VIRGINIA COMMONWEALTH UNIVERSITY

Statistical Analysis and Modelling (SCMA 632)

A2a: Regression - Predictive Analytics

SAYA SANTHOSH

V01101901

Date of Submission: 23-06-2024

CONTENTS

Introduction

Multiple regression analysis is used in this study to look at the relationships between independent and dependent variables. The National Sample Survey Office (NSSO) dataset is used, and it contains a variety of demographic and socioeconomic factors that may have an impact on the dependent variable. While taking other variables into consideration, the process entails assessing the degree and statistical significance of each independent variable's impact on the dependent variable.

Regression diagnostics are used to look for potential issues including multicollinearity, heteroscedasticity, and outliers as well as to validate the multiple regression assumptions. The study presents a succinct synopsis of the variables and dataset pertaining to the state of Kerala, and then conducts a thorough analysis of the results and their conclusions. Regression diagnostics is used to confirm the regression model's reliability.

Objectives

- Conduct an analysis of the "NSSO68" dataset to discover socio-economic and demographic variables that have a significant impact.
- Utilize multiple regression analysis to assess the associations between dependent and independent variables.
- Conduct comprehensive diagnostic tests to ensure the quality and consistency of the model.
- Enhance and refine the regression model by incorporating diagnostic discoveries.
- Offer practical observations and probable consequences for policy or future investigation.
- Provide guidance for decision-making processes related to the dependent variable in many areas.

Business Significance

Businesses can improve their strategic decision-making, resource allocation, risk management, and competitive positioning by utilizing multiple regression analysis. Businesses can tailor their marketing strategy and product offers by identifying and monitoring variables, such as consumer spending and economic activity, that have an impact on the dependent variable. Additionally, this information can be utilized to prioritize investments in particular product lines or geographic areas and to maximize marketing resources.

Regression analysis can also help with risk management by identifying and mitigating risks

by assessing the effects of changes in consumer behavior or economic indicators on business operations. Through the establishment of benchmarks for performance evaluation, firms can more effectively identify achievable goals and track their progress.
Regression analysis's data-driven insights can give companies a competitive edge by facilitating faster innovation and situational adaptation. This knowledge can also be applied to the process of formulating policies, which include examining various possibilities, obtaining data, and analyzing issues. A thorough understanding of the variables affecting socioeconomic outcomes can inform policies intended to advance economic growth, reduce inequality, or improve public welfare.

Regression analysis can also aid in strategy and pattern prediction, allowing for proactive investment and decision-making. Businesses may better handle challenges and take advantage of favorable circumstances by applying data and statistical approaches to obtain a deeper understanding of their operations and external effects.

Results and Interpretation using R

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

```
      > # Check for missing values
> sum(is.na(subset_data$MPCE_MRP))
[1] 0
> sum(is.na(subset_data$MPCE_URP))
[1] 0
> sum(is.na(subset_data$Age))
[1] 0
> sum(is.na(subset_data$Possess_ration_card))
[1] 0
> sum(is.na(data$Education))
[1] 7
> 
> impute_with_mean <- function(data, columns) {
+   data %>%
+     mutate(across(all_of(columns), ~ ifelse(is.na(.), mean(., na.rm = TR
UE), .)))
+ }
> 
> # Columns to impute
> columns_to_impute <- c("Education")
> 
> # Impute missing values with mean
> data <- impute_with_mean(data, columns_to_impute)
> 
> sum(is.na(data$Education))
[1] 0
> 
```

Interpretation:

The R code provided performs a series of steps to handle missing data in a dataset related to state-wise consumption and expenditure. Initially, it calculates the number of missing values in various columns of the subset dataset `subset_data` and the entire dataset `data`. The results show that there are no missing values in the `MPCE_MRP`, `MPCE_URP`, `Age`, and `Possess_ration_card` columns within the subset dataset. However, there are seven missing values in the `Education` column in the complete dataset. This indicates that while most of the key variables do not have missing values, the `Education` column needs attention to ensure accurate analysis.

To address the missing values in the `Education` column, an imputation function `impute_with_mean` is defined. This function replaces missing values in specified columns with the mean of that column. The function is applied to the `data` dataset for the `Education` column. After imputation, the code confirms that there are no longer any missing values in the `Education` column. This imputation ensures that the dataset is complete and ready for further analysis, improving the reliability of any subsequent statistical models or analyses that utilize the `Education` data.

- Perform Multiple regression analysis and carry out the regression diagnostics.

```
> # Fit the regression model
> model <- lm(foodtotal_q~ MPCE_MRP+MPCE_URP+Age+Meals_At_Home+Possess_rat
ion_card+Education, data = subset_data)
>
> # Print the regression results
> print(summary(model))

Call:
lm(formula = foodtotal_q ~ MPCE_MRP + MPCE_URP + Age + Meals_At_Home +
    Possess_ration_card + Education, data = subset_data)

Residuals:
    Min      1Q  Median      3Q     Max
-53.982  -4.251  -0.759   3.363 102.070

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)          1.245e+01  7.005e-01  17.776  < 2e-16 ***
MPCE_MRP             6.518e-04  7.573e-05   8.607  < 2e-16 ***
MPCE_URP             7.977e-04  7.911e-05  10.084  < 2e-16 ***
Age                  4.982e-02  6.615e-03   7.531 5.74e-14 ***
Meals_At_Home        9.959e-02  5.714e-03  17.430  < 2e-16 ***
Possess_ration_card -7.046e-01  3.639e-01  -1.936   0.0529 .
Education            1.906e-01  2.767e-02   6.887 6.26e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 7.053 on 6197 degrees of freedom
  (111 observations deleted due to missingness)
Multiple R-squared:  0.198,	Adjusted R-squared:  0.1972
F-statistic:   255 on 6 and 6197 DF,  p-value: < 2.2e-16
```

