Statistics Software Lab Report - 6

Name of the Student: Shatansh Patnaik
Roll No: 20MA20067

IIT Kharagpur
Statistics Software Lab

Calculation of Mean, Median, Mode, Quantiles, Trimmed Sample Mean and Winsorized Sample Mean

Mean:

The mean, also known as the average, is a measure of central tendency that represents the typical value in a dataset. It is calculated by summing up all the values in the dataset and dividing by the total number of values. The mean provides a balanced representation of the data's central value.

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n}$$

Median:

The median is another measure of central tendency that represents the middle value of a dataset when the values are arranged in ascending order. Unlike the mean, the median is not influenced by extreme values or outliers, making it a robust measure of central tendency.

$$\text{Median} = \begin{cases} x_{\frac{n+1}{2}} & \text{if } n \text{ is odd} \\ \frac{1}{2}(x_{\frac{n}{2}} + x_{\frac{n}{2}+1}) & \text{if } n \text{ is even} \end{cases}$$

Mode:

The mode is the value that occurs most frequently in the dataset. Unlike the mean and median, the mode can be non-unique, meaning there can be more than one mode in a dataset.

$$\text{Mode} = \text{value(s) with the highest frequency}$$

Quantiles:

Quantiles divide a dataset into equal portions. The median is the 50th percentile, while quartiles divide the dataset into quarters. Quantiles provide insights into the distribution of the data across various percentiles, helping to understand the spread and variability of the dataset.

$$Q_p = x_{\lfloor np \rfloor + 1} + (np - \lfloor np \rfloor) \times (x_{\lfloor np \rfloor + 1} - x_{\lfloor np \rfloor})$$

where p is the desired percentile.

Trimmed Sample Mean:

The trimmed sample mean is a variation of the mean that reduces the impact of outliers by removing a certain percentage of the highest and lowest values from the dataset. This method is useful when extreme values significantly affect the mean, providing a more robust measure of central tendency.

$$\text{Trimmed Sample Mean} = \frac{\sum_{i=k+1}^{n-k} x_i}{n - 2k}$$

where k is the number of values trimmed from each end.

Winsorized Sample Mean:

The Winsorized sample mean is calculated by replacing extreme values in the dataset with the nearest non-extreme values and then calculating the mean of the adjusted dataset.

$$\text{Winsorized Sample Mean} = \frac{\sum_{i=1}^{n} y_i}{n}$$

Where y_i are the adjusted values after applying the Winsorization process to the original dataset x_i.

In the Winsorization process, extreme values are replaced with the nearest non-extreme values based on a specified trimming percentage p. Let k be the number of extreme values trimmed from each end of the dataset.

The adjusted values y_i are calculated as follows:

$$y_i = \begin{cases} x_{(k+1)} & \text{if } x_i < x_{(k+1)} \\ x_i & \text{if } x_{(k+1)} \leq x_i \leq x_{(n-k)} \\ x_{(n-k)} & \text{if } x_i > x_{(n-k)} \end{cases}$$

Where $x_{(k+1)}$ and $x_{(n-k)}$ are the (k+1)th smallest and (n-k)th largest values in the dataset respectively, after it's been sorted.

This process ensures that extreme values are replaced with values that are closer to the center of the dataset, thereby reducing their influence on the mean.

