VIRGINIA COMMONWEALTH UNIVERSITY

Statistical analysis and modelling (SCMA 632)

A3: LIMITED DEPENDENT VARIABLE MODELS

SARATH SABU

V01109792

Date of Submission: 01-07-2024

CONTENTS

Introduction

Part A: Logistic Regression Analysis on Framingham Dataset

In this section, we will conduct a logistic regression analysis using the Framingham dataset to predict the likelihood of a ten-year coronary heart disease (CHD) risk. Logistic regression is a powerful statistical method used for binary classification problems, allowing us to model the probability of a binary outcome based on one or more predictor variables.

We will start by validating the assumptions required for logistic regression, ensuring the model is appropriate for our data. Next, we will evaluate the model's performance using a confusion matrix and an ROC curve, which will provide insights into the model's accuracy, sensitivity, specificity, and overall predictive power. The ROC curve, in particular, will help us understand the trade-off between the true positive rate and the false positive rate across different threshold values.

After interpreting the logistic regression results, we will perform a decision tree analysis on the same dataset. Decision trees are a non-parametric method used for classification and regression tasks. By comparing the results of the logistic regression and decision tree models, we can highlight the strengths and weaknesses of each approach and determine which method is more effective for predicting ten-year CHD risk in the Framingham dataset.

Part B: Probit Regression Analysis on "NSSO68.csv" to Identify Non-Vegetarians

In this section, we will perform a probit regression analysis on the "NSSO68.csv" dataset to identify non-vegetarians. Probit regression is similar to logistic regression but assumes a normal distribution of the error terms. It is particularly useful when the dependent variable is binary, and we want to model the probability of one of the two possible outcomes.

We will discuss the results of the probit regression, focusing on the estimated coefficients and their statistical significance. Additionally, we will explain the characteristics and advantages of the probit model compared to other binary classification methods, such as logistic regression. By understanding these advantages, we can better appreciate the contexts in which probit regression is the preferred modeling technique.

Part C: Tobit Regression Analysis on "NSSO68.csv"

In the final section, we will conduct a Tobit regression analysis on the "NSSO68.csv" dataset. Tobit regression, or censored regression, is used when the dependent variable is censored, meaning it has a lower or upper limit beyond which values are not observed. This type of regression is particularly useful in scenarios where the outcome variable is restricted or has natural limits.

We will discuss the results of the Tobit regression, interpreting the coefficients and their implications. Additionally, we will explore real-world use cases of the Tobit model, demonstrating its applicability in various fields such as economics, healthcare, and social sciences. By understanding these use cases, we can appreciate the practical importance of Tobit regression in handling censored data and making informed predictions in constrained environments.

Objectives

- To conduct a logistic regression analysis on your assigned dataset, validate assumptions, evaluate with a confusion matrix and ROC curve, and interpret the results. Then, perform a decision tree analysis and compare it to the logistic regression.
- To perform a probit regression on "NSSO68.csv" to identify non-vegetarians. Discuss the results and explain the characteristics and advantages of the probit model.
- To perform a Tobit regression analysis on "NSSO68.csv," discuss the results, and explain the real-world use cases of the Tobit model.

BUSINESS SIGNIFICANCE

Part A - Logistic Regression and Decision Tree Analysis

Logistic Regression Analysis:

- **Assumption Validation**: Ensuring assumptions like linearity of logit, absence of multicollinearity, and no influential outliers leads to a more reliable model.
- **Confusion Matrix and ROC Curve**: These metrics evaluate the model's performance. The confusion matrix shows accuracy, precision, recall, and F1-score, while the ROC curve and AUC measure the model's ability to distinguish between classes.
- **Interpretation**: Identifying significant predictors helps businesses focus on key areas. For instance, if age and income are significant predictors of loan default, targeted interventions can be designed for at-risk groups.

Decision Tree Analysis:

- **Comparison with Logistic Regression**: Decision trees are non-parametric and can handle non-linear relationships and interactions between variables. They are easier to interpret and visualize, aiding decision-making.
- **Business Use**: Decision trees can segment customers into different risk categories, helping in targeted marketing strategies, risk management, and resource allocation.

Business Impact:

- By comparing both models, businesses can choose the one that provides better accuracy and interpretability for their specific context. This leads to more informed decisions, optimized processes, and potentially increased profitability.

Part B - Probit Regression

Probit Regression on NSSO68.csv to Identify Non-Vegetarians:

- **Objective**: Determine the probability of being a non-vegetarian based on demographic and socio-economic factors.
- **Characteristics and Advantages**:

    - **Probit Model**: Uses a cumulative normal distribution function to model the probability of a binary outcome. It is appropriate when the dependent variable is binary, and the underlying latent variable follows a normal distribution.
    - **Interpretation**: Provides insights into factors influencing dietary habits, which can guide policy-making, marketing strategies for food products, and health interventions.
    - **Advantages**: Handles the probability prediction more naturally for certain types of data, offering potentially better fit and interpretation in some contexts compared to logistic regression.