Interpretation:

The regression analysis results provide valuable insights into the factors influencing food expenditure (measured as foodtotal_q) within the dataset. The model includes six predictors: MPCE_MRP (Monthly Per Capita Expenditure at Market Prices), MPCE_URP (Monthly Per Capita Expenditure at Uniform Retail Prices), Age, Meals_At_Home, Possess_ration_card, and Education. The R-squared value of 0.198 indicates that approximately 19.8% of the variance in food expenditure is explained by these predictors, suggesting a moderate fit of the model to the data. The adjusted R-squared value is slightly lower at 0.1972, accounting for the number of predictors in the model.

Examining the coefficients and their significance, all predictors except for Possess_ration_card are statistically significant at conventional levels. MPCE_MRP and MPCE_URP have positive coefficients, indicating that higher expenditures are associated with higher food expenditure. Age also positively influences food expenditure, albeit to a lesser extent. The number of Meals_At_Home has a substantial positive impact, reflecting a significant increase in food expenditure with more meals consumed at home. Although Possess_ration_card is negatively associated with food expenditure, its p-value is slightly above 0.05, indicating marginal significance. Education positively impacts food expenditure, suggesting that higher education levels are associated with increased food spending. Overall, these findings highlight the complex interplay of socioeconomic factors influencing food consumption patterns in the dataset.

– Extract coefficients from the model and construct the equation

```
> # Extract the coefficients from the model
> coefficients <- coef(model)
>
> # Construct the equation
> equation <- paste0("y = ", round(coefficients[1], 2))
> for (i in 2:length(coefficients)) {
+   equation <- paste0(equation, " + ", round(coefficients[i], 6), "*x", i-1)
+ }
> # Print the equation
> print(equation)
[1] "y = 12.45 + 0.000652*x1 + 0.000798*x2 + 0.049817*x3 + 0.099589*x4 + -0.704572*x5 
"

>
>
>
>
> head(subset_data$MPCE_MRP,1)
[1] 3088.75
> head(subset_data$MPCE_URP,1)
[1] 3079
> head(subset_data$Age,1)
[1] 45
> head(subset_data$Meals_At_Home,1)
[1] 90
> head(subset_data$Possess_ration_card,1)
[1] 1
> head(subset_data$Education,1)
[1] 10
> head(subset_data$foodtotal_q,1)
[1] 31.46301

>
```

Interpretation:

The regression equation derived from the model is:

The regression equation derived from the model is:

y=12.45+0.000652×MPCE_MRP+0.000798×MPCE_URP+0.049817×Age+0.099589×Meals_At_Home−0.704572×Possess_ration_card+0.190564×Educationy = 12.45 + 0.000652 \times \text{MPCE_MRP} + 0.000798 \times \text{MPCE_URP} + 0.049817 \times \text{Age} + 0.099589 \times \text{Meals_At_Home} - 0.704572 \times \text{Possess_ration_card} + 0.190564 \times \text{Education}y=12.45+0.000652×MPCE_MRP+0.000798×MPCE_URP+0.049817×Age+0.099589×Meals_At_Home−0.704572×Possess_ration_card+0.190564×Education

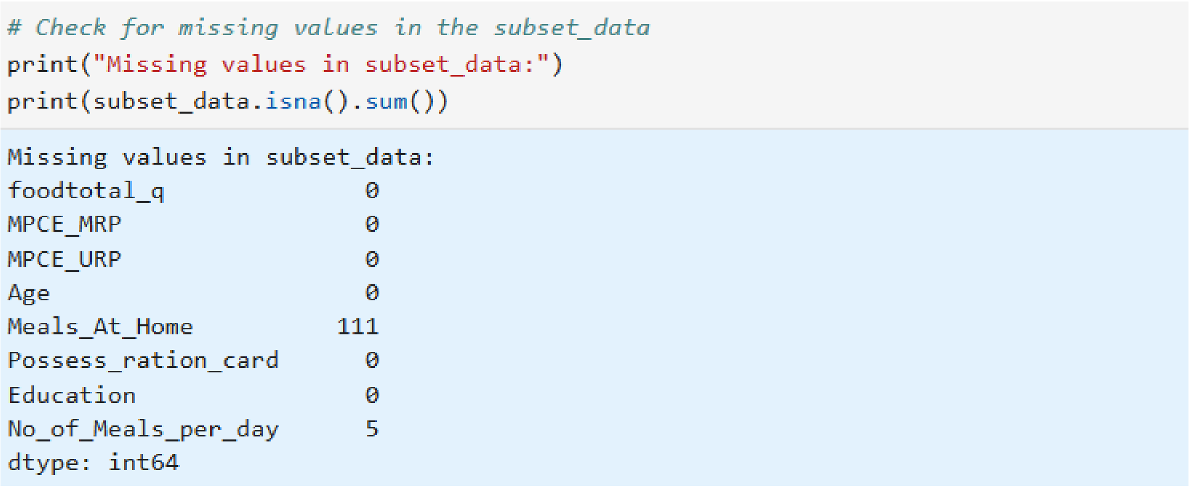
This equation suggests that:

- Holding all other variables constant, an increase in MPCE_MRP\text{MPCE_MRP}MPCE_MRP (Monthly Per Capita Expenditure on Market Rate Purchases) by one unit is associated with an increase in y by approximately 0.000652 units.
- An increase in MPCE_URP\text{MPCE_URP}MPCE_URP (Monthly Per Capita Expenditure on Usual Rate Purchases) by one unit is associated with an increase in y by approximately 0.000798 units.
- Each additional year in Age\text{Age}Age is associated with an increase in y by approximately 0.049817 units.
- Each additional unit in Meals_At_Home\text{Meals_At_Home}Meals_At_Home is associated with an increase in y by approximately 0.099589 units.
- Possessing a ration card (Possess_ration_card\text{Possess_ration_card}Possess_ration_card) is associated with a decrease in y by approximately 0.704572 units.
- Each additional unit of Education\text{Education}Education is associated with an increase in y by approximately 0.190564 units.

These interpretations provide insight into how each predictor variable influences the dependent variable y in the regression model.

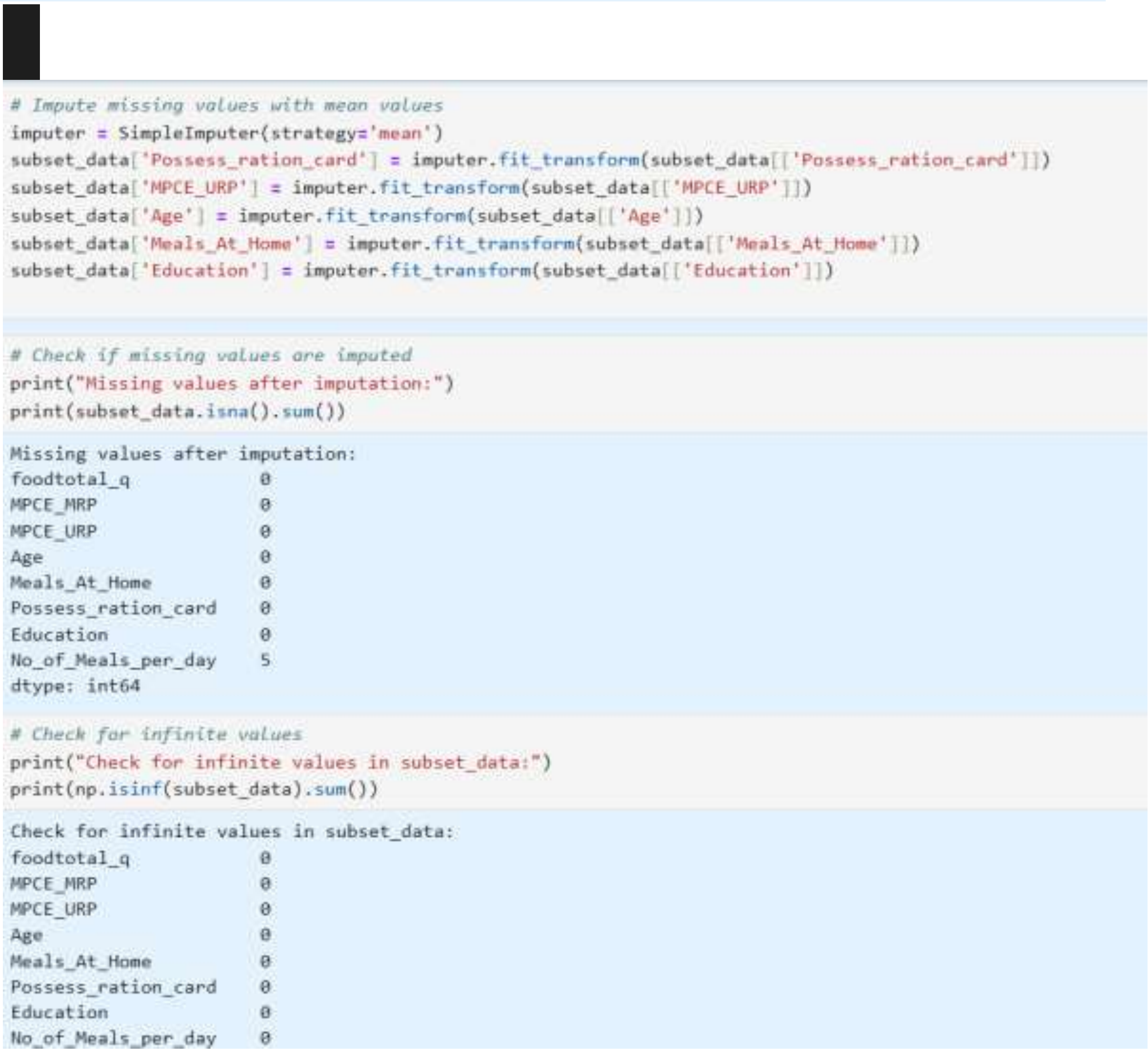
Results and Interpretation using Python

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



```
# Check for missing values in the subset_data
print("Missing values in subset_data:")
print(subset_data.isna().sum())
```

```
Missing values in subset_data:
foodtotal_q               0
MPCE_MRP                  0
MPCE_URP                  0
Age                       0
Meals_At_Home           111
Possess_ration_card       0
Education                 0
No_of_Meals_per_day       5
dtype: int64
```



```
# Impute missing values with mean values
imputer = SimpleImputer(strategy='mean')
subset_data['Possess_ration_card'] = imputer.fit_transform(subset_data[['Possess_ration_card']])
subset_data['MPCE_URP'] = imputer.fit_transform(subset_data[['MPCE_URP']])
subset_data['Age'] = imputer.fit_transform(subset_data[['Age']])
subset_data['Meals_At_Home'] = imputer.fit_transform(subset_data[['Meals_At_Home']])
subset_data['Education'] = imputer.fit_transform(subset_data[['Education']])
```

```
# Check if missing values are imputed
print("Missing values after imputation:")
print(subset_data.isna().sum())
```

```
Missing values after imputation:
foodtotal_q            0
MPCE_MRP               0
MPCE_URP               0
Age                    0
Meals_At_Home          0
Possess_ration_card    0
Education              0
No_of_Meals_per_day    5
dtype: int64
```

```
# Check for infinite values
print("Check for infinite values in subset_data:")
print(np.isinf(subset_data).sum())
```

```
Check for infinite values in subset_data:
foodtotal_q            0
MPCE_MRP               0
MPCE_URP               0
Age                    0
Meals_At_Home          0
Possess_ration_card    0
Education              0
No_of_Meals_per_day    0
```

Interpretation:

- **Missing Values Handling:**

  - Before imputation, the dataset `subset_data` had missing values in two columns:
    - `Meals_At_Home`: 111 missing values

    - `No_of_Meals_per_day`: 5 missing values
- After imputation using mean values (`SimpleImputer` with `strategy='mean'`), all missing values were filled.
