standard deviation σ of the values $\{x_i\}_{i=1}^n$ we get:

$$\sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu)^2$$

$$\Rightarrow (n-1)\sigma^2 = \sum_{i=1}^{n}(x_i - \mu)^2$$

Now, since all terms on R.H.S. are non-negative, each individual term must be less than the total sum

$$(x_i - \mu)^2 \leq \left(\sum_{i=1}^{n}(x_i - \mu)^2\right) = (n-1)\sigma^2$$

and thus for each i:

$$(x_i - \mu)^2 \leq (n-1)\sigma^2$$
$$\Rightarrow |x_i - \mu| \leq \sigma\sqrt{n-1}$$

On increasing the size n of data points, the given inequality suggests that each x_i can possibly lie at larger distances from the mean μ. Thus it provides a weak bound on the range as n increases.

Whereas, Chebyshev's inequality suggests that the probability of any value being far from the mean becomes smaller as we consider larger k. It provides a bound on the number of data points that can possibly lie at least by a certain distance away from the mean. And thus it provides a better idea on how these data points are distributed rather than just saying that its range is larger.

4 Staff Assistant

4.1 (a)

E is the event that we hire the best assistant, and E_i is the event that ith candidate is the best and we hire him. Clearly,

$$Pr(E) = \sum_{i=1}^{n} Pr(E_i)$$

- The probability that we hire the ith the candidate and the ith candidate is the best, is equal to the probability for the ith candidate to be the best, multiplied by the probability that we choose the ith candidate given he is the best (Bayes Theorem).

$$Pr(E_i) = P(ith \text{ candidate is selected} \mid ith \text{ candidate is the best})$$

$$\text{x } P(ith \text{ candidate is the best})$$

- Since the candidates are interviewed in a random order, chosen uniformly at random from all $n!$ possible orderings, probability for the ith candidate to be the best is $\frac{1}{n}$.
- Now, let the ith candidate be the best.
- For $i \leq m$: Since we reject all the first m candidates, the probability that we select the best candidate if he is among the first m candidates is zero.

$$Pr(E_i) = 0, \text{ for all } 0 \leq i \leq m$$

- For $i > m$: The best one among the interviewed $i-1$ candidates must be among the first m candidates for the ith candidate to get chosen. The probability for such an event is $\frac{m}{i-1}$. And thus,

$$Pr(E_i) = \frac{m}{i-1}\frac{1}{n}, \text{ for all } m < i \leq n$$

Therefore, the probability $Pr(E)$ that we select the best candidate is

$$Pr(E) = \sum_{i=m+1}^{n} \frac{m}{n}\frac{1}{(i-1)}$$

4.2 (b)

Consider the graph of $y = \frac{1}{x}$ in the range m to n.

The expression $\sum_{j=m+1}^{n} \frac{1}{j-1}$ can be interpreted as sum of areas of rectangular strips with width 1 and heights $\frac{1}{m}, \frac{1}{m+1}, ...\frac{1}{m-1}$. This area can be bounded by considering integral of y from m to n and from $m-1$ to $n-1$.

For the lower bound, we consider the integral of $\frac{1}{x}$ from m to n

$$\int_m^n \frac{1}{x} dx = ln(n) - ln(m)$$

Thus, we have

$$\sum_{j=m+1}^{n} \frac{1}{j-1} \geq ln(n) - ln(m)$$

Now for upper bound, consider the integral of $\frac{1}{x}$ from $m-1$ to $n-1$:

$$\int_{m-1}^{n-1} \frac{1}{x} dx = ln(n-1) - ln(m-1)$$

and thus,

$$\sum_{j=m+1}^{n} \frac{1}{j-1} \leq ln(n-1) - ln(m-1)$$

Combining these results, we get the bound for $Pr(E)$ as

$$\frac{m}{n}\left(ln(n) - ln(m)\right) \leq Pr(E) \leq \frac{m}{n}\left(ln(n-1) - ln(m-1)\right)$$

4.3 (c)

To find maximum consider the function,

$$f(m) = \frac{m}{n}(\ln(n) - \ln(m))$$

and differentiate $f(m)$ with respect to m:

$$f'(m) = \frac{d}{dm}\left(\frac{m}{n}(\ln(n) - \ln(m))\right)$$

$$f'(m) = \frac{1}{n}(\ln(n) - \ln(m)) - \frac{1}{n}$$

Solve $f'(m) = 0$ to find the critical points

$$(\ln(n) - \ln(m)) - 1 = 0$$

$$\frac{n}{m} = e$$

Thus, $m = \frac{n}{e}$ is a critical point.

To verify that this critical point is a maximum, consider the second derivative:

$$f''(m) = \frac{d}{dm}\left(\frac{1}{n}(\ln(n) - \ln(m)) - \frac{1}{n}\right)$$

$$f''(m) = -\frac{1}{mn}$$

Since $f''(m) < 0$ for all $m > 0$, the function $f(m)$ is concave, and $m = \frac{n}{e}$ is indeed a maximum point.

