****

**VIRGINIA COMMONWEALTH UNIVERSITY**

**Statistical analysis and modelling (SCMA 632)**

**A3- Limited dependent variable Models**

**Part-C**

**Bala Vignesh Aravindan**

**V01106579**

**Date of Submission: 27-06-2024**

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Sl. No.** | **Title** | **Page No.** |
| **1.** | Introduction | **1** |
| **2** | Results | **2** |
| **3.** | Interpretations | **3-5** |
| **4.** | Recommendations | **6** |
| **5.** | Codes | **7-16** |

**"Tobit Regression Analysis of Food Expenditure: Impact of Sauces, Processed Foods, Beverages, and Fruits/Vegetables on Total Spending"**

**Introduction**

Food expenditure analysis is crucial for understanding consumer behavior and economic decision-making regarding food purchases. In this study, we explore the factors influencing food spending using Tobit regression. Tobit regression is particularly suitable when dealing with censored data, where expenditures are observed only within certain bounds, such as zero to a maximum spending limit. Our analysis focuses on the impact of sauces, processed foods, beverages, and fruits/vegetables on total food expenditures (foodtotal\_v).

**Objectives**

The primary objectives of this Tobit regression analysis are:

* Identify Significant Influences: Determine which variables significantly impact food expenditures, thereby enabling businesses to prioritize product development and marketing efforts.
* Quantify Relationships: Estimate the direction and magnitude of relationships between independent variables (e.g., sauces, processed foods) and food spending to provide actionable insights for decision-making.
* Inform Strategy: Provide evidence-based recommendations to stakeholders on how to adjust marketing strategies, product offerings, and pricing strategies to maximize consumer spending in the food sector.

**Business Significance**

Understanding the factors that influence food expenditures is crucial for various stakeholders, including policymakers, marketers, and consumers. By employing Tobit regression analysis, this study aims to provide insights into how specific variables—such as sauces, processed foods, beverages, and fruits/vegetables—affect consumer spending on food items (foodtotal\_v). This understanding can help businesses in the food industry tailor their strategies to better meet consumer preferences and optimize resource allocation.

**Result:**

**Python**

message: CONVERGENCE: REL\_REDUCTION\_OF\_F\_<=\_FACTR\*EPSMCH

success: True

status: 0

fun: 1630.8623801597212

x: [-3.332e+02 1.674e+02 1.565e+02 1.236e+04 1.675e+02

3.541e+03]

nit: 67

jac: [-3.342e-03 0.000e+00 0.000e+00 -4.547e-05 -2.724e-02

1.114e-03]