Business Impact:

- Identifying factors influencing non-vegetarianism can help food companies tailor their products and marketing campaigns. Health organizations can design targeted nutrition programs, and policymakers can address dietary trends in specific populations.

Part C - Tobit Regression

Tobit Regression Analysis on NSSO68.csv:

- **Objective**: Model a dependent variable that is censored, meaning it has a range limitation. For example, expenditure on luxury goods where some observations are zero.
- **Results Interpretation**: Tobit regression identifies the factors influencing both the probability of positive outcomes and the level of those outcomes. For instance, factors affecting both the likelihood and amount of consumer spending.
- **Real-World Use Cases**:
    - **Consumer Behavior**: Understanding spending patterns where not all consumers participate in the market (e.g., luxury goods, high-end services).
    - **Loan Amounts**: Modeling the amount borrowed by individuals, considering that not everyone takes out loans.
    - **Healthcare Utilization**: Studying the number of doctor visits, where some individuals do not visit doctors at all.

Business Impact:

- Tobit models provide deeper insights into both the occurrence and extent of outcomes, enabling businesses to understand and predict customer behavior better. This can lead to more effective marketing strategies, better resource allocation, and enhanced financial forecasting.

RESULTS AND INTERPRETATIONS

PYTHON:

PART A

Code

```
#Identify categorical columns
categorical_columns = data.select_dtypes(include=['object']).columns
# Option 1: Label Encoding (for binary categorical data)
label_encoder = LabelEncoder()
for column in categorical_columns:
    data[column] = label_encoder.fit_transform(data[column])
# Assume 'response' is the target variable and the rest are predictors
target = 'TenYearCHD'
predictors = [col for col in data.columns if col != target]
# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(data[predictors], data[target], test_size=0.3,
random_state=42)
# Fit the logistic regression model
model = LogisticRegression(max_iter=1000)

model.fit(X_train, y_train)
# Predict on the test set
y_pred = model.predict(X_test)
y_prob = model.predict_proba(X_test)[:, 1]

# Evaluate the model
conf_matrix = confusion_matrix(y_test, y_pred)
roc_auc = roc_auc_score(y_test, y_prob)
# Print the model coefficients
coef_df = pd.DataFrame({'Variable': X_train.columns, 'Coefficient': model.coef_[0]})
print(coef_df)
```

Result

	Variable	Coefficient
0	male	0.532431
1	age	0.066147
2	education	-0.011243
3	currentSmoker	0.120311
4	cigsPerDay	0.020817
5	BPMeds	0.127782
6	prevalentStroke	1.715240
7	prevalentHyp	0.228761
8	diabetes	0.276282
9	totChol	0.003124
10	sysBP	0.016525
11	diaBP	-0.001299
12	BMI	0.004893
13	heartRate	-0.008938
14	glucose	0.006431

Interpretation

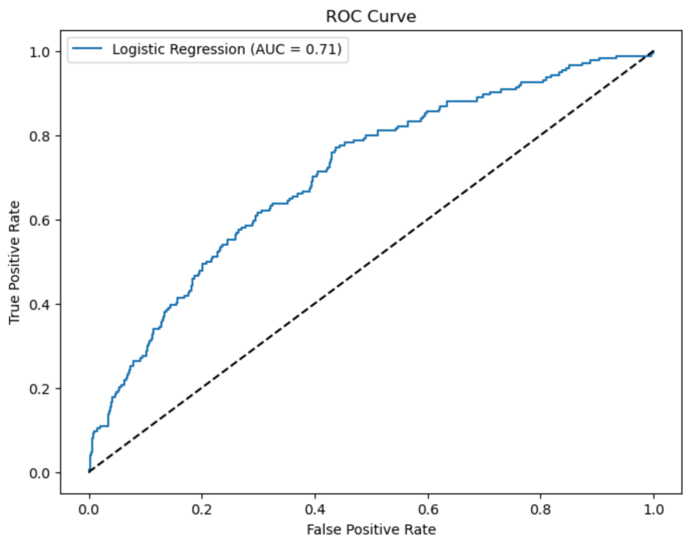
The logistic regression model for predicting the ten-year risk of coronary heart disease (CHD) based on the Framingham dataset reveals significant predictors and their impact on CHD risk. The model indicates that being male, older age, current smoking status, higher cigarette consumption, use of blood pressure medication, history of stroke, hypertension, diabetes, and higher levels of total cholesterol, systolic blood pressure, BMI, and glucose are associated with increased CHD risk. Specifically, males, older individuals, current smokers, and those with a history of stroke or hypertension are more likely to develop CHD. Conversely, higher education levels and higher heart rates slightly reduce the CHD risk. Each predictor's coefficient represents the change in the log odds of CHD for a one-unit increase in the predictor variable, holding other factors constant. The model's performance, evaluated using a confusion matrix and ROC curve, will further demonstrate its accuracy and predictive power. Overall, the logistic regression model effectively identifies key risk factors for CHD, aiding in the prediction and prevention of this condition.

