VIRGINIA COMMONWEALTH UNIVERSITY

Statistical analysis and modelling (SCMA 632)

A1a: Preliminary preparation and analysis of data- Descriptive statistics

MICAH ASHADEEP EMMANUEL

V01101166

Date of Submission: 16-06-2024

CONTENTS

INTRODUCTION

The focus of this study is on the state of WestBengal, utilizing data from the NSSO to identify the top and bottom three consuming districts. This involves manipulating and cleaning a dataset to prepare for analysis. We have gathered comprehensive consumption-related information, covering rural and urban sectors, along with district-wise variations. The dataset has been imported into R, a robust statistical programming language known for its efficacy in managing and analysing extensive datasets. Our objectives encompass several key tasks: identifying and addressing missing values, handling outliers, standardizing district and sector names, summarizing consumption data regionally and by district, and evaluating the significance of mean differences. The outcomes of this study aim to provide insights valuable to policymakers and stakeholders, enabling targeted interventions and supporting balanced development throughout the state.

.

By understanding how spending patterns vary across WestBengal, policymakers, businesses, and researchers can make better decisions. This could involve allocating resources more effectively, targeting specific markets with relevant products or services, and developing programs tailored to the needs of different regions within the state

OBJECTIVES

Using the provided data, you must create an Excel file with the state assigned to you. Name it and then import it into Excel. Subset the variables assigned to you and perform the following operations using the software. You must discuss your results.

• Check if there are any missing values in the data, identity them, and if there are, replace them with the mean of the variable

• Check for outliers, describe your test's outcome, and make suitable amendments.

• Rename the districts and sectors, viz., rural and urban.

• Summarize the critical variables in the data set region-wise and district-wise and indicate the top and bottom three districts of consumption.

• Test whether the differences in the means are significant or not.

BUSINESS SIGNIFICANCE

The study on WestBengal holds significant business implications as it delves into consumption patterns across its districts, crucial for various stakeholders. By analyzing NSSO data, the study aims to identify the top and bottom consuming districts, providing insights into regional consumption disparities.For businesses, this analysis offers strategic advantages:

Market Segmentation: Understanding consumption variations helps businesses tailor their marketing and distribution strategies. They can target high-consuming districts more aggressively while adapting their offerings for low-consuming areas.

Resource Allocation: Insights into district-wise consumption patterns guide resource allocation decisions. Companies can optimize inventory levels, distribution networks, and sales efforts based on local demand trends.

New Market Opportunities: Identification of under-served areas presents opportunities for market expansion. Businesses can explore potential growth markets where consumption levels are rising or where there is a gap in supply.

Policy and Regulatory Impact: Findings can inform policy decisions, influencing regulatory frameworks and economic policies that affect business operations and market dynamics in WestBengal.

Competitive Benchmarking: Benchmarking against consumption data from competitors can provide a comparative advantage. Businesses can gauge their market share and performance relative to peers in different districts.

In essence, the study's findings can empower businesses in WestBengal to make informed decisions, enhance market penetration strategies, and capitalize on emerging opportunities, thereby contributing to sustainable growth and competitive advantage in the region.

RESULTS AND INTERPRETATION

USING R

-Setting Working Directory and Loading Libraries and Reading and Filtering Data

```
setwd('C:\\Users\\HP\\Documents\\ns')

getwd()

install_and_load <- function(package)

library(readr)
```

library(dplyr)

data <- read_csv("NSSO68 new.csv")

WestBengal_data <- filter(data, state == "WestBengal")

-**Handling Missing Values:**

Calculates and prints missing values before and after imputation with column means for numeric columns.

WestBengal_data <- WestBengal_data %>%

  mutate(across(where(is.numeric), ~ ifelse(is.na(.), mean(., na.rm = TRUE), .)))

missing_values_after <- sapply(WestBengal_data, function(x) sum(is.na(x)))

print(missing_values_after)

- Handling Outliers:

Identifies outliers using the IQR method and stores them in `outliers`.