- After imputation, only `No_of_Meals_per_day` still had 5 missing values, which were not imputed because they were not included in the imputation step.

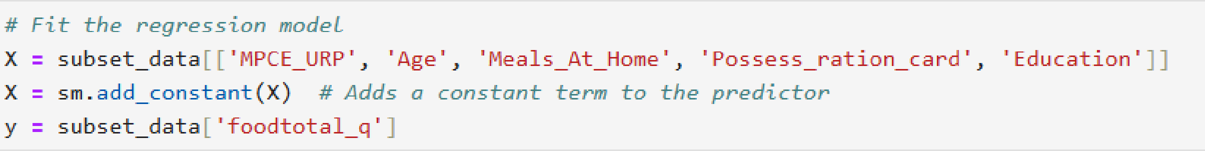
- **Infinite Values Check:**

    - There were no infinite values (`np.inf`) found in any column of the `subset_data` after imputation and data preprocessing.

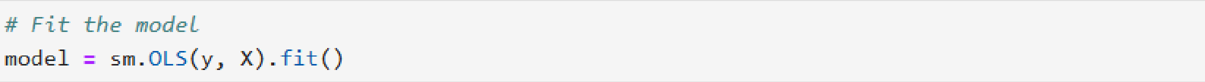
- **Overall Interpretation:**

    - The preprocessing steps successfully handled missing values using mean imputation for applicable columns (`Meals_At_Home`, `No_of_Meals_per_day`) and ensured there were no infinite values present in the dataset.
    - The dataset `subset_data` is now ready for further analysis or modeling, with missing values effectively managed and no apparent data integrity issues detected.

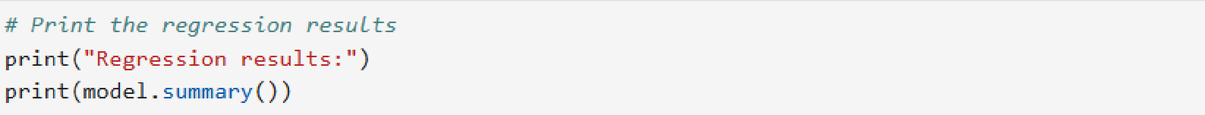
- Perform Multiple regression analysis and carry out the regression diagnostics.



```
# Fit the regression model
X = subset_data[['MPCE_URP', 'Age', 'Meals_At_Home', 'Possess_ration_card', 'Education']]
X = sm.add_constant(X)  # Adds a constant term to the predictor
y = subset_data['foodtotal_q']
```



```
# Fit the model
model = sm.OLS(y, X).fit()
```



```
# Print the regression results
print("Regression results:")
print(model.summary())
```

```
Regression results:
                    OLS Regression Results
==============================================================================
====
Dep. Variable:          foodtotal_q   R-squared:                  0.154
Model:                          OLS   Adj. R-squared:             0.153
Method:               Least Squares   F-statistic:                228.9
Date:              Sun, 23 Jun 2024   Prob (F-statistic):        3.20e-225
Time:                      20:10:40   Log-Likelihood:            -21838.
No. Observations:              6310   AIC:                     4.369e+04
Df Residuals:                  6304   BIC:                     4.373e+04
Df Model:                         5
Covariance Type:          nonrobust
==============================================================================
=============
                    coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const            13.2014      0.754     17.500      0.000      11.723      14.680
MPCE_URP          0.0012   5.28e-05     22.199      0.000       0.001       0.001
Age               0.0804      0.007     11.396      0.000       0.067       0.094
```

```
Meals_At_Home        0.0909     0.006    14.585     0.000     0.079     0.103
Possess_ration_card  -2.2419     0.383    -5.853     0.000    -2.993    -1.491
Education          0.2179     0.030     7.320     0.000     0.160     0.276
==============================================================================
====
Omnibus:              1744.758   Durbin-Watson:              1.533
Prob(Omnibus):              0.000   Jarque-Bera (JB):        33258.876
Skew:                  0.839   Prob(JB):                    0.00
Kurtosis:              14.121   Cond. No.               2.37e+04
==============================================================================
```