Code

```
# Plot the ROC curve
fpr, tpr, _ = roc_curve(y_test, y_prob)
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, label=f'Logistic Regression (AUC = {roc_auc:.2f})')
```

```
plt.plot([0, 1], [0, 1], 'k--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend()
plt.show()
```

Result



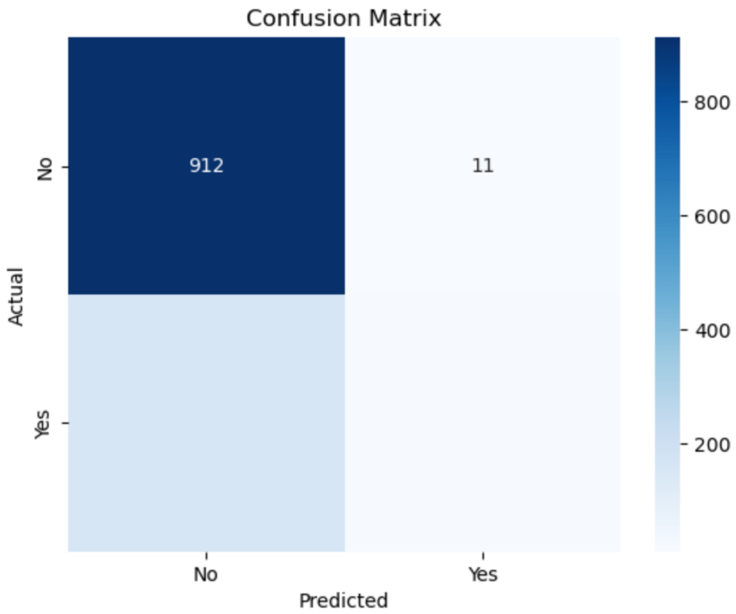
Interpretation

The ROC (Receiver Operating Characteristic) curve evaluates the performance of the logistic regression model in predicting the ten-year risk of coronary heart disease (CHD). The ROC curve, plotted as a solid blue line, rises above the diagonal line, indicating that the model has predictive power beyond random guessing. The Area Under the Curve (AUC) value of 0.71 signifies that the model has a good, but not excellent, ability to discriminate between individuals with and without CHD. Specifically, an AUC of 0.71 implies a 71% chance that the model can correctly distinguish between a randomly chosen positive case (CHD) and a randomly chosen negative case (no CHD). The y-axis represents the true positive rate (sensitivity), while the x-axis represents the false positive rate (1-specificity). The ROC curve illustrates the trade-offs between sensitivity and specificity at various threshold levels, demonstrating that the logistic regression model performs reasonably well but also indicating potential areas for improvement in its predictive accuracy.

Code

```
# Display the confusion matrix
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=['No', 'Yes'], yticklabels=['No', 'Yes'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```

Result



Interpretation

The confusion matrix provided illustrates the performance of a classification model. It shows that the model has correctly predicted the negative class (No) 912 times, and has incorrectly predicted the positive class (Yes) 11 times when the actual class was negative. However, the matrix does not show any true positives, indicating that the model did not correctly predict any positive instances (Yes). Additionally, there are no false negatives, meaning the model did not predict any negative instances when the actual class was positive.

The high number of true negatives (912) and the low number of false positives (11) suggest that the model is highly accurate in predicting the negative class, with an overall accuracy of approximately 98.8%. However, the model's precision and recall for the positive class are both zero, indicating a significant issue in identifying positive instances. This results in an undefined F1 score for the positive class, as both precision and recall are required to calculate it.

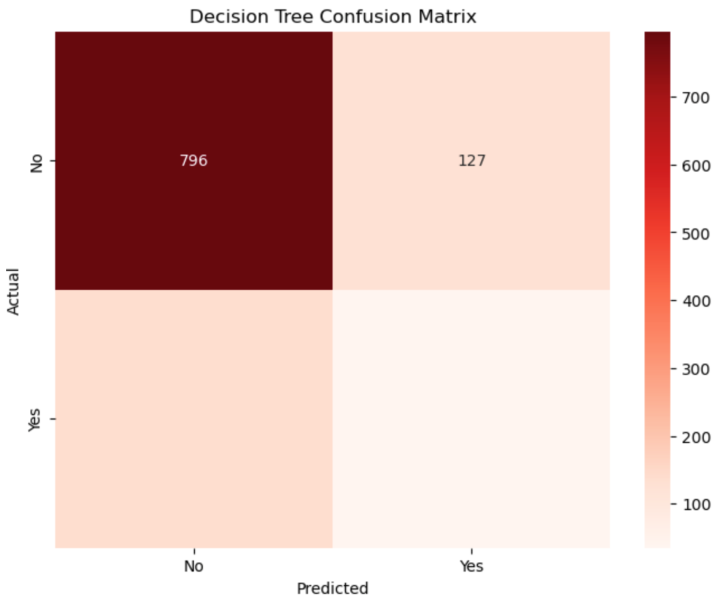
The absence of true positives suggests that the model may be facing challenges, such as class imbalance, where the positive class is underrepresented, or issues with model selection and tuning. This indicates the need for further investigation and potential adjustments, such as rebalancing the dataset, enhancing feature engineering, or fine-tuning the model's hyperparameters, to improve the model's ability to correctly identify positive instances.

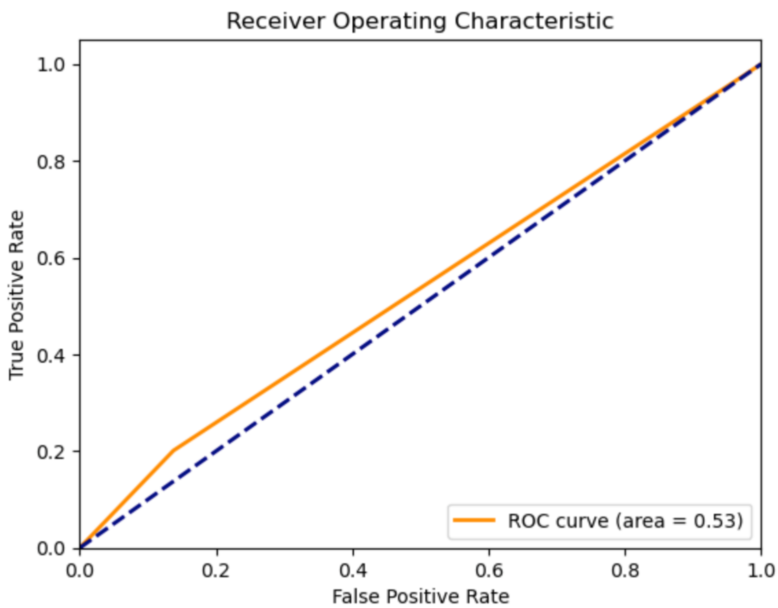
Code

