****

**VIRGINIA COMMONWEALTH UNIVERSITY**

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

**A3a: Logistic Regression of Airline in Customer Satisfaction**

**A3b: Probit Regression of Consumption of Arunachal Pradesh**

**A3c: Tobit Regression of Consumption of Arunachal Pradesh**

**DEEPTHI ANNA ALEX**

**V01101949**

**Date of Submission: 01-07-2024**

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Sl. No.** | **Title (A3a)** | **Page No.** |
| **1.** | Introduction | **1** |
| **2** | Objectives and results | **1-2** |
| **3.** | Interpretations | **3** |
| **4.** | Codes | **4-10** |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **Sl. No.** | **Title (A3 b and c)** | **Page No.** |
| **1.** | Introduction | **10** |
| **2** | Results | **10-11** |
| **3.** | Interpretations | **11** |
| **4.** | Codes | **12-19** |

**A3a**

**Introduction:**

This analysis examines factors affecting airline customer satisfaction using logistic regression and decision tree analysis. These techniques help guide strategic decisions to improve customer experience and satisfaction levels. Logistic regression models predict satisfaction based on predictors, while decision tree analysis captures non-linear relationships and interactions. The analysis validates logistic regression assumptions, evaluates models, interprets results, and compares strengths and weaknesses of each approach. The goal is to provide a comprehensive understanding of customer satisfaction in the airline industry.

**Objective**

1.To assess the effectiveness of decision trees and logistic regression on a given data set. We will evaluate their prowess and limitations in forecasting the target variable, stressing how crucial it is to comprehend the crucial elements in our investigation.

**Results:**

Logistic Regression Results:

The model of logistic regression provided important information about the variables affecting customer satisfaction. It was discovered that important factors like age, flight distance, seat comfort, convenience of departure and arrival times, and the quality of food and drink were statistically significant in predicting whether or not a passenger would be happy with their airline experience. Which factors have a positive or negative impact on satisfaction levels is highlighted by the model coefficients, which also revealed the direction and degree of these associations.

Confusion Matrix:

The logistic regression model's performance was assessed using the confusion matrix. It included information about the number of false positives (erroneously predicted as satisfied), false negatives (erroneously projected as unsatisfied), and true positives (accurately predicted as satisfied). A fuller grasp of the model's recall, accuracy, precision, and general efficacy in forecasting customer happiness was made possible by this evaluation.

Decision Tree Analysis:

To investigate non-linear correlations and offer a different method of satisfaction prediction, a decision tree analysis was also carried out. The decision tree was utilized to segment the data according to many criteria, thereby discovering thresholds and inter-predictor interactions that have the greatest impact on deciding customer happiness. By displaying the dataset's hierarchical decision-making process, this method provided interpret ability.

**Interpretation:**

1.Tree Structure: Look at the nodes and splits in the decision tree structure to see how different factors affect the prediction of customer satisfaction.  
2.Confusion Matrix: Use this tool to analyze the confusion matrix and determine how well the decision tree model predicts each class.  
3.Comparison: Examine how the logistic regression model and the decision tree compare in terms of interpret ability, sensitivity, specificity, and accuracy.

**Codes**

**R code**

read.csv("C:\\Users\\HP\\OneDrive\\Desktop\\deepthi asignments scma\\Airline\_customer\_satisfaction.csv\\Airline\_customer\_satisfaction.csv")

file\_path <-("C:\\Users\\HP\\OneDrive\\Desktop\\deepthi asignments scma\\Airline\_customer\_satisfaction.csv\\Airline\_customer\_satisfaction.csv")

df <- read.csv(file\_path)

library(dplyr)

summary(df)

plot\_missing(df)

sum(is.na(df))

head(df)

tail(df)

names(df)

str(df)

# Replace missing values with the mean for numeric columns

df <- df %>%

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

#Checking missing values after filling it with mean values of the column

missing\_info <- colSums(is.na(df))

cat("Missing Values Information:\n")

print(missing\_info)

cor\_matrix<-cor(df)

print(cor\_matrix)

heatmap(cor\_matrix)

boxplot(flight disrance+seat comfort+departure+foodanddrink+onlinesupport+inflightentertainment+baggagehandling+,data="C:\\Users\\HP\\OneDrive\\Desktop\\Airline\_customer\_satisfaction.csv")

install.packages("caTools")

install.packages("MLmetrics")

library(dplyr)

library(pROC)

library(rpart)

library(rpart.plot)

library(MLmetrics)