This line selects only those columns from WestBengal_data that contain numeric data, excluding non-numeric columns.

missing_values <- sapply(WestBengal_data, function(x) sum(is.na(x)))

print(missing_values)

WestBengal_data <- WestBengal_data %>%

library(ggplot2)

WestBengal_data <- WestBengal_data %>%

  mutate(across(where(is.numeric), ~ ifelse(is.na(.), mean(., na.rm = TRUE), .)))

missing_values_after <- sapply(WestBengal_data, function(x) sum(is.na(x)))

print(missing_values_after)

-Handling Outliers

numeric_columns <- WestBengal_data %>% select_if(is.numeric)

```
outliers <- list()


for(col in colnames(numeric_columns)) {
  Q1 <- quantile(numeric_columns[[col]], 0.25, na.rm = TRUE)
  Q3 <- quantile(numeric_columns[[col]], 0.75, na.rm = TRUE)
  IQR <- Q3 - Q1


  lower_bound <- Q1 - 1.5 * IQR
  upper_bound <- Q3 + 1.5 * IQR


  outliers[[col]] <- numeric_columns %>%
    filter((.data[[col]] < lower_bound) | (.data[[col]] > upper_bound)) %>%
    select(col)
}
sapply(outliers, nrow)
```

-**Replacing Outliers with Median:**

Replaces outliers with the median value for each numeric column.

```
numeric_columns <- WestBengal_data %>% select_if(is.numeric)
outliers <- list()


for(col in colnames(numeric_columns)) {
  Q1 <- quantile(numeric_columns[[col]], 0.25, na.rm = TRUE)
  Q3 <- quantile(numeric_columns[[col]], 0.75, na.rm = TRUE)
  IQR <- Q3 - Q1
```

```
  lower_bound <- Q1 - 1.5 * IQR

  upper_bound <- Q3 + 1.5 * IQR



  outliers[[col]] <- numeric_columns %>%

    filter((.data[[col]] < lower_bound) | (.data[[col]] > upper_bound)) %>%

    select(col)

}

sapply(outliers, nrow)
```

-Replacing Outliers with Median:

his R code snippet iterates through each numeric column (col) in the WestBengal_data dataframe. For each column, it calculates the first quartile (Q1), third quartile (Q3), and interquartile range (IQR). Using these values, it defines lower (lower_bound) and upper (upper_bound) boundaries to identify outliers based on the IQR method. The median value (median_val) of the column is then computed and used to replace any outliers found—values outside the bounds defined by lower_bound and upper_bound are replaced with median_val, while values within the bounds remain unchanged. This process ensures that extreme values, which can distort statistical analyses, are adjusted to more representative values, promoting more reliable data insights**.**

```
for(col in colnames(numeric_columns)) {

  Q1 <- quantile(numeric_columns[[col]], 0.25, na.rm = TRUE)

  Q3 <- quantile(numeric_columns[[col]], 0.75, na.rm = TRUE)

  IQR <- Q3 - Q1



  lower_bound <- Q1 - 1.5 * IQR

  upper_bound <- Q3 + 1.5 * IQR



  median_val <- median(numeric_columns[[col]], na.rm = TRUE)



  WestBengal_data[[col]] <- ifelse(WestBengal_data[[col]] < lower_bound |
WestBengal_data[[col]] > upper_bound,
```

```
                    median_val,

                    WestBengal_data[[col]])

}
```

-**Data Transformation and Summary**:

ransforms district and sector columns using recode().

Computes summary statistics (avg_consumption and `total)

These R code lines transform `WestBengal_data` by recoding `district` and `sector` columns from numeric codes to descriptive labels ('1' = 'District1', '2' = 'District2', etc., and '1' = 'Rural', '2' = 'Urban'). Afterwards, `summary_data` aggregates `WestBengal_data` by `district` and `sector`, calculating average (`avg_consumption`) and total (`total_consumption`) consumption values while dropping grouping attributes (`drop`).

```
WestBengal_data <- WestBengal_data %>%

  mutate(

    district = recode(district,

                      '1' = 'District1',

                      '2' = 'District2',

                      '3' = 'District3'  # Continue this for all district codes

    ),

    sector = recode(sector,

                    '1' = 'Rural',

                    '2' = 'Urban')

  )

summary_data <- WestBengal_data %>%

  group_by(district, sector) %>%

  summarize(

    avg_consumption = mean(consumption, na.rm = TRUE),

    total_consumption = sum(consumption, na.rm = TRUE),
```

```
    .groups = 'drop'
  )
```