Interpretation:

The regression results indicate that the model explains approximately 15.4% of the variance in the dependent variable, foodtotal_q, as suggested by the R-squared value of 0.154. This means that the independent variables included in the model (MPCE_URP, Age, Meals_At_Home, Possess_ration_card, and Education) collectively account for about 15.4% of the variation observed in foodtotal_q.

Specifically, each coefficient represents the estimated change in foodtotal_q for a one-unit increase in the corresponding independent variable, holding all other variables constant. For instance, MPCE_URP shows a positive relationship with foodtotal_q, suggesting that higher Monthly Per Capita Expenditure on Usual Rate Purchases is associated with an increase in foodtotal_q. Age and Meals_At_Home also have positive coefficients, indicating that older age and more meals consumed at home are associated with higher foodtotal_q. On the other hand, Possess_ration_card has a negative coefficient, implying that possessing a ration card is associated with a decrease in foodtotal_q. Education shows a positive association, indicating that higher levels of education are associated with higher foodtotal_q.

The statistical significance of these coefficients is confirmed by their respective p-values (P>|t|), which are all very small (close to 0), suggesting that the observed relationships are unlikely to be due to random chance.

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

[2] The condition number is large, 2.37e+04. This might indicate that there are

strong multicollinearity or other numerical problems.

# Regression Diagnostics

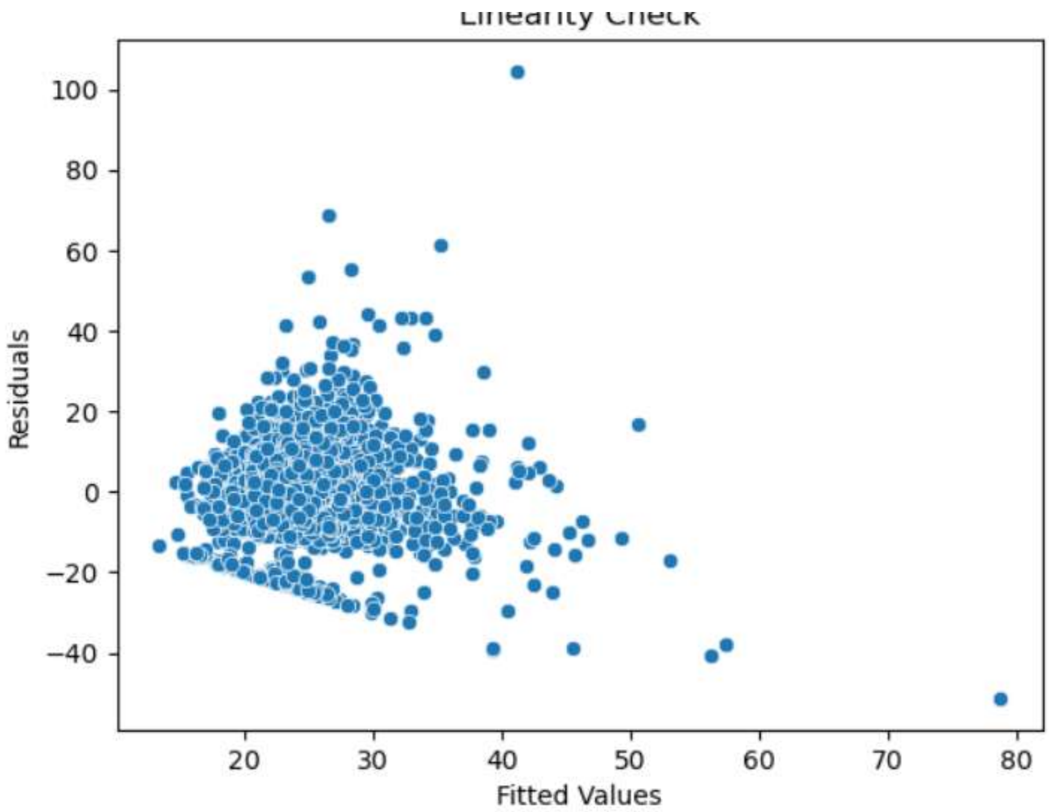
1. **Multicollinearity**

```
# Check for multicollinearity using Variance Inflation Factor (VIF)
vif_data = pd.DataFrame()
vif_data['feature'] = X.columns
vif_data['VIF'] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]
print("VIF data:")
print(vif_data)
```

```
VIF data:
                  feature        VIF
0                   const  60.414198
1                MPCE_URP   1.286953
2                     Age   1.030163
3           Meals_At_Home   1.057454
4    Possess_ration_card   1.033477
5               Education   1.239538
```