The above formulas can be implemented in code for Task-A as follows:

```r
# Task A Solution
givenData <- c(0.72, 0.92, 0.92, 1.43, 0.83, 0.48, 0.65, 0.78,
                  0.48, 0.96, 0.72, 0.48, 0.83, 0.49, 0.78, 0.96,
                  0.88, 1.03, 0.78, 1.12, 0.83, 0.78, 0.83, 1.06,
                  1.23, 0.18, 0.96, 1.18, 0.48, 0.55, 0.97, 1.21,
                  0.94, 0.38, 0.73, 0.65, 1.36, 0.47, 0.72, 0.77,
                  0.79, 1.26, 1.06, 0.90, 0.77, 0.35, 0.78, 0.77,
                  0.88, 1.20, 0.71, 0.95, 0.91, 0.64, 0.73, 1.09,
                  0.83, 0.78, 1.04, 1.33, 0.47, 0.16, 0.57, 0.65,
                  0.64, 0.65, 1.43, 0.63, 0.79, 1.00, 0.92, 0.45,
                  0.48, 0.79, 0.97, 0.57, 0.95, 1.12, 0.70, 1.05)

getClassIntervals <- seq(min(givenData), max(givenData), by = 0.20)
getFreq <- cut(givenData, breaks = getClassIntervals, include.lowest = TRUE)
getFreq  <- table(getFreq )

meanValue <- mean(givenData)
medianValue <- median(givenData)
quartiles <- quantile(givenData, probs = c(0.25, 0.5, 0.75))
modeValue <- names(sort(-table(givenData)))[1]
sdValue <- sd(givenData)
IQRValue <- IQR(givenData)

trimmedMean <- mean(givenData, trim = 0.05)
winsorizedMean <- mean(pmin(pmax(givenData, quantile(givenData, 0.05)),
    quantile(givenData, 0.95)))
```

```

printResults <- function (){
  cat("The Frequency Distribution is as follows:\n")
  cat(getFreq)
  cat("\nThe Mean is as follows:\n")
  cat(meanValue)
  cat("\nThe Median is as follows:\n")
  cat(medianValue)
  cat("\nThe Quartiles are as follows:\n")
  cat(quartiles)
  cat("\nThe Mode is as follows:\n")
  cat(modeValue)
  cat("\nThe Standard Deviation is as follows:\n")
  cat(sdValue)
  cat("\nThe Interquartile Range: is as follows: \n")
  cat(IQRValue)
  cat("\nThe Trimmed Mean with 5% trimming is :\n")
  cat(trimmedMean)
  cat("\nThe Winsorized Mean with 5% Winsorizing:\n")
  cat(winsorizedMean)
}

printResults()
```

Calculations of Central, Non-Central Moments, Skewness and Kurtosis

Central Moments:

Central moments are statistical measures that describe the shape and variability of a distribution around its mean. They quantify how spread out the data values are from the mean. The rth central moment of a dataset x is calculated by taking each data point, subtracting the mean ($\bar{x}$), raising the result to the power of r, and then averaging the results. Central moments provide insights into the dispersion, symmetry, and tails of the distribution.

$$\mu_r = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^r$$

Noncentral Moments:

Noncentral moments are similar to central moments, but they measure the moments around a point other than the mean. They are useful for describing the distribution's shape relative to a specific reference point. The rth noncentral moment of a dataset x about a point a is calculated by taking each data point, subtracting the reference point (a), raising the result to the power of r, and then averaging the results.

$$\mu'_r = \frac{1}{n}\sum_{i=1}^{n}(x_i - a)^r$$

Skewness:

Skewness measures the asymmetry of the distribution. It indicates whether the data distribution is symmetric or skewed to the left or right. A skewness value of zero indicates a perfectly symmetric distribution. Positive skewness indicates a longer tail on the right side, while negative skewness indicates a longer tail on the left side. Skewness is calculated using the third central moment (μ_3) and the standard deviation (σ).

$$\text{Skewness} = \frac{\mu_3}{\sigma^3}$$

Kurtosis:

Kurtosis measures the peakedness or flatness of the distribution compared to a normal distribution. It indicates whether the data distribution has heavier or lighter tails than a normal distribution. Positive kurtosis indicates a sharper peak (leptokurtic), while negative kurtosis indicates a flatter peak (platykurtic). Kurtosis is calculated using the fourth central moment (μ_4) and the variance (σ^2).