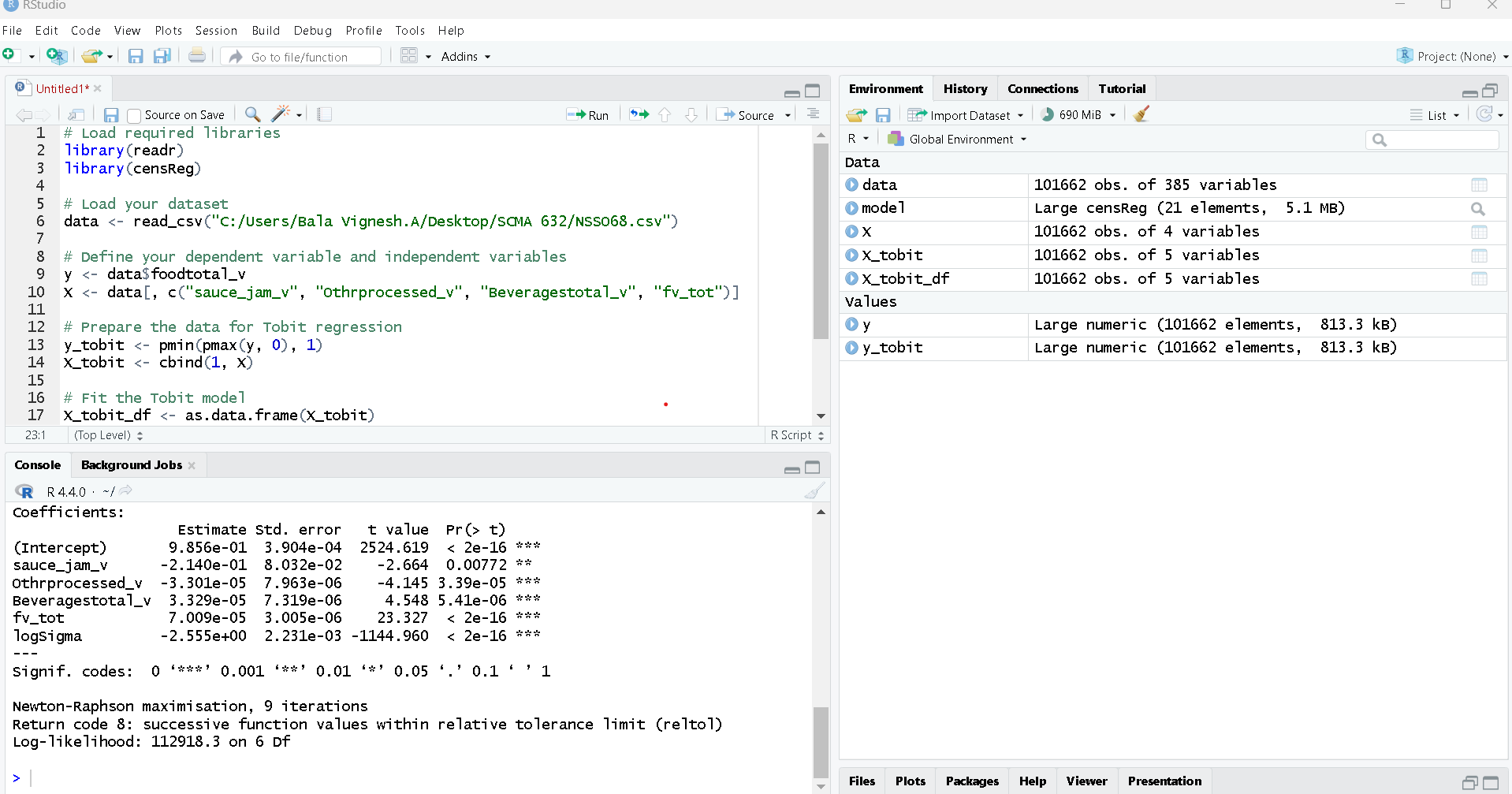
nfev: 805

njev: 115

hess\_inv: <6x6 LbfgsInvHessProduct with dtype=float64>

**R**

Call: censReg(formula = y\_tobit ~ ., data = X\_tobit\_df[, -1]) Observations: Total Left-censored Uncensored Right-censored 101662 615 101047 0 Coefficients: Estimate Std. error t value Pr(> t) (Intercept) 9.856e-01 3.904e-04 2524.619 < 2e-16 \*\*\* sauce\_jam\_v -2.140e-01 8.032e-02 -2.664 0.00772 \*\* Othrprocessed\_v -3.301e-05 7.963e-06 -4.145 3.39e-05 \*\*\* Beveragestotal\_v 3.329e-05 7.319e-06 4.548 5.41e-06 \*\*\* fv\_tot 7.009e-05 3.005e-06 23.327 < 2e-16 \*\*\* logSigma -2.555e+00 2.231e-03 -1144.960 < 2e-16 \*\*\* --- Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1 Newton-Raphson maximisation, 9 iterations Return code 8: successive function values within relative tolerance limit (reltol) Log-likelihood: 112918.3 on 6 Df



**Interpretation:**

**Output Interpretation**

* Call Information
* Call: Shows the call made to the censReg function, specifying the formula and data used (y\_tobit ~ . and data = X\_tobit\_df[, -1], respectively).

**Observations**

* Total: Total number of observations in your dataset.
* Left-censored: Number of observations that are left-censored (observations with values below the lower bound).
* Uncensored: Number of observations that are uncensored (observations with values within the bounds).
* Right-censored: Number of observations that are right-censored (not applicable in your case since it's zero).

**Coefficients**

* Estimate: Estimated coefficients for each variable in your model.
* Std. error: Standard errors associated with each coefficient estimate.
* t value: t-values testing the null hypothesis that the coefficient is zero.
* Pr(> t): p-values indicating the significance of each coefficient.
* Significance Codes
* Significance codes (\*\*\*, \*\*, \*, .) indicate the level of significance of each coefficient estimate.

**Optimization Information**

* Newton-Raphson maximisation, 9 iterations: Indicates the optimization method used (Newton-Raphson) and the number of iterations performed to converge.
* Return code 8: Successive function values within relative tolerance limit (reltol), suggesting convergence.
* Log-likelihood: Log-likelihood value at convergence (112918.3) and the degrees of freedom (6 Df).