#install.packages("pROC")

#install.packages("rpart")

#install.packages("rpart.plot")

#install.packages("MLmetrics")

# Logistic Regression

set.seed(123)

split<-sample.split(Outcome,SplitRatio = 0.7)

train<-subset(df,split==TRUE)

test<-subset(df,split==FALSE)

model<-glm(Outcome~.,data=train,family=binomial)

predicted\_probs<-predict(model,newdata=test,type="response")

predicted\_class<-ifelse(predicted\_probs>=0.5,1,0)

# Confusion Matrix

CM<-ConfusionMatrix(factor(predicted\_class),factor(test$Outcome))

print(CM)

roc\_obj<-roc(test$Outcome,predicted\_probs)

auc<-auc(roc\_obj)

print(paste("AUC-ROC:",auc))

plot(roc\_obj,main="ROC Curve",print.auc=TRUE)

#decision tree analysis for the data in part A and compare the results of the

#Logistic regression and Decision tree

library(stats)

install.packages("rpart")

library(rpart)

install.packages("caTools")

library(caTools)

set.seed(123)

split<-sample.split(dia$Outcome,SplitRatio = 0.7)

train<-subset(df,split == TRUE)

test<-subset(df,split == FALSE)

model<-rpart(Outcome~.,data=train,method="class")

predicted\_probs<-predict(model,newdata=test,type="prob")

predicted\_class<-ifelse(predicted\_probs[,2]>=0.5,1,0)

ConfM<-ConfusionMatrix(factor(predicted\_class),factor(test$Outcome))

print(ConfM)-

roc\_obj<-roc(test$Outcome,predicted\_probs[,2])

auc<-auc(roc\_obj)

print(paste("AUC-ROC:",auc))

plot(roc\_obj,main="ROC Curve",print.auc=TRUE)

**PYTHON CODE**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

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

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import confusion\_matrix, roc\_curve, auc, ConfusionMatrixDisplay

from sklearn.tree import DecisionTreeClassifier

# Load the dataset

file\_path = "C:\\Users\\HP\\OneDrive\\Desktop\\Airline\_customer\_satisfaction.csv"

df = pd.read\_csv(file\_path)

# Check the first few rows of the dataframe

print(df.head())

# Check for missing values

print(df.isnull().sum())

# Assuming 'satisfaction' is the target variable and the rest are features

# Replace 'satisfaction' with the correct column name in your dataset

if 'satisfaction' not in df.columns:

print("Column 'satisfaction' not found in the dataset.")

else:

X = df.drop('satisfaction', axis=1)

y = df['satisfaction']

# Convert categorical variables to dummy/indicator variables

X = pd.get\_dummies(X, drop\_first=True)

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=123)

# Logistic Regression Model

log\_reg = LogisticRegression(max\_iter=1000)

log\_reg.fit(X\_train, y\_train)

predicted\_probs = log\_reg.predict\_proba(X\_test)[:, 1]

predicted\_class = (predicted\_probs >= 0.5).astype(int)

# Confusion Matrix

CM = confusion\_matrix(y\_test, predicted\_class)

ConfusionMatrixDisplay(confusion\_matrix=CM).plot()

plt.title('Logistic Regression Confusion Matrix')

plt.show()

# ROC and AUC

fpr, tpr, \_ = roc\_curve(y\_test, predicted\_probs)

roc\_auc = auc(fpr, tpr)

print(f"AUC-ROC for Logistic Regression: {roc\_auc}")

plt.figure()

plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve (area = {roc\_auc:.2f})')

plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')

plt.xlim([0.0, 1.0])

plt.ylim([0.0, 1.05])

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('ROC Curve for Logistic Regression')

plt.legend(loc="lower right")

plt.show()

# Decision Tree Model

tree\_model = DecisionTreeClassifier(random\_state=123)

tree\_model.fit(X\_train, y\_train)

predicted\_probs\_tree = tree\_model.predict\_proba(X\_test)[:, 1]

predicted\_class\_tree = (predicted\_probs\_tree >= 0.5).astype(int)

# Confusion Matrix for Decision Tree

CM\_tree = confusion\_matrix(y\_test, predicted\_class\_tree)

ConfusionMatrixDisplay(confusion\_matrix=CM\_tree).plot()

plt.title('Decision Tree Confusion Matrix')

plt.show()