$$\text{Kurtosis} = \frac{\mu_4}{\sigma^4} - 3$$

The above formulas can be implemented in code for Task-B as follows:

```r
# Task B Solution
newData <- c(9.3, 6.8, 9.8, 6.6, 4.3, 6.7, 6.4, 10.1, 8.9, 3.7,
             5.3, 6.5, 7.4, 8.3, 4.6, 7.9, 6.5, 5.1, 7.2, 8.7,
             7.9, 6.3, 2.7, 5.3, 8.8, 7.3, 9.0, 7.7, 8.4, 7.8,
             5.8, 6.4, 6.2, 5.8, 6.5, 6.0, 7.7, 5.0, 4.4, 4.7,
             5.4, 2.9, 4.0, 4.1, 4.1, 5.5, 3.1, 3.5, 5.4, 4.1,
             4.7, 6.2, 3.2, 2.7, 4.8, 2.6, 3.4, 6.2, 5.1, 4.0,
             5.0, 3.3, 2.4, 4.6, 2.8, 1.7, 0.9, 7.2, 9.9, 4.0,
             2.0, 2.0, 1.0, 3.2, 5.6, 3.4, 5.7, 7.0, 4.3, 3.4,
             3.0, 4.4, 2.0, 5.8, 1.5, 5.1, 5.0, 8.8, 4.0, 6.1,
             5.6, 5.4, 8.3, 8.8, 10.0, 4.8, 3.6, 2.5, 5.3, 2.2,
             4.1, 5.0)

classInts <- seq(floor(min(newData)), ceiling(max(newData)), by = 1)
freqTable <- cut(newData, breaks = classInts, include.lowest = TRUE)
freqTable <- table(freqTable)

generateMoments <- function(data, order) {
  n <- length(data)
  meanValue <- mean(data)
  centralMoments <- sum((data - meanValue)^order) / n
  noncentralMoments <- sum((data - meanValue)^order) / n
  return(list(central = centralMoments, noncentral = noncentralMoments))
}

firstMoment <- generateMoments(newData, 1)
secondMoment <- generateMoments(newData, 2)
thirdMoment <- generateMoments(newData, 3)
```

```r
fourthMoment <- generateMoments(newData, 4)

skewness <- thirdMoment$central / (secondMoment$central^(3/2))
kurtosis <- fourthMoment$central / (secondMoment$central^2) - 3

printNewResults <- function() {
  cat("\nThe Frequency Distribution is as follows: \n")
  cat(freqTable)
  cat("\nThe First Moment (Mean) is as follows: \n")
  cat(firstMoment$noncentral)
  cat("\n The Second Moment (Variance) is as follows: \n")
  cat(secondMoment$central)
  cat("\n The Third Moment (Skewness) is as follows: \n")
  cat(skewness)
  cat("\nThe Fourth Moment (Kurtosis) is as follows: \n")
  cat(kurtosis)
}

printNewResults()
```

Generation of Frequency Histogram, Frequency Polygon, Frequency Curve and Cummulative Frequency Curve (Ogive)

Frequency Histogram:

A frequency histogram is a graphical representation of the frequency distribution of a dataset. It consists of bars where the height of each bar represents the frequency of values falling within a specific interval.

From a frequency histogram, we can infer the shape of the distribution, the central tendency, spread, and variability of the data, as well as any outliers or unusual patterns.

Frequency Polygon:

A frequency polygon is a line graph that displays the frequencies of different values or intervals in a dataset. It is created by joining the midpoints of the tops of the bars in a frequency histogram with straight lines.

From a frequency polygon, we can infer the overall trend or shape of the distribution, the central tendency, and any patterns or anomalies present in the data.

Frequency Curve:

A frequency curve is a smooth curve that represents the frequency distribution of a dataset. It is obtained by joining the midpoints of the tops of the bars in a frequency histogram with a smooth curve.

From a frequency curve, we can infer the shape of the distribution, the central tendency, spread, and variability of the data, as well as any patterns or trends present.

Cumulative Frequency Curve (Ogive):

A cumulative frequency curve, also known as an ogive, is a line graph that represents the cumulative frequencies of values or intervals in a dataset. It is created by plotting the cumulative frequency against the upper class boundary of each interval.

From a cumulative frequency curve, we can infer the cumulative distribution of the data, identify percentiles or cumulative probabilities, and compare distributions.