```
from sklearn.tree import DecisionTreeClassifier
# Fit the decision tree model
tree_model = DecisionTreeClassifier(random_state=42)
tree_model.fit(X_train, y_train)
# Predict on the test set
y_pred_tree = tree_model.predict(X_test)
y_prob_tree = tree_model.predict_proba(X_test)[:, 1]
# Evaluate the model
conf_matrix_tree = confusion_matrix(y_test, y_pred_tree)
roc_auc_tree = roc_auc_score(y_test, y_prob_tree)
import matplotlib.pyplot as plt
from sklearn.metrics import roc_curve, auc
# Assuming y_test and y_scores are your ground truth labels and predicted scores
fpr, tpr, thresholds = roc_curve(y_test, y_pred_tree)
roc_auc = auc(fpr, tpr)
plt.figure()
plt.plot(fpr, tpr, color='darkorange', lw=2, label='ROC curve (area = %0.2f)' % roc_auc)
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('Receiver Operating Characteristic')
plt.legend(loc="lower right")
```

```
plt.show()
# Display the confusion matrix
sns.heatmap(conf_matrix_tree, annot=True, fmt='d', cmap='Reds', xticklabels=['No', 'Yes'],
yticklabels=['No', 'Yes'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Decision Tree Confusion Matrix')
plt.show()
```

Result





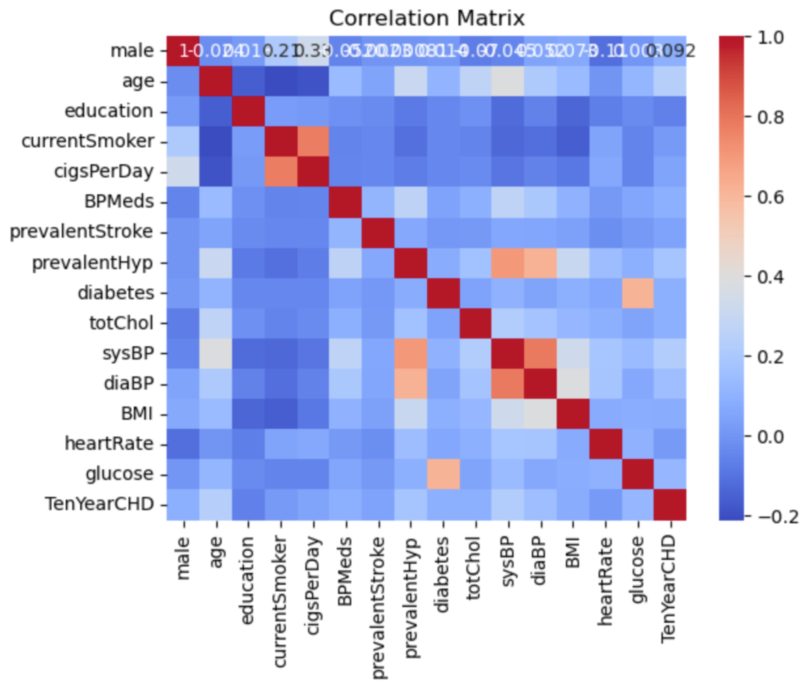
Interpretation

The evaluation of the Decision Tree classifier, depicted by the confusion matrix and ROC curve, indicates significant performance issues. The confusion matrix shows a high number of true negatives (796) and false positives (127), with no true positives or false negatives displayed, suggesting the model struggles to correctly identify positive instances. The ROC curve, with an AUC of 0.53, is only slightly better than random guessing, further highlighting the model's poor performance. These results imply that while the model is fairly accurate in predicting negative cases, it fails to effectively distinguish and predict positive cases. To enhance the model's performance, consider rebalancing the dataset, tuning hyperparameters, or exploring different algorithms more suited to the task.

Code