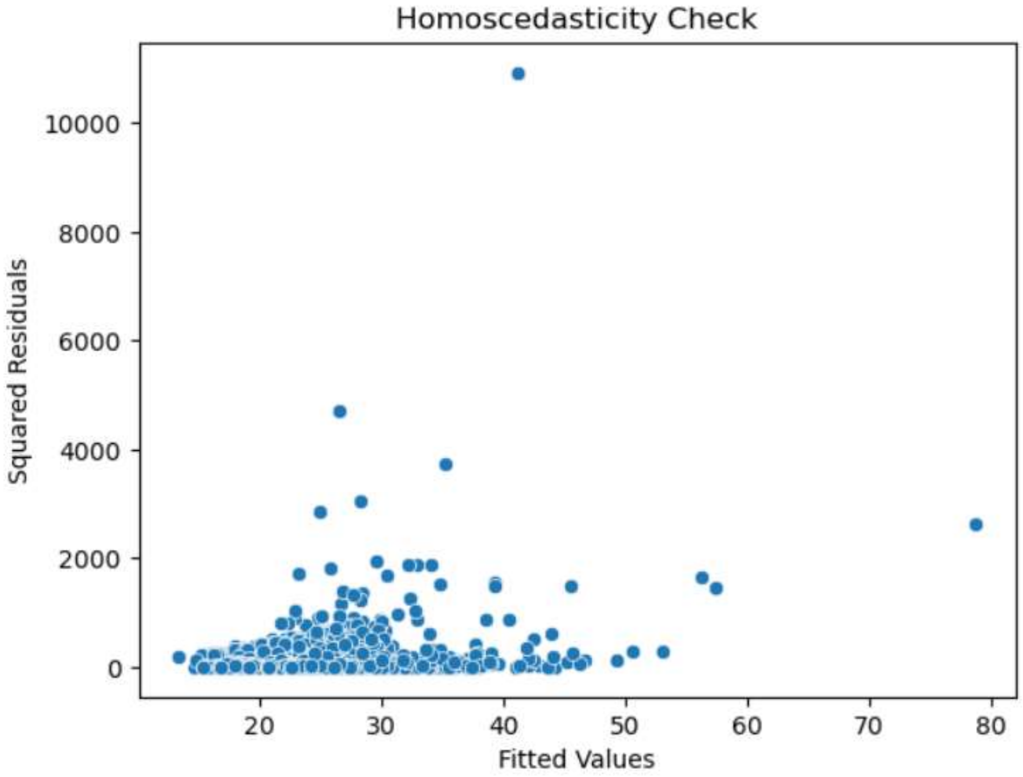
2.**Linearity Check**

```
sns.scatterplot(x=model.fittedvalues, y=model.resid)
plt.xlabel('Fitted Values')
plt.ylabel('Residuals')
plt.title('Linearity Check')
plt.show()
```



3.**Homoschedasticity check**

```
 sns.scatterplot(x=model.fittedvalues, y=model.resid**2)
plt.xlabel('Fitted Values')
plt.ylabel('Squared Residuals')
plt.title('Homoscedasticity Check')
plt.show()
```