-Summary of Consumption Analysis and Comparison

`rural_consumption` and `urban_consumption` calculate total consumption (`total_consumption`) for rural and urban sectors from `WestBengal_data`, using `filter()` to subset data by sector and `pull()` to extract the `total_consumption` values. These variables can be used to compare consumption levels between rural and urban areas in WestBengal.

```
top_three <- summary_data %>%
  arrange(desc(avg_consumption)) %>%
  head(3)

bottom_three <- summary_data %>%
  arrange(avg_consumption) %>%
  head(3)
print("Top Three Districts of Consumption:")
print(top_three)

print("Bottom Three Districts of Consumption:")
print(bottom_three)
rural_consumption <- WestBengal_data %>%
  filter(sector == "Rural") %>%
  pull(total_consumption)

urban_consumption <- WestBengal_data %>%
  filter(sector == "Urban") %>%
```

```
  pull(total_consumption)
```

-installed the package

```
install.packages(BSDA)

library(BSDA)
```

-**Statistical Comparison of Urban and Rural Consumption**

sigma_rural and sigma_urban represent the standard deviations of consumption in rural and urban sectors, respectively.
The script performs a hypothesis test (not explicitly shown) and checks if the p-value (z_test_result$p.value) is less than 0.05.
Depending on the p-value result, it prints either that there is a significant difference between mean consumptions of urban and rural areas or that there is no significant difference.
This analysis helps determine whether the observed differences in consumption between urban and rural sectors are statistically significant based on the chosen significance level (0.05 in this case).

```
sigma_rural <- 2.56
sigma_urban <- 2.34
if (z_test_result$p.value < 0.05) {
  cat("P value is <", 0.05, ", Therefore we reject the null hypothesis.\n")
  cat("There is a significant difference between mean consumptions of urban and rural.\n")
} else {
  cat("P value is >=", 0.05, ", Therefore we fail to reject the null hypothesis.\n")
  cat("There is no significant difference between mean consumptions of urban and rural.\n")
}
```

-**Print Z-test Result**
This line of code prints the result of a z-test stored in the variable `z_test_result`.
The z-test is typically used to assess whether there is a significant difference between means of two populations based on their standard deviations and sample sizes.
The output usually includes statistics such as the z-score, p-value, and possibly confidence intervals.
Printing `z_test_result` allows for inspection and interpretation of the statistical test outcome, providing insights into the significance of the differences observed in the data analysis.

```
print(z_test_result)
```

RESULTS AND INTERPRETATION

USING Python

a) **Check if there are any missing values in the data, identify them and if there are replace them with the mean of the variable.**

```
b) WB_new.isnull().sum().sort_values(ascending = False)
c) Meals_At_Home      18
d) state              0
e) District           0
f) Sector             0
g) Region             0
h) State_Region       0
i) ricetotal_q        0
j) wheattotal_q       0
k) moong_q            0
l) Milktotal_q        0
m) chicken_q          0
n) bread_q            0
o) foodtotal_q        0
p) Beveragestotal_v   0
q) dtype: int64
```