# ROC and AUC for Decision Tree

fpr\_tree, tpr\_tree, \_ = roc\_curve(y\_test, predicted\_probs\_tree)

roc\_auc\_tree = auc(fpr\_tree, tpr\_tree)

print(f"AUC-ROC for Decision Tree: {roc\_auc\_tree}")

plt.figure()

plt.plot(fpr\_tree, tpr\_tree, color='blue', lw=2, label=f'ROC curve (area = {roc\_auc\_tree:.2f})')

plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')

plt.xlim([0.0, 1.0])

plt.ylim([0.0, 1.05])

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('ROC Curve for Decision Tree')

plt.legend(loc="lower right")

plt.show()

**A3 b and c**

**Introduction:**

The script's objectives are to investigate and evaluate the NSSO dataset in order to learn more about food consumption trends and how they connect to demographic characteristics and religious affiliations. To investigate the drivers of non-vegetarianism and to account for censoring in food intake data, it uses regression models (Probit and Tobit).  
  
Researchers can learn how dietary practices differ among various religious communities and geographical areas by conducting these analyses, which can be used to influence targeted nutrition and health interventions and policy decisions.

**Results**

1. Data Preparation and Exploration

Information from the data-set: The data-set includes data on several states (state\_1), districts, regions, industries, religious affiliations (Religion), and economic metrics (emftt\_q, emftt\_v).  
Absent Values: In order to address missing values, the mean for the columns emftt\_q and emftt\_v was imputed

1. Probit Regression:

Based on food intake (eggs no.\_q, fishprawn\_q, etc.), a Probit regression model was constructed to predict non-vegetarians (target).  
For every predictor variable, the model yielded coefficients and significance levels that showed how each variable affected the likelihood of not becoming a vegetarian.  
 Results of the Probit Regression  
Important Forecasters:  
The Probit regression model's coefficients showed which dietary items—such as fishprawn\_q, eggsno\_q, and so on—have a substantial impact on the likelihood of being labeled as non-vegetarian.  
To determine the direction and strength of the link, interpretation would include examining the sign and magnitude of these coefficients.

3. Tobit Regression Indicators  
Regression of Tobit:  
The goal of this analysis was to simulate censoring in the dependent variable, foodtotal\_v, which has a 0–1 range.  
The contribution of independent variables (sauce\_jam\_v, OTHERprocessed\_v, Beveragestotal\_v, fv\_tot) to the observed food consumption levels is explained by the Tobit model.

**Interpretation:**

Inquiries into Dietary Patterns: The study sheds light on the variables influencing food consumption patterns and how dietary preferences differ among various religious groups (Religion).  
These findings have the potential to inform nutrition programs and dietary guidelines that are adapted to certain religious and regional contexts.

**Codes**

**R CODE**

#NSSO

library(dplyr)

setwd("C:\\Users\\HP\\OneDrive\\Desktop\\deepthi asignments scma")

getwd()

# Load the dataset

data <- read.csv("C:\\Users\\HP\\OneDrive\\Desktop\\deepthi asignments scma\\NSSO68 (2).csv")

unique(data$state\_1)

# Function to install and load libraries

install\_and\_load <- function(package) {

if (!require(package, character.only = TRUE)) {

install.packages(package, dependencies = TRUE)

library(package, character.only = TRUE)

}

}

# Reading the file into R

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

dim(data)

unique(data$Religion)

# Filtering for ARP

ARP <- data %>%

filter(state == "10")

# Display dataset info

cat("Dataset Information:\n")

print(names(ARP))

print(head(ARP))

print(dim(ARP))

# Finding missing values

missing\_info <- colSums(is.na(ARP))

cat("Missing Values Information:\n")

print(missing\_info)

# Sub-setting the data

ARPnew <- ARP %>%

select(state\_1,Religion, District, Region, Sector,emftt\_q, emftt\_v)

# Check for missing values in the subset

cat("Missing Values in Subset:\n")

print(colSums(is.na(ARPnew)))

dim(ARPnew)

# Impute missing values with mean for specific columns

impute\_with\_mean <- function(column) {

if (any(is.na(column))) {

column[is.na(column)] <- mean(column, na.rm = TRUE)

}

return(column)

}

ARPnew$emftt\_q <- impute\_with\_mean(ARPnew$emftt\_q)

ARPnew$emftt\_v <- impute\_with\_mean(ARPnew$emftt\_v)

dim(ARPnew)