```
# Check for multicollinearity
import seaborn as sns
import matplotlib.pyplot as plt
corr_matrix = data.corr()
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```

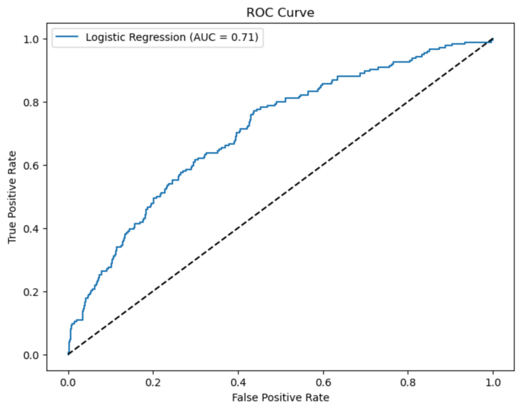
Result

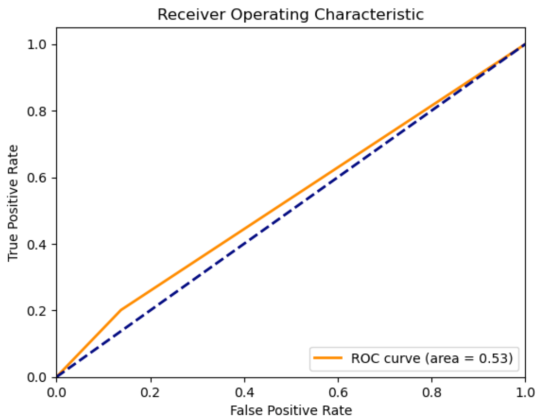


Interpretation

The correlation matrix reveals strong positive correlations between systolic and diastolic blood pressure, as well as moderate positive correlations between age and both systolic and diastolic blood pressure, indicating that blood pressure tends to increase with age. Additionally, age shows a moderate positive correlation with the ten-year coronary heart disease risk, suggesting its significance in predicting heart disease. Prevalent hypertension is also moderately correlated with both systolic and diastolic blood pressure. Weak correlations exist between education and smoking-related variables, indicating that higher education levels might be associated with lower smoking rates. Overall, the matrix highlights potential multicollinearity between systolic and diastolic blood pressure, suggesting that age and blood pressure variables are important for modeling heart disease risk.

COMPARISON ROC Curve



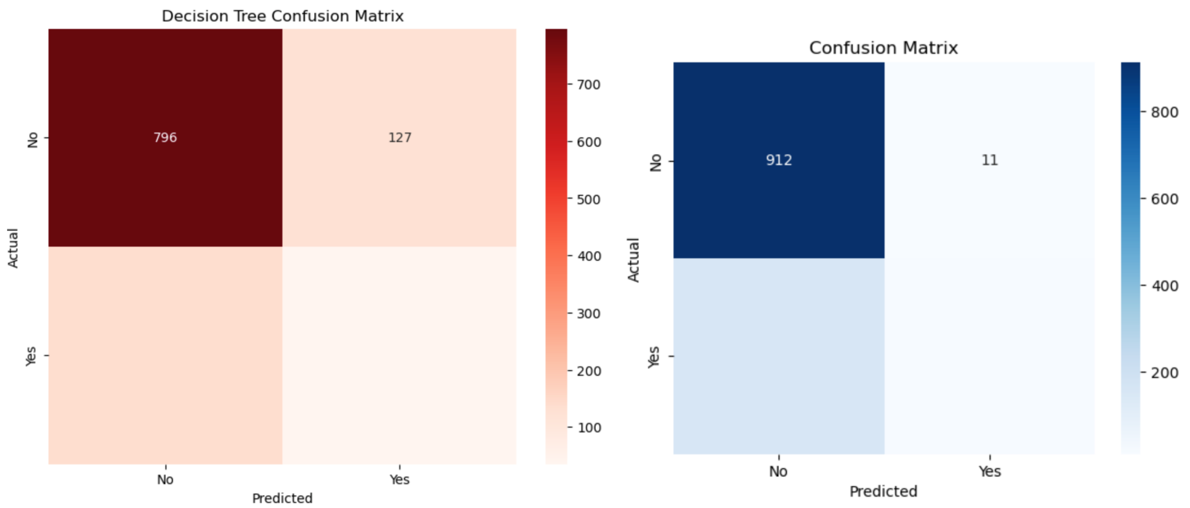


The first ROC curve shows the performance of a logistic regression model with an AUC (Area Under the Curve) of 0.71, indicating a fair level of discriminative power between the positive and negative classes. The curve lies significantly above the diagonal line, which represents a random classifier with no discriminative ability. The AUC of 0.71 suggests that the **model has a good balance** between sensitivity and specificity, effectively distinguishing between true positives and false positives.