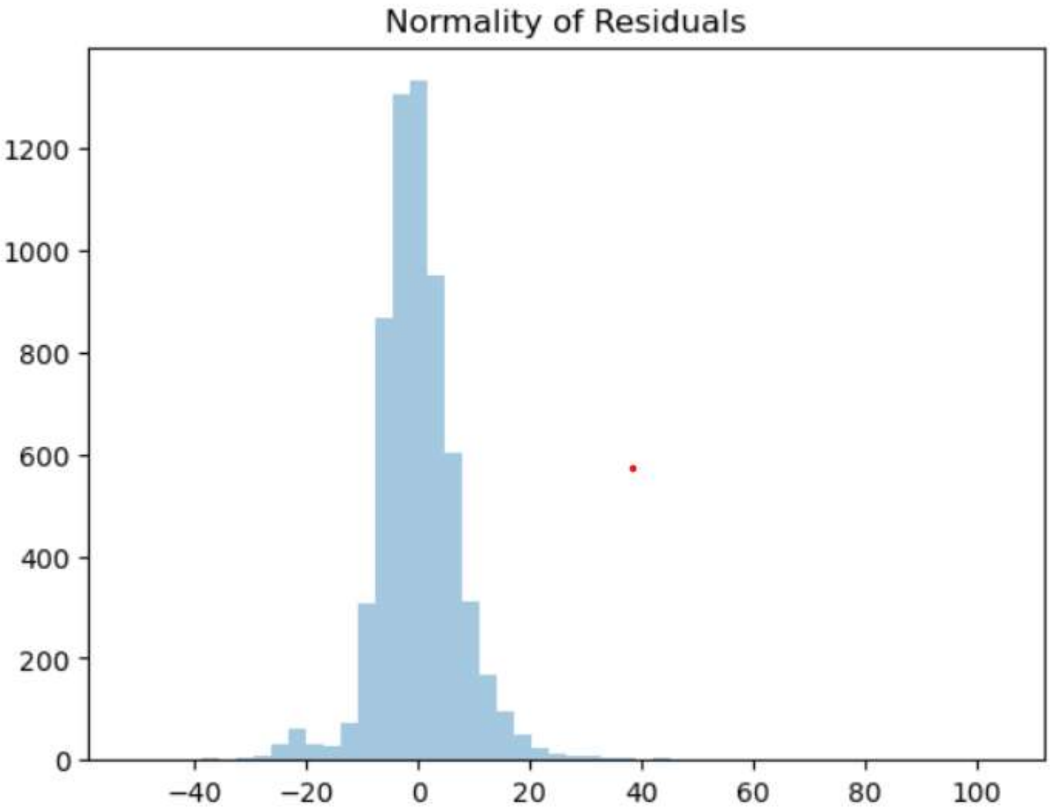


4.**Normality of Residuals**

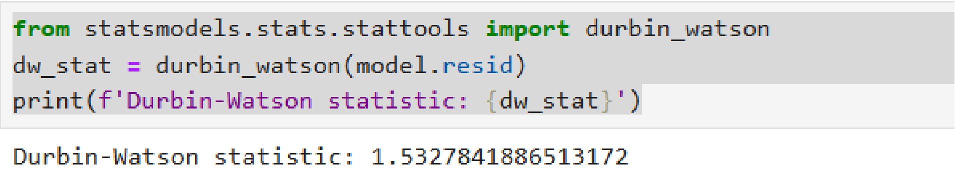
```
sns.distplot(model.resid, kde=False)

plt.title('Normality of Residuals')

plt.show()
```

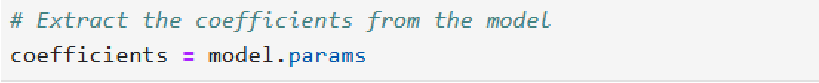


5.**Durbin Watson Statistic**

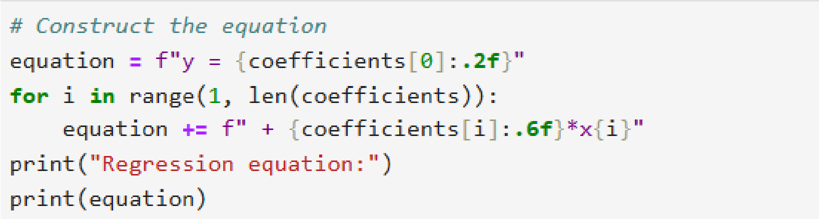


```
from statsmodels.stats.stattools import durbin_watson
dw_stat = durbin_watson(model.resid)
print(f'Durbin-Watson statistic: {dw_stat}')
```

```
Durbin-Watson statistic: 1.5327841886513172
```



```
# Extract the coefficients from the model
coefficients = model.params
```



```
# Construct the equation
equation = f"y = {coefficients[0]:.2f}"
for i in range(1, len(coefficients)):
    equation += f" + {coefficients[i]:.6f}*x{i}"
print("Regression equation:")
print(equation)
```

Regression equation:

$$y = 13.20 + 0.001171*x1 + 0.080444*x2 + 0.090922*x3 + -2.241875*x4 + 0.217936*x5$$

Interpretation:

In regression diagnostics, multicollinearity refers to the situation where independent variables in a regression model are highly correlated with each other. This can inflate the variance of the coefficient estimates and make the model unstable.

Variance Inflation Factor (VIF)

The Variance Inflation Factor (VIF) quantifies the severity of multicollinearity in an ordinary least squares regression analysis. A VIF value greater than 10 typically indicates high multicollinearity, though some consider values above 5 as problematic.

In the provided VIF data:

- **MPCE_URP** has a VIF of 1.286953.
- **Age** has a VIF of 1.030163.
- **Meals_At_Home** has a VIF of 1.057454.
- **Possess_ration_card** has a VIF of 1.033477.
- **Education** has a VIF of 1.239538.

These VIF values are all below the threshold of 5, suggesting that multicollinearity is not a major concern in this model. The variables are not highly correlated, and the estimates of the coefficients should be reliable.

Regression Equation

The regression equation is:

y = 13.20 + 0.001171 \times \text{MPCE_URP} + 0.080444 \times \text{Age} + 0.090922 \times \text{Meals_At_Home} - 2.241875 \times \text{Possess_ration_card} + 0.217936 \times \text{Education}

This equation indicates how the dependent variable yyy changes with one-unit changes in the independent variables:

- For every one-unit increase in **MPCE_URP**, yyy increases by approximately 0.001171 units, holding other variables constant.
- For every one-year increase in **Age**, yyy increases by approximately 0.080444 units, holding other variables constant.
- For every one-unit increase in **Meals_At_Home**, yyy increases by approximately 0.090922 units, holding other variables constant.
- For those possessing a ration card, yyy decreases by approximately 2.241875 units, holding other variables constant.
- For every one-unit increase in **Education**, yyy increases by approximately 0.217936 units, holding other variables constant.

The intercept (13.20) represents the expected value of yyy when all independent variables are zero. This model gives a detailed insight into how each factor affects the dependent variable, considering the impact of other variables in the model.