Therefore, the function $\frac{m}{n}(\ln(n) - \ln(m))$ is maximized when $m = \frac{n}{e}$.

From the lower bound of $Pr(E)$ obtained above,

$$Pr(E) \geq \frac{m}{n}\left(ln(\frac{n}{m})\right)$$

For the choice of $m = \frac{n}{e}$, we get

$$Pr(E) \geq \frac{1}{e}\left(ln(e)\right)$$

$$Pr(E) \geq \frac{1}{e}$$

5 Free Trade

Let the k^{th} place be the most probable place:

— — — k

- There are $n = 200$ numbers. Only one of the $(k-1)$ people before k should have the same number as k, i.e., there are $\binom{k-1}{1}$ possibilities.
- The remaining $k-2$ people should have $n-1$ numbers distributed among themselves with no two having the same number, i.e., $\binom{n-1}{k-2} \cdot (k-2)!$.

The total number of possibilities, as each person before k can get any number from 1 to n, is:

$$n^{k-1}.$$

Thus, the probability $P(k)$ that the k^{th} place is the most probable is:

$$P(k) = \frac{\binom{k-1}{1} \cdot \binom{n-1}{k-2} \cdot (k-2)!}{n^{k-1}}.$$

Simplifying $P(k)$:

$$P(k) = \frac{(k-1) \cdot (n-1)!}{(n-k+1)!} \cdot \frac{1}{n^{k-1}}.$$

To maximize $P(k)$, it should be greater than both of its neighbors $P(k-1)$ and $P(k+1)$:

$$P(k-1) < P(k) > P(k+1).$$

We compare $P(k)$ with $P(k+1)$:

$$\frac{(k-1) \cdot (n-1)!}{(n-k+1)!} \cdot \frac{1}{n^{k-1}} > \frac{k \cdot (n-1)!}{(n-k)!} \cdot \frac{1}{n^k}.$$

Simplifying this inequality:

$$(k-1) \cdot n > k \cdot (n-k+1).$$
$$k^2 - k - 200 > 0.$$

Solving the quadratic inequality:

$$k > \frac{1+\sqrt{1+800}}{2} \quad \text{or} \quad k < \frac{1-\sqrt{1+800}}{2}.$$

This results in:

$$k > 14.65 \quad \text{(since } k \text{ must be positive).}$$

Next, we compare $P(k)$ with $P(k-1)$:

$$\frac{(k-1) \cdot (n-1)!}{(n-k+1)!} \cdot \frac{1}{n^{k-1}} > \frac{(k-2) \cdot (n-1)!}{(n-k+2)!} \cdot \frac{1}{n^{k-2}}.$$

This simplifies to:

$$k^2 - 3k - 198 < 0.$$

Solving the quadratic inequality:

$$\frac{3-\sqrt{9+792}}{2} < k < \frac{3+\sqrt{9+792}}{2}.$$

This results in:

$$-12.65 < k < 15.65.$$

The intersection of both conditions gives us the most probable place for k:

$$k = 15.$$

6 Update Functions

The code for this problem can be found in the attached files named *P6-Update-Functions.py*.

Given a dataset stored in array A with n elements, we have computed the mean, median, and standard deviation. Now, we want to update these values when a new data value x_{new} is added, without recalculating them from scratch.

Updating the Mean

Formula Derivation

The mean of a dataset is given by:

$$\text{OldMean} = \frac{1}{n} \sum_{i=1}^{n} A[i]$$

When a new data value x_{new} is added, the new number of elements is $n+1$. The new mean (NewMean) can be calculated as:

$$\text{NewMean} = \frac{n \times \text{OldMean} + x_{\text{new}}}{n+1}$$

Python Function

```
def UpdateMean (OldMean , NewDataValue , n , A) :
    NewMean = (OldMean*n + NewDataValue)/(n+1)
    return NewMean
```

Updating the Median

Method for finding NewMedian

- Insert the new data value into the sorted array A.
- Recalculate the median based on the new number of elements $n+1$.
- The median is defined as the middle value in a sorted list. If the number of elements is even, the median is the average of the two middle numbers.