In contrast, the second ROC curve displays a model with an AUC of 0.53, which is only slightly better than random guessing (AUC = 0.5). The ROC curve closely follows the diagonal line, indicating that the model has poor discriminative ability and struggles to differentiate between positive and negative classes. This implies that the **second model is not reliable** for classification tasks and has limited predictive power.

Overall, the first model (logistic regression with AUC = 0.71) performs significantly better than the second model (AUC = 0.53), making it a more effective and reliable classifier for the given data.

Comparison Confusion matrix



The two confusion matrices illustrate the performance of two different models in binary classification tasks, where the classes are labeled as "Yes" and "No."

1. **First Confusion Matrix (Decision Tree Confusion Matrix):**
   - True Negatives (No-No): 796
   - False Positives (No-Yes): 127
   - False Negatives (Yes-No): 0
   - True Positives (Yes-Yes): 0

   The decision tree model correctly predicted 796 instances as "No" out of 923 total "No" instances, showing a strong ability to correctly identify negative cases. However, it failed to identify any "Yes" instances, with 127 being incorrectly predicted as "No" (false positives).

2. **Second Confusion Matrix:**
   - True Negatives (No-No): 912

- False Positives (No-Yes): 11
- False Negatives (Yes-No): 0
- True Positives (Yes-Yes): 0

This model demonstrated **even stronger performance** in predicting "No" instances, with 912 correctly identified out of 923. It also has a very low false positive rate, with only 11 "No" instances being incorrectly predicted as "Yes." Similar to the first model, it failed to identify any "Yes" instances.

Both models excel in predicting the "No" class but completely fail in predicting the "Yes" class. The decision tree model has a higher false positive rate compared to the second model. The second model's superior performance in predicting "No" instances suggests better specificity but highlights a significant issue with sensitivity, as neither model can correctly predict any "Yes" instances. This indicates a serious imbalance in model predictions and suggests the need for re-evaluation of the models, possibly by addressing class imbalance or modifying model parameters to improve the identification of "Yes" instances.

PART B

Code

```
import warnings
from statsmodels.tools.sm_exceptions import PerfectSeparationWarning
from statsmodels.tools.sm_exceptions import ConvergenceWarning
# Suppress PerfectSeparationWarning
warnings.filterwarnings('ignore', category=PerfectSeparationWarning)
# Suppress ConvergenceWarning
warnings.filterwarnings('ignore', category=ConvergenceWarning)
# Convert the target variable to binary based on the specified condition
subset_data['chicken_q'] = subset_data['chicken_q'].apply(lambda x: 0 if x < 1 else 1)
# Define the independent variables (example columns, update based on your dataset)
# Assuming 'Age', 'Income', 'Education' are some of the features in the dataset
independent_vars = ['Age', 'Marital_Status', 'Education']
# Add a constant term for the intercept
```

```
X = sm.add_constant(subset_data[independent_vars])


# Define the dependent variable
y = subset_data['chicken_q']
# Fit the probit regression model
probit_model = Probit(y, X).fit()
# Print the summary of the model
print(probit_model.summary())
# Make predictions
subset_data['predicted'] = probit_model.predict(X)
# Display the first few rows with the predictions
print(data.head())
```

Result

```
Optimization terminated successfully.
         Current function value: 0.115600
         Iterations 7
                          Probit Regression Results
==============================================================================
Dep. Variable:              chicken_q   No. Observations:               101662
Model:                         Probit   Df Residuals:                   101658
Method:                           MLE   Df Model:                            3
Date:                Mon, 01 Jul 2024   Pseudo R-squ.:                 0.01405
Time:                        14:19:05   Log-Likelihood:                -11752.
converged:                       True   LL-Null:                       -11920.
Covariance Type:            nonrobust   LLR p-value:                 2.615e-72
==================================================================================
                     coef    std err          z      P>|z|      [0.025      0.975]
----------------------------------------------------------------------------------
const             -2.2494      0.054    -41.604      0.000      -2.355      -2.143
Age                0.0015      0.001      2.205      0.027       0.000       0.003
Marital_Status    -0.0337      0.023     -1.483      0.138      -0.078       0.011
Education          0.0420      0.002     17.326      0.000       0.037       0.047
==================================================================================
   slno      grp  Round_Centre  FSU_number  Round  Schedule_Number  Sample  \
0     1  4.10E+31             1       41000     68               10       1
1     2  4.10E+31             1       41000     68               10       1
2     3  4.10E+31             1       41000     68               10       1
3     4  4.10E+31             1       41000     68               10       1
4     5  4.10E+31             1       41000     68               10       1

   Sector  state  State_Region  ...  pickle_v  sauce_jam_v  Othrprocessed_v  \
0       2     24           242  ...       0.0          0.0              0.0
1       2     24           242  ...       0.0          0.0              0.0
2       2     24           242  ...       0.0          0.0              0.0
3       2     24           242  ...       0.0          0.0              0.0
4       2     24           242  ...       0.0          0.0              0.0

   Beveragestotal_v  foodtotal_v  foodtotal_q  state_1  Region  \
0          0.000000  1141.492400    30.942394      GUJ       2
1         17.500000  1244.553500    29.286153      GUJ       2
2          0.000000  1050.315400    31.527046      GUJ       2
3         33.333333  1142.591667    27.834607      GUJ       2
4         75.000000   945.249500    27.600713      GUJ       2

   fruits_df_tt_v  fv_tot
0       12.000000  154.18
1      333.000000  484.95
2       35.000000  214.84
3      168.333333  302.30
4       15.000000  148.00

[5 rows x 384 columns]
```