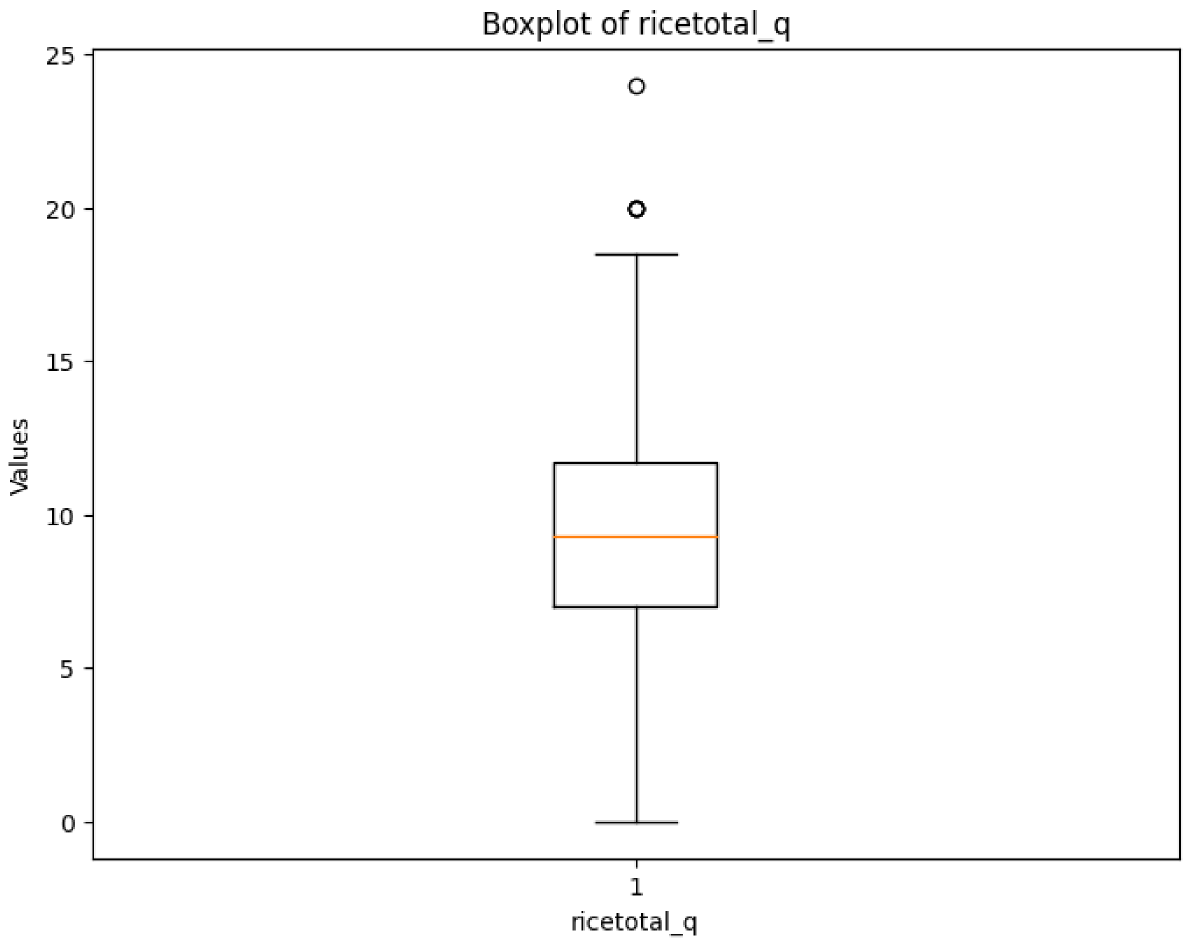
Interpretation: From the selected variables, after sorting the data for the state of Andhra Pradesh, it is seen that only the column ‘Meals_At_Home has 18 missing variables. Since missing values in the dataset can be problematic as they lead to incomplete or biased analyses, hindering the accuracy of results and potentially skewing interpretations and decision-making processes.Therefore we replace the missing values with the mean of the variable using following code

```
WB_clean = WB_new.copy()
WB_clean.loc[:, 'Meals_At_Home'] =
WB_clean['Meals_At_Home'].fillna(WB_new['Meals_At_Home'].mean())
WB_clean.isnull().any()
state           False
District        False
Sector          False
Region          False
State_Region    False
```

```
ricetotal_q        False
wheattotal_q       False
moong_q            False
Milktotal_q        False
chicken_q          False
bread_q            False
foodtotal_q        False
Beveragestotal_v   False
Meals_At_Home      False
dtype: bool
```

Check for outliers and describe the outcome of your test and make suitable amendments. Boxplots can be used to find outliers in the dataset. Boxplots visually reveal outliers in a dataset by displaying individual points located beyond the whiskers of the boxplot. #Checking for outliers Plotting the boxplot to visualize outliers. Code and Result:

```
import matplotlib.pyplot as plt
# Assuming WB_clean is your DataFrame
plt.figure(figsize=(8, 6))
plt.boxplot(WB_clean['ricetotal_q'])
plt.xlabel('ricetotal_q')
plt.ylabel('Values')
plt.title('Boxplot of ricetotal_q')
plt.show()
```