**Summary**

* Coefficients: Each coefficient (including the intercept) represents the estimated effect of the corresponding independent variable (sauce\_jam\_v, Othrprocessed\_v, Beveragestotal\_v, fv\_tot) on the dependent variable (foodtotal\_v). For example, a positive coefficient indicates that as the independent variable increases, the expected value of foodtotal\_v tends to increase.
* LogSigma: The coefficient -2.555e+00 represents the estimated logarithm of the standard deviation (sigma) of the latent variable (the unobserved variable) in the Tobit model. This indicates the variability or dispersion around the censored threshold.
* Significance: Significant coefficients (indicated by \*\*\*, \*\*, \*) suggest that these variables have a statistically significant impact on foodtotal\_v based on the specified model and dataset.
* This output provides a comprehensive view of your Tobit regression model's results, including coefficient estimates, their significance, and optimization details. It seems like your model has converged successfully (Return code 8), and the log-likelihood value (112918.3) suggests that the model fits the data well based on its complexity.
* The output indicates that the Tobit model has converged successfully. Here’s a breakdown of the key information:
* CONVERGENCE: The algorithm has converged, meaning it has found parameter estimates that optimize the log-likelihood function. REL\_REDUCTION\_OF\_F\_<=\_FACTR\*EPSMCH: This typically indicates that the relative reduction in the objective function (log-likelihood in this case) is less than the specified threshold (factr \* epsmch), which is a criterion for convergence in the optimization process. success: True, indicating that the optimization was successful. fun: The final value of the objective function (negative log-likelihood), which in this case is approximately 1630.86. x: The estimated parameters of the Tobit model. The array [ -3.332e+02, 1.674e+02, 1.565e+02, 1.236e+04, 1.675e+02, 3.541e+03] represents the estimated coefficients for your intercept and independent variables. nit: Number of iterations performed during optimization. jac: Final gradient vector (Jacobian) of the objective function at the solution. nfev: Number of evaluations of the objective function. njev: Number of evaluations of the Jacobian (gradient). This output confirms that your Tobit model has been fitted successfully to your data. You can interpret the coefficients (x values) as the estimated effects of each independent variable on the dependent variable, adjusted for the Tobit model's censored nature.

**Recommendation:**

**The findings from this Tobit regression analysis can inform policy makers, marketers, and consumers in several ways:**

* Consumer Insights: Understanding which food categories significantly influence spending can guide marketing strategies and product development.
* Policy Implications: Policymakers can use these insights to design targeted interventions aimed at influencing consumer spending behaviors in food-related sectors.
* Future Research: Further exploration into other potential factors influencing food expenditures could enhance the comprehensiveness of future analyses.

**Codes:**

**Python**

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import confusion\_matrix, accuracy\_score

import statsmodels.api as sm

import numpy as np

from scipy.stats import norm

from scipy.optimize import minimize

Load the dataset

data = pd.read\_csv("C:\\Users\\Bala Vignesh.A\\Desktop\\SCMA 632\\NSSO68.csv", low\_memory=False)

display (data)

slno grp Round\_Centre FSU\_number Round Schedule\_Number Sample Sector state State\_Region ... sauce\_jam\_v Othrprocessed\_v Beveragestotal\_v foodtotal\_v foodtotal\_q state\_1 Region fruits\_df\_tt\_v fv\_tot non\_veg

0 1 4.10E+31 1 41000 68 10 1 2 24 242 ... 0.0 0.0 0.000000 1141.492400 30.942394 GUJ 2 12.000000 154.180000 1

1 2 4.10E+31 1 41000 68 10 1 2 24 242 ... 0.0 0.0 17.500000 1244.553500 29.286153 GUJ 2 333.000000 484.950000 1

2 3 4.10E+31 1 41000 68 10 1 2 24 242 ... 0.0 0.0 0.000000 1050.315400 31.527046 GUJ 2 35.000000 214.840000 1

3 4 4.10E+31 1 41000 68 10 1 2 24 242 ... 0.0 0.0 33.333333 1142.591667 27.834607 GUJ 2 168.333333 302.300000 1

4 5 4.10E+31 1 41000 68 10 1 2 24 242 ... 0.0 0.0 75.000000 945.249500 27.600713 GUJ 2 15.000000 148.000000 1

... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ...