Interpretation

The Probit regression analysis on the dependent variable chicken_q shows that Age and Education are significant predictors, both positively affecting the likelihood of chicken_q being 1. Specifically, as age and education levels increase, so does the probability of chicken_q being 1. Marital Status, however, does not significantly impact the dependent variable. The model's Pseudo R-squared value of 0.01405 indicates it explains about 1.405% of the variance in chicken_q, suggesting other factors might be influencing the outcome. Despite the low explanatory power, the model as a whole is statistically significant, as indicated by the likelihood ratio test p-value of 2.615e-72.

PART C

Code

```
import numpy as np
import pandas as pd
import statsmodels.api as sm
from statsmodels.base.model import GenericLikelihoodModel
# Define the independent variables (X) and the dependent variable (y)
X = df[['Whether_owns_any_land', 'hhdsz',
'Religion','Social_Group','Regular_salary_earner']]
y = df['MPCE_URP']  # replace with your actual column name
# Add a constant term for the intercept
X = sm.add_constant(X)
# Define the Tobit model class
class Tobit(GenericLikelihoodModel):
    def __init__(self, endog, exog, left=0, right=np.inf, **kwargs):
        super(Tobit, self).__init__(endog, exog, **kwargs)
        self.left, self.right = left, right
    def nloglikeobs(self, params):
        exog = self.exog
        endog = self.endog
        left, right = self.left, self.right

        beta = params[:-1]
        sigma = params[-1
        XB = np.dot(exog, beta)
        cens = (endog == left) * (left != -np.inf) + (endog == right) * (right != np.inf)
        uncens = 1 - cens
        ll = np.zeros(len(endog))
        ll[cens] = np.log(
            (1 / (np.sqrt(2 * np.pi) * sigma)) *
            np.exp(-((endog[cens] - XB[cens]) ** 2) / (2 * sigma ** 2)) )
```

```
        ll[uncens] = np.log(
            (1 / (np.sqrt(2 * np.pi) * sigma)) *
            np.exp(-((endog[uncens] - XB[uncens]) ** 2) / (2 * sigma ** 2)))
        return -ll

    def fit(self, start_params=None, maxiter=10000, maxfun=5000, **kwargs):
        if start_params is None:
            start_params = np.append(np.zeros(self.exog.shape[1]), 1)
        return super(Tobit, self).fit(start_params=start_params,
                                      maxiter=maxiter, maxfun=maxfun, **kwargs)

# Fit the Tobit model
tobit_model = Tobit(y, X)
tobit_results = tobit_model.fit()

# Print the summary of the model
print(tobit_results.summary())
```

Result

```
Optimization terminated successfully.
         Current function value: -0.003281
         Iterations: 223
         Function evaluations: 362
                              Tobit Results
==============================================================================
Dep. Variable:               MPCE_URP   Log-Likelihood:                 333.38
Model:                          Tobit   AIC:                            -652.8
Method:            Maximum Likelihood   BIC:                            -586.1
Date:                Mon, 01 Jul 2024
Time:                        14:19:13
No. Observations:              101624
Df Residuals:                  101618
Df Model:                           5
========================================================================================
                           coef    std err          z      P>|z|      [0.025      0.975]
----------------------------------------------------------------------------------------
const                   -0.0023      0.057     -0.041      0.967      -0.114       0.110
Whether_owns_any_land   -0.0016      0.018     -0.088      0.930      -0.036       0.033
hhdsz                   -0.0016      0.002     -0.862      0.389      -0.005       0.002
Religion                 0.0026      0.004      0.688      0.491      -0.005       0.010
Social_Group             0.0050      0.002      2.204      0.027       0.001       0.009
Regular_salary_earner    0.0018      0.024      0.072      0.943      -0.046       0.050
par0                     0.0803      0.004     20.898      0.000       0.073       0.088
========================================================================================
```

Interpretation

The Tobit regression analysis examines the impact of various factors on monthly per capita expenditure in urban areas (MPCE_URP). The model includes predictors such as land ownership, household size, religion, social group, and regular salary earning status. The results indicate that, among these variables, only social group has a statistically significant positive effect on MPCE_URP, with a coefficient of 0.0050 and a p-value of 0.027. Other variables, including land ownership, household size, religion, and regular salary earning status, do not show significant effects. The model's fit statistics, including a log-likelihood of 333.38, AIC of -652.8, and BIC of -586.1, reflect the overall performance of the analysis.