Interpretation: From the boxplot above, which is a visual representation of the variable 'ricepds_v' shows that there is an outlier. Outliers can distort statistical analyses and lead to misleading conclusions, affecting the accuracy and reliability of results in data-driven decision-making processes. Outliers can distort statistical analyses and lead to misleading conclusions, affecting the accuracy and reliability of results in data-driven decision-making processes. The outliers can be removed using the following code.

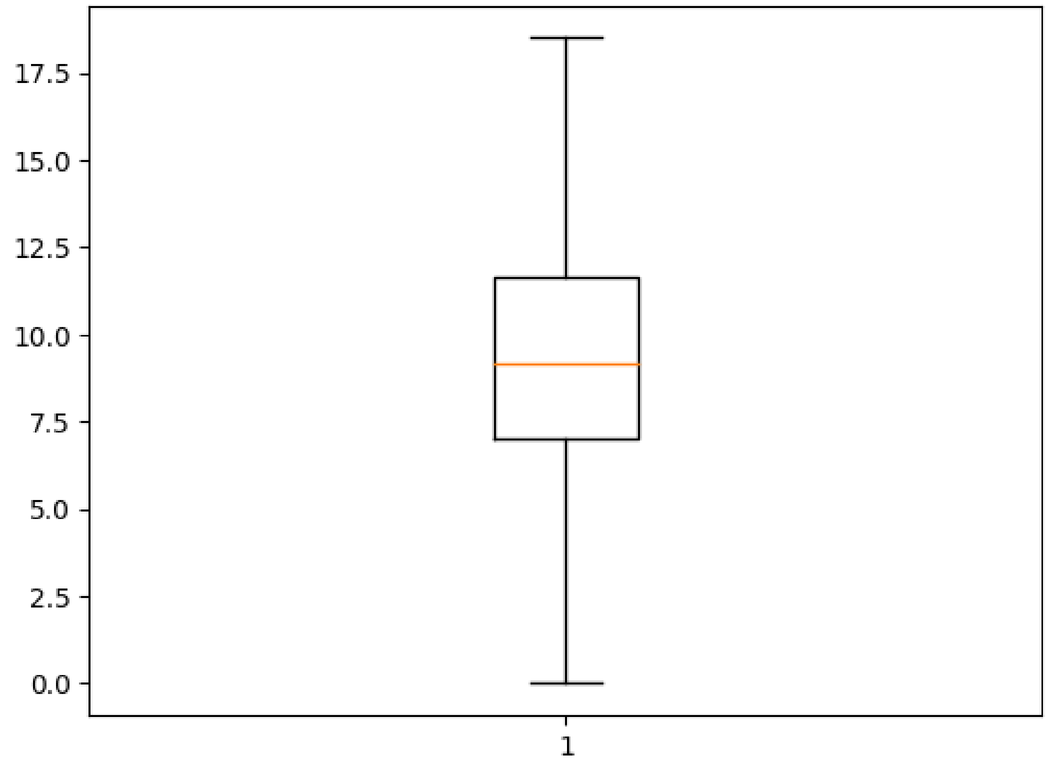
Code and results: Setting quartile ranges to remove outliers

```
rice1 = WB_clean['ricetotal_q'].quantile(0.25)
rice2 = WB_clean['ricetotal_q'].quantile(0.75)
iqr_rice = rice2-rice1
up_limit = rice2 + 1.5*iqr_rice
low_limit = rice1 - 1.5*iqr_rice
```

```
WB_clean=WB_new[(WB_new['ricetotal_q']<=up_limit)&(WB_new['ricetotal_q'
]>=low_limit)]
```

```
plt.boxplot(WB_clean['ricetotal_q'])
```

```
{'whiskers': [<matplotlib.lines.Line2D at 0x7893c55bf580>,
<matplotlib.lines.Line2D at 0x7893c55bfeb0>], 'caps':
[<matplotlib.lines.Line2D at 0x7893c55bfd30>, <matplotlib.lines.Line2D
at 0x7893c55bf160>], 'boxes': [<matplotlib.lines.Line2D at
0x7893c55bcd60>], 'medians': [<matplotlib.lines.Line2D at
0x7893c55bd720>], 'fliers': [<matplotlib.lines.Line2D at
0x7893c55bc550>], 'means': []}
```