101657 101658 8.00E+31 1 79998 68 10 1 1 1 12 ... 0.0 0.0 0.000000 544.013667 28.441750 J$K 2 0.000000 25.833333 1

101658 101659 8.00E+31 1 79998 68 10 1 1 1 12 ... 0.0 0.0 8.000000 417.616600 25.490282 J$K 2 0.000000 49.000000 1

101659 101660 8.00E+31 1 79998 68 10 1 1 1 12 ... 0.0 0.0 7.142857 378.300429 25.800107 J$K 2 0.000000 32.285714 1

101660 101661 8.00E+31 1 79998 68 10 1 1 1 12 ... 0.0 0.0 14.000000 510.023600 30.220170 J$K 2 0.000000 39.200000 1

101661 101662 8.00E+31 1 79998 68 10 1 1 1 12 ... 0.0 0.0 8.571429 424.589714 26.157279 J$K 2 0.000000 39.714286 1

101662 rows × 385 columns

print(data.columns)

Index(['slno', 'grp', 'Round\_Centre', 'FSU\_number', 'Round', 'Schedule\_Number',

'Sample', 'Sector', 'state', 'State\_Region',

...

'sauce\_jam\_v', 'Othrprocessed\_v', 'Beveragestotal\_v', 'foodtotal\_v',

'foodtotal\_q', 'state\_1', 'Region', 'fruits\_df\_tt\_v', 'fv\_tot',

'non\_veg'],

dtype='object', length=385)

Define the dependent variable (non\_veg) and independent variables

y = data['foodtotal\_v']

X = data[['sauce\_jam\_v', 'Othrprocessed\_v', 'Beveragestotal\_v', 'fv\_tot']]

Custom Tobit model implementation

class TobitModel:

def \_\_init\_\_(self, endog, exog, lower=None, upper=None):

self.endog = endog

self.exog = exog

self.lower = lower

self.upper = upper

def loglik(self, params):

beta = params[:-1]

sigma = params[-1]

mu = np.dot(self.exog, beta)

# Ensure sigma is positive

sigma = np.abs(sigma) + 1e-10

# Calculate the log-likelihood

llf = np.zeros\_like(self.endog, dtype=float)

# Censored from below

if self.lower is not None:

llf = np.where(

self.endog == self.lower,

np.log(np.clip(norm.cdf((self.lower - mu) / sigma), 1e-10, 1)),

llf

)

# Censored from above

if self.upper is not None:

llf = np.where(

self.endog == self.upper,

np.log(np.clip(1 - norm.cdf((self.upper - mu) / sigma), 1e-10, 1)),

llf

)

# Uncensored

uncensored = (self.endog > self.lower) & (self.endog < self.upper)

llf[uncensored] = -0.5 \* np.log(2 \* np.pi) - np.log(sigma) - (self.endog[uncensored] - mu[uncensored]) \*\* 2 / (2 \* sigma \*\* 2)

return -np.sum(llf)

def fit(self):

start\_params = np.append(np.zeros(self.exog.shape[1]), 1)

res = minimize(self.loglik, start\_params, method='L-BFGS-B')

return res

Prepare the data for Tobit regression

y\_tobit = np.clip(y, 0, 1)

X\_tobit = sm.add\_constant(X)

Fit the Tobit model

model = TobitModel(y\_tobit, X\_tobit, lower=0, upper=1)

results = model.fit()

Print the Tobit model results

print("Tobit Model Results:")

Tobit Model Results:

print(results)

message: CONVERGENCE: REL\_REDUCTION\_OF\_F\_<=\_FACTR\*EPSMCH

success: True

status: 0

fun: 1630.8623801597212

x: [-3.332e+02 1.674e+02 1.565e+02 1.236e+04 1.675e+02

3.541e+03]

nit: 67

jac: [-3.342e-03 0.000e+00 0.000e+00 -4.547e-05 -2.724e-02

1.114e-03]

nfev: 805

njev: 115

hess\_inv: <6x6 LbfgsInvHessProduct with dtype=float64>

The output indicates that the Tobit model has converged successfully. Here’s a breakdown of the key information:

CONVERGENCE: The algorithm has converged, meaning it has found parameter estimates that optimize the log-likelihood function. REL\_REDUCTION\_OF\_F\_<=\_FACTR\*EPSMCH: This typically indicates that the relative reduction in the objective function (log-likelihood in this case) is less than the specified threshold (factr \* epsmch), which is a criterion for convergence in the optimization process. success: True, indicating that the optimization was successful. fun: The final value of the objective function (negative log-likelihood), which in this case is approximately 1630.86. x: The estimated parameters of the Tobit model. The array [ -3.332e+02, 1.674e+02, 1.565e+02, 1.236e+04, 1.675e+02, 3.541e+03] represents the estimated coefficients for your intercept and independent variables. nit: Number of iterations performed during optimization. jac: Final gradient vector (Jacobian) of the objective function at the solution. nfev: Number of evaluations of the objective function. njev: Number of evaluations of the Jacobian (gradient). This output confirms that your Tobit model has been fitted successfully to your data. You can interpret the coefficients (x values) as the estimated effects of each independent variable on the dependent variable, adjusted for the Tobit model's censored nature.