Python Function

```
def UpdateMedian (OldMedian , NewDataValue , n , A) :
    A.append(NewDataValue)
    A.sort()
    if n//2 == 0 :
        NewMedian = ( A[n//2] + A[n//2 + 1] )/ 2.0
    else :
```

```
        NewMedian = A[n//2 + 1]
    return NewMedian
```

Updating the Standard Deviation

Formula Derivation

The standard deviation σ of a dataset is:

$$\text{OldStd} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(A[i] - \text{OldMean})^2}$$

When a new value x_{new} is added, the new standard deviation (NewStd) can be derived using the following steps:

1. **Old Variance**: Compute the variance from the old standard deviation.

$$\text{OldVariance} = (\text{OldStd})^2$$

2. **New Variance**:

$$\text{NewVariance} = \frac{n \times \text{OldVariance} + (\text{OldMean})^2 - (n+1) \times (\text{NewMean})^2}{n+1}$$

3. **New Standard Deviation**:

$$\text{NewStd} = \sqrt{\text{NewVariance}}$$

Python Function

```
import math

def UpdateStd ( OldMean , OldStd , NewMean , NewDataValue , n , A) :
    OldVar = OldStd**2
    NewVar = ( n*( OldVar + OldMean*2 ) - (NewMean2)(n+1) + (NewDataValue)*2 ) / (n+1)
    NewStd = math.sqrt(NewVar)
    return NewStd
```

Updating the Histogram

Explanation

To update the histogram when a new value is added:

- **Increment the appropriate bin**: Identify the bin in which the new data value x_{new} falls. Increase the count for that bin by 1.
- **Recompute the bin boundaries if necessary**: If you're using dynamic bin sizes or boundaries based on the data range, you might need to recompute bin boundaries if the new value is outside the current range.

7 Plots

We have generated plots whose codes can be found in the attached files named : (libraries used are listed)

- ***P7-violin.ipynb***- numpy, matplotlib, seaborn, pandas
- ***P7-pareto.ipynb***- numpy, matplotlib
- ***P7-coxcomb.ipynb***- numpy, matplotlib
- ***P7-waterfall.ipynb***- numpy, matplotlib

7.1 Violin Plot

Violin plots display the probability density of data across different values, making them particularly useful for visualizing distributions that may be multimodal (i.e., distributions with more than one peak). They are commonly used to compare the distribution of numerical data between multiple groups. Violin plots are effective in visualizing data symmetry or skewness, as well as highlighting the presence of outliers and differences in distribution patterns between groups.

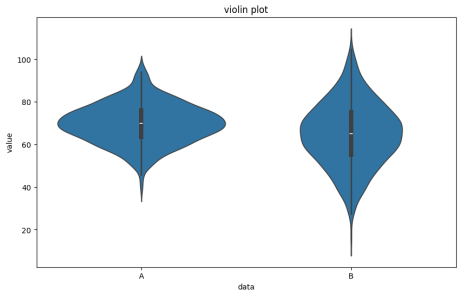

Figure 1: Generated Violin plot

7.2 Pareto Chart

A Pareto chart is useful for identifying the most significant factors within a dataset. This helps to prioritize issues that require attention. They are commonly used in quality control processes, particularly in methodologies like Six Sigma. By visualizing which categories contribute the most to a specific outcome or issue, Pareto charts make it easier to allocate resources and efforts toward the factors that will have the greatest impact on improvement.

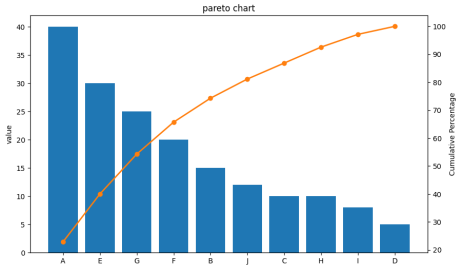

Figure 2: Generated Pareto Chart

7.3 Coxcomb Chart

Coxcomb chart was famously used by Florence Nightingale to show mortality causes during the Crimean War. The length of each segment is proportional to the data value it represents. They are used in visualizing cyclical data such as monthly or seasonal variations and for comparing different categories over space.

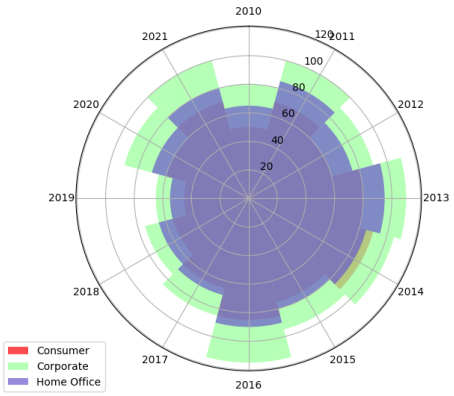

Figure 3: Generated Coxcomb Chart

7.4 Waterfall Plot

A waterfall plot is used to visualize the cumulative effect of sequentially introduced positive or negative values. It starts from an initial value and visualizes how incremental positive or negative changes affect the final total value. It is useful for showing stepwise changes in data. They are useful in Financial reporting for explaining the breakdown of profits or losses.

Figure 4: Generated Waterfall Plot

8 Monalisa

The Python code for this problem can be found in the attached file named **P8-Monalisa.ipynb**.
The image of Monalisa is downloaded from wikipedia.
Libraries used are-

- numpy
- matplotlib