The following is the R code for the generation of the above plots:

```r
# Task-C Solution
classIntsA <- seq(floor(min(givenData)), ceiling(max(givenData)) + 0.2, by =
    0.20)
freqTableA <- cut(givenData, breaks = classIntsA, include.lowest = TRUE)
freqTableA <- table(freqTableA)
midsA <- classIntsA[-length(classIntsA)] + diff(classIntsA) / 2
densityA <- density(givenData)
upperLimsA <- classIntsA + 0.20
upperLimsA <- upperLimsA[-length(upperLimsA)]

classIntsB <- seq(floor(min(newData)), ceiling(max(newData)) + 1, by = 1)
freqTableB <- cut(newData, breaks = classIntsB, include.lowest = TRUE)
freqTableB <- table(freqTableB)
midsB <- classIntsB[-length(classIntsB)] + diff(classIntsB) / 2
densityB <- density(newData)
upperLimsB <- c(classIntsB[-1] + 1, max(classIntsB) + 1)
upperLimsB <- upperLimsB[-length(upperLimsB)]

par(mfrow=c(2, 2))

hist(givenData, breaks = classIntsA, main = "Frequency Histogram (Task A)",
    xlab = "Iron Solution Index", ylab = "Frequency", col="red")
plot(midsA, freqTableA, type = "l", main = "Frequency Polygon (Task A)",
    xlab = "Iron Solution Index", ylab = "Frequency" , col="blue")
plot(densityA, main = "Frequency Curve (Task A)", xlab = "Iron Solution
    Index", ylab = "Density", col="pink")
plot(upperLimsA, cumsum(freqTableA), type = "s", main = "Cumulative
    Frequency Curve (Task A)", xlab = "Iron Solution Index", ylab = "
    Cumulative Frequency", col="green")

hist(newData, breaks = classIntsB, main = "Frequency Histogram (Task B)",
    xlab = "Radioactive newData", ylab = "Frequency", col="lightblue")
plot(midsB, freqTableB, type = "l", main = "Frequency Polygon (Task B)",
    xlab = "Radioactive newData", ylab = "Frequency", col="red")
plot(densityB, main = "Frequency Curve (Task B)", xlab = "Radioactive
    newData", ylab = "Density", col="pink")
plot(upperLimsB, cumsum(freqTableB), type = "s", main = "Cumulative
    Frequency Curve (Task B)", xlab = "Radioactive newData", ylab = "
    Cumulative Frequency", col="green")

par(mfrow=c(1, 1))
```

Finally we need to prepare frequency histogram, frequency polygon, frequency curve and cumulative frequency curve (ogive) from the classified data obtained in Task-A and Task-B.

Task-A

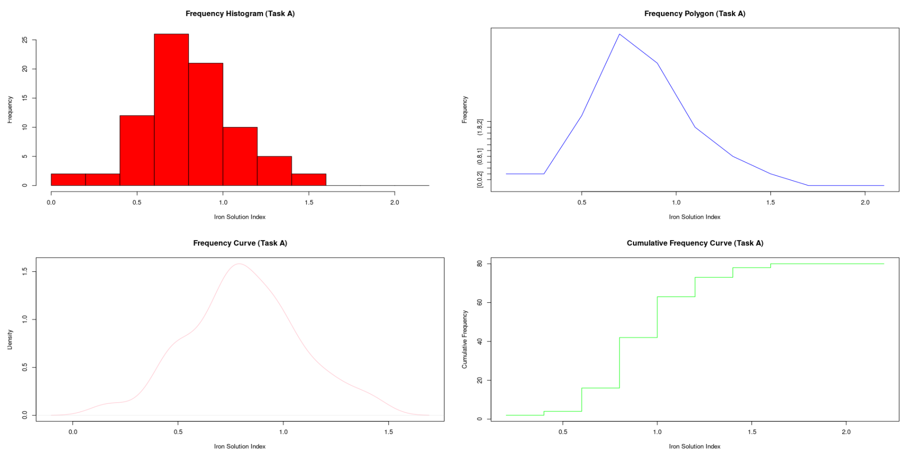

Figure 1: Task-A Plots

Task-B

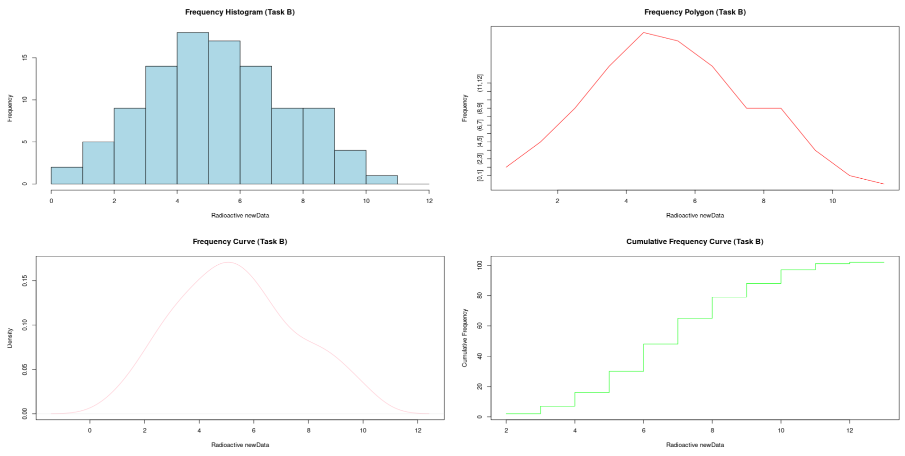

Figure 2: Task-B Plots