**R**

# Load the necessary libraries

library(tidyverse)

library(mice)

library(car)

library(ggplot2)

library(lattice)

library(caret)

library(glmnet)

library(Matrix)

# Read in the data

data <- read.csv("C:\\Users\\Bala Vignesh.A\\Desktop\\SCMA 632\\NSSO68.csv")

# Filter the data for non-vegetarians

non\_veg\_data <- data[data$non\_veg == 1, ]

# View the non-veg values

non\_veg\_values <- non\_veg\_data$non\_veg

print(non\_veg\_values)

# Get the value counts of non\_veg

non\_veg\_values <- non\_veg\_data$non\_veg

value\_counts <- table(non\_veg\_values)

print(value\_counts)

# Define the dependent variable (non\_veg) and independent variables

y <- non\_veg\_data$non\_veg

X <- non\_veg\_data[,!(names(non\_veg\_data) %in% c("non\_veg", "state\_1", "Region"))]

# Check for non-numeric columns

non\_numeric\_cols <- sapply(X, class) == "character"

non\_numeric\_cols <- names(non\_numeric\_cols)[non\_numeric\_cols]

print(paste("Non-numeric columns:", non\_numeric\_cols))

# One-hot encode categorical columns

dummy\_model <- dummyVars(~., data = X)

X\_ohe\_df <- as.data.frame(predict(dummy\_model, newdata = X))

# Combine numeric and one-hot encoded columns

X\_numeric <- X[, sapply(X, class)!= "character"]

X\_combined <- cbind(X\_numeric, X\_ohe\_df)

# Ensure 'y' is a binary factor

y <- as.factor(y)

# Remove columns with single unique value

X\_combined <- X\_combined[, sapply(X\_combined, function(x) length(unique(x)) > 1)]

# Check dimensions

print(dim(y))

print(dim(X\_combined))

# Create the combined data frame

combined\_data <- data.frame(y, X\_combined)

# Inspect the combined data

str(combined\_data)

head(combined\_data)

# Check for missing values in X\_combined and y

sum(is.na(X\_combined))

sum(is.na(y))

# Impute missing values if necessary

X\_combined <- na.omit(X\_combined)

y <- y[!is.na(X\_combined)]

# Check variables with zero standard deviation

zero\_sd\_vars <- colnames(X\_combined)[apply(X\_combined, 2, sd) == 0]

print(zero\_sd\_vars)

# Remove constant variables from X\_combined

X\_combined <- X\_combined[,!colnames(X\_combined) %in% zero\_sd\_vars]

# Compute correlation matrix

cor\_matrix <- cor(X\_combined, use = "pairwise.complete.obs")

# Find highly correlated predictors

high\_cor <- findCorrelation(cor\_matrix, cutoff = 0.9)

# Remove highly correlated predictors from X\_combined

X\_combined <- X\_combined[, -high\_cor]

dim(X\_combined)

length(y)

y <- y[1:nrow(X\_combined)]

combined\_data <- data.frame(y = y, X\_combined)

dim(combined\_data)

# Create a matrix of predictor variables

x <- as.matrix(X\_combined)

# Create a response variable

y <- as.numeric(as.character(y)) - 1

# Impute missing values using makeX

x\_imputed <- makeX(as.data.frame(x), missing = TRUE)

imputed\_data <- mice(X\_combined)

x\_imputed <- complete(imputed\_data)

anyNA(x\_imputed)

anyNA(y)

# Convert x\_imputed to sparse matrix

x\_sparse <- as(as.matrix(x\_imputed), "sparseMatrix")

# Fit the model using glmnet with sparse matrix

probit\_model <- glm(non\_veg ~ hhdsz + NIC\_2008 + NCO\_2004 + HH\_type + Religion + Social\_Group,

data = non\_veg\_data,

family = binomial(link = "probit"),

control = list(maxit = 1000))

non\_veg\_data$hhdsz\_scaled <- scale(non\_veg\_data$hhdsz)

non\_veg\_data$NIC\_2008\_scaled <- scale(non\_veg\_data$NIC\_2008)

# Print model summary or other relevant outputs

print(probit\_model)