# Check for missing values after imputation

cat("Missing Values After Imputation:\n")

print(colSums(is.na(ARPnew)))

ARP$Religion

ARPnew$emftt\_v

ARP$Religion

unique(ARP$Religion)

str(ARP$Religion)

# Sub-setting the data

ARP\_pr <- APR %>%

select(Religion, eggsno\_q, fishprawn\_q, goatmeat\_q, beef\_q, pork\_q, chicken\_q, othrbirds\_q)

dim(ARP\_pr)

ARP\_pr$eggsno\_q

data

names(ARP\_pr)

str(ARP\_pr)

# Fitting a probit regression to identify non-vegetarians.

religion\_mapping <- c("Hinduism", "Islam", "Christianity","Jainism","Others")

ARP\_pr$Religion <- factor(ARP\_pr$Religion, labels = religion\_mapping)

table(ARP\_pr$Religion)

columns <- c('eggsno\_q','fishprawn\_q', 'goatmeat\_q', 'beef\_q','pork\_q', 'chicken\_q', 'othrbirds\_q')

data1 <- ARP[columns]

data1$target <- ifelse(data1$eggsno\_q>0,1,0)

probit\_modet <- glm(target~., data = data1, family = binomial(link = "probit"))

summary(probit\_modet)

# Performorming a Tobit regression analysis on "NSSO68.csv"

df\_ARP = data[data$state\_1 == 'ARP',]

vars <- c("state\_1","Religion", "District", "Region", "Sector","emftt\_q", "emftt\_v")

df\_ARP\_p = df\_ARP[vars]

names(df\_ARP\_p)

df\_ARP\_p$price = df\_ARP\_p$emftt\_v / df\_ARP\_p$emftt\_q

names(df\_ARP\_p)

summary(df\_ARP\_p)

head(table(df\_ARP\_p$emftt\_q))

dim(df\_ARP\_p)

names(ARP)

# dependent variable and independent variables

y <- ARP$foodtotal\_v

X <- ARP[, c("sauce\_jam\_v", "Othrprocessed\_v", "Beveragestotal\_v", "fv\_tot")]

# data for Tobit regression

y\_tobit <- pmin(pmax(y, 0), 1)

X\_tobit <- cbind(1, X)

install.packages("censReg")

library(censReg)

# Fitting the Tobit model

X\_tobit\_df <- as.data.frame(X\_tobit)

model <- censReg(y\_tobit ~ ., data = X\_tobit\_df[, -1])

# Printing model summary

summary(model)

**PYTHON CODE**

import pandas as pd

import numpy as np

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

from sklearn.linear\_model import LogisticRegression

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import confusion\_matrix, classification\_report

# Load the dataset

data\_path = "C:\\Users\\HP\\OneDrive\\Desktop\\deepthi asignments scma\\NSSO68 (2).csv"

data = pd.read\_csv(data\_path)

# Preprocessing: Handle missing values

data = data.fillna(data.mean())

# Encode categorical variables

data = pd.get\_dummies(data, drop\_first=True)

# Define features and target variable

X = data.drop('satisfaction', axis=1)

y = data['satisfaction'].apply(lambda x: 1 if x == 'satisfied' else 0) # Convert to binary classification

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Train logistic regression model

logreg = LogisticRegression(max\_iter=1000)

logreg.fit(X\_train, y\_train)

# Predict and evaluate logistic regression model

y\_pred\_logreg = logreg.predict(X\_test)

conf\_matrix\_logreg = confusion\_matrix(y\_test, y\_pred\_logreg)

class\_report\_logreg = classification\_report(y\_test, y\_pred\_logreg)

# Train decision tree classifier

dtree = DecisionTreeClassifier()

dtree.fit(X\_train, y\_train)

# Predict and evaluate decision tree model

y\_pred\_dtree = dtree.predict(X\_test)

conf\_matrix\_dtree = confusion\_matrix(y\_test, y\_pred\_dtree)

class\_report\_dtree = classification\_report(y\_test, y\_pred\_dtree)

# Output results

print("Logistic Regression Confusion Matrix:")

print(conf\_matrix\_logreg)

print("\nLogistic Regression Classification Report:")

print(class\_report\_logreg)

print("Decision Tree Confusion Matrix:")

print(conf\_matrix\_dtree)

print("\nDecision Tree Classification Report:")

print(class\_report\_dtree)

 