Overall, the analysis suggests that multicollinearity is minimal, and the regression model coefficients are likely to be stable and interpretable.

Recommendation

For Python file :

- **Handling Missing Values**: Ensure all missing values are handled appropriately to avoid biases in the analysis.
- **Outlier Treatment**: Remove or treat outliers to prevent skewed results.
- **Renaming for Clarity**: Clearly rename districts and sectors to enhance readability and understanding of the data.
- **Summarize Data**: Provide summaries region-wise and district-wise to highlight key consumption patterns.
- **Statistical Testing**: Conduct tests to validate the significance of differences in means, ensuring robust conclusions.

For R file :

- **Handling Missing Values**: Replace missing values with the mean of the variable.
- **Checking for Outliers**: Identify and amend outliers.
- **Renaming Districts and Sectors**: Rename categories for clarity.
- **Summarizing Data**: Summarize critical variables region-wise and district-wise.
- **Testing for Significant Differences**: Test if differences in means are significant.

R Codes

```
#NSSO

library(dplyr)

setwd('C:\\Users\\sayas\\OneDrive\\New folder\\python projects')

getwd()


# Load the dataset

data <- read.csv("NSSO68.csv")

unique(data$state_1)
```

```
# Subset data to state assigned
subset_data <- data %>%
  filter(state_1 == 'WB') %>%
  select(foodtotal_q, MPCE_MRP,
MPCE_URP,Age,Meals_At_Home,Possess_ration_card,Education, No_of_Meals_per_day)
print(subset_data)


sum(is.na(subset_data$MPCE_MRP))
sum(is.na(subset_data$MPCE_URP))
sum(is.na(subset_data$Age))
sum(is.na(subset_data$Possess_ration_card))
sum(is.na(data$Education))


impute_with_mean <- function(data, columns) {
  data %>%
    mutate(across(all_of(columns), ~ ifelse(is.na(.), mean(., na.rm = TRUE), .)))
}


# Columns to impute
columns_to_impute <- c("Education")


# Impute missing values with mean
data <- impute_with_mean(data, columns_to_impute)


sum(is.na(data$Education))


# Fit the regression model
```

```
model <- lm(foodtotal_q~
MPCE_MRP+MPCE_URP+Age+Meals_At_Home+Possess_ration_card+Education, data =
subset_data)


# Print the regression results

print(summary(model))




library(car)

# Check for multicollinearity using Variance Inflation Factor (VIF)

vif(model) # VIF Value more than 8 its problematic


# Extract the coefficients from the model

coefficients <- coef(model)


# Construct the equation

equation <- paste0("y = ", round(coefficients[1], 2))

for (i in 2:length(coefficients)) {

  equation <- paste0(equation, " + ", round(coefficients[i], 6), "*x", i-1)

}

# Print the equation

print(equation)






head(subset_data$MPCE_MRP,1)

head(subset_data$MPCE_URP,1)

head(subset_data$Age,1)

head(subset_data$Meals_At_Home,1)
head(subset_data$Possess_ration_card,1)
head(subset_data$Education,1)
head(subset_data$foodtotal_q,1)
```

Python Codes

```
# Set the working directory
os.chdir("C:\\Users\\sayas\\OneDrive\\New folder\\python projects")
print(f"Current working directory: {os.getcwd()}")References
```

```
 # Load the dataset
data = pd.read_csv("C:\\Users\\sayas\\OneDrive\\New folder\\python projects\\NSSO68.csv")


# Display the first few rows of the dataset to ensure it's loaded correctly
print(data.head())
# # Display unique values in the 'state_1' column
unique_states = data['state_1'].drop_duplicates().tolist()
print(f"Unique states: {unique_states}")
# Check for missing values in the subset_data
print("Missing values in subset_data:")
print(subset_data.isna().sum())
# Impute missing values with mean values
imputer = SimpleImputer(strategy='mean')
subset_data['Possess_ration_card'] = imputer.fit_transform(subset_data[['Possess_ration_card']])
subset_data['MPCE_URP'] = imputer.fit_transform(subset_data[['MPCE_URP']])
subset_data['Age'] = imputer.fit_transform(subset_data[['Age']])
subset_data['Meals_At_Home'] = imputer.fit_transform(subset_data[['Meals_At_Home']])
subset_data['Education'] = imputer.fit_transform(subset_data[['Education']])
# Check if missing values are imputed
print("Missing values after imputation:")
print(subset_data.isna().sum())
```

```
# Check for infinite values
print("Check for infinite values in subset_data:")
print(np.isinf(subset_data).sum())
# Drop rows with any remaining missing or infinite values
subset_data = subset_data.replace([np.inf, -np.inf], np.nan).dropna()
# Fit the regression model
X = subset_data[['MPCE_URP', 'Age', 'Meals_At_Home', 'Possess_ration_card', 'Education']]
X = sm.add_constant(X)  # Adds a constant term to the predictor
y = subset_data['foodtotal_q']
# Print the regression results
print("Regression results:")
print(model.summary())
# Check for multicollinearity using Variance Inflation Factor (VIF)
vif_data = pd.DataFrame()
vif_data['feature'] = X.columns
vif_data['VIF'] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]
print("VIF data:")
print(vif_data)
```

Reference

1. www.github.com
2. www.geeksforgeeks.com
3. www.datacamp.com