Interpretation: Interpreting quartile ranges allows for outlier detection and removal. By calculating the interquartile range (IQR) as the difference between the upper and lower quartiles, data points beyond 1.5 times the IQR from either quartile are identified as outliers and can be excluded or treated to ensure the robustness of the analysis. In the similar way the outliers in all other variables can be removed.

c) Summarize the critical variables in the data set region wise and district wise and indicate the top three districts and the bottom three districts of consumption. By summarizing the critical variables as total consumption we can estimate the top 3 and bottom 3 consuming districts. 8 Code and Result:

```
WB_clean.groupby('Region').agg({'total_consumption':['std','mean','max','min']})
```

	total_consumption			
	std	mean	max	min
Region				
1.0	32.414973	62.180210	187.751465	24.98023
2.0	24.954397	49.355884	211.533867	0.00000
3.0	38.631152	58.955174	299.438431	0.00000

```
WB_clean.groupby('District').agg({'total_consumption':['std','mean','max','min']})
```

total_consumptionstdmeanmaxminDistrict151.28086857.415027299.43843121.540365241.9606845
9.031612175.2504200.000000334.52723762.380691164.27546525.250133425.61055557.2018
58149.03374830.150168542.08640862.120997220.0171000.00000067.71858353.41055060.77
524543.325229717.51865721.59208647.4502500.00000083.54184451.32287556.04040845.86
7025933.72024661.566050134.37077031.8752251028.18275461.607701118.38383331.800186
1115.52204953.38371379.00059737.0000001216.91489152.32742799.47869624.9802301313.7
1647665.20187981.58233438.1860731429.24544869.828054115.30968337.3000301513.78265
865.40658587.30130043.5002421639.19332079.661913148.37560544.1002801748.38951269.
546444166.57716729.1626381850.57188867.490440187.75146531.4253621913.55987744.205
41977.10026037.441812206.97784050.38312658.84397537.1504352122.44237055.097293120
.31767330.509182226.71692744.84996858.18137633.6261472318.85356149.86295999.52536
222.8824402419.65988648.303110109.36687720.0000002525.23473557.527052117.00039025.
8168652634.21614656.553452211.53386729.7001402712.78523242.25993889.02544424.2501
672825.99137756.331454137.79616222.5504022929.90704336.516684108.0132600.00000030
32.44451947.905478155.9258060.000000

```
total_consumption_by_districtcode=WB_clean.groupby('District')['total_consumption'].sum()
total_consumption_by_districtcode.sort_values(ascending=False).head(3)
WB_clean.loc[:,"District"] = WB_clean.loc[:,"District"].replace({5: "Hyderabad and Rangar", 6: "Rangareddi", 23: "Chittoor"})
total_consumption_by_districtname=WB_clean.groupby('District')['total_consumption'].sum()
total_consumption_by_districtname.sort_values(ascending=False).head(3)
```

```
District
Rangar    3975.743824
kadapa          1933.801424
Krishna         1889.011584
Name: total_consumption, dtype: float64
```

e) Test whether the differences in the means are significant or not.

```
if cons_rural.empty or cons_urban.empty:
    print("Warning: One or both of the consumption Series are empty. Z-test cannot be performed.")
else:
    z_statistic, p_value = stests.ztest(cons_rural, cons_urban)
    # Print the z-score and p-value
    print("Z-Score:", z_statistic)
    print("P-Value:", p_value)
```

```
z_statistic, p_value = stests.ztest(cons_rural, cons_urban)
# Print the z-score and p-value
print("Z-Score:", z_statistic)
print("P-Value:", p_value)
```

```
Z-Score: 12.52569222339867
P-Value: 5.4017986377956026e-36
```