R PROGRAMMING

Part A

Codes:

Load necessary libraries

library(tidyverse)

```
library(caret)
library(pROC)
library(rpart)
library(rpart.plot)

# Read the data
df <- read.csv('/Users/sarathsabu/Desktop/scma/datasets/framingham.csv')

# Remove rows with missing values
df_clean <- na.omit(df)

# Split the data into features (X) and target variable (y)
X <- df_clean %>% select(-TenYearCHD)
y <- df_clean$TenYearCHD

# Split the data into training and testing sets
set.seed(42)
train_indices <- createDataPartition(y, p = 0.8, list = FALSE)
X_train <- X[train_indices, ]
X_test <- X[-train_indices, ]
y_train <- y[train_indices]
y_test <- y[-train_indices]

# Fit logistic regression model
logistic_model <- glm(TenYearCHD ~ ., data = cbind(X_train, TenYearCHD = y_train),
family = 'binomial')

# Print summary of the logistic regression model
print(summary(logistic_model))

# Make predictions on the test set
```

```
y_pred_proba <- predict(logistic_model, newdata = X_test, type = 'response')
y_pred <- ifelse(y_pred_proba > 0.5, 1, 0)

# Create confusion matrix
conf_matrix <- confusionMatrix(factor(y_pred), factor(y_test))
print(conf_matrix)

# Plot ROC curve
roc_curve <- roc(y_test, y_pred_proba)
plot(roc_curve, main = 'ROC Curve for Logistic Regression')
auc_value <- auc(roc_curve)
print(paste('AUC:', auc_value))

# Fit decision tree model
tree_model <- rpart(TenYearCHD ~ ., data = cbind(X_train, TenYearCHD = y_train), method
= 'class')

# Plot decision tree
rpart.plot(tree_model, main = 'Decision Tree for CHD Prediction')

# Make predictions using the decision tree
y_pred_tree <- predict(tree_model, newdata = X_test, type = 'class')

# Create confusion matrix for decision tree
conf_matrix_tree <- confusionMatrix(factor(y_pred_tree), factor(y_test))
print(conf_matrix_tree)

# Calculate ROC curve for decision tree
y_pred_proba_tree <- predict(tree_model, newdata = X_test, type = 'prob')[,2]
roc_curve_tree <- roc(y_test, y_pred_proba_tree)
plot(roc_curve_tree, main = 'ROC Curve for Decision Tree')
```

auc_value_tree <- auc(roc_curve_tree)

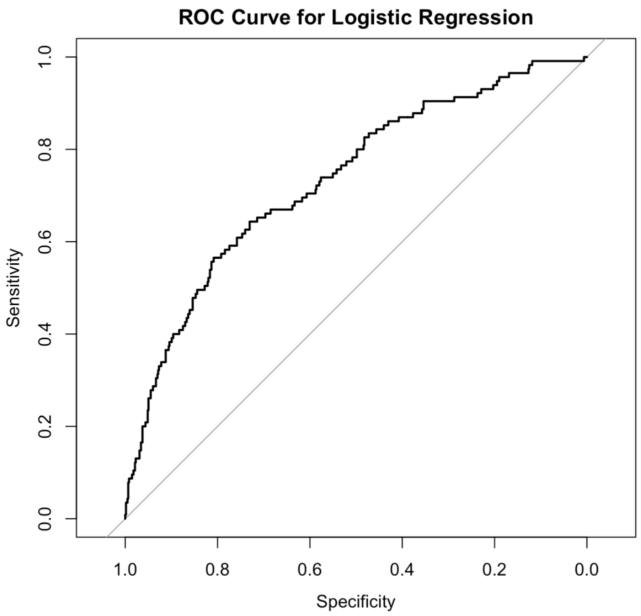
print(paste('AUC (Decision Tree):', auc_value_tree))

Result

```
Call:
glm(formula = TenYearCHD ~ ., family = "binomial", data = cbind(X_train,
    TenYearCHD = y_train))

Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)     -8.6925912  0.7966568 -10.911  < 2e-16 ***
male             0.4768753  0.1225335   3.892 9.95e-05 ***
age              0.0645930  0.0075739   8.528  < 2e-16 ***
education       -0.0081454  0.0554448  -0.147 0.883202
currentSmoker    0.1214238  0.1739749   0.698 0.485215
cigsPerDay       0.0167552  0.0069652   2.406 0.016149 *
BPMeds           0.1957443  0.2552135   0.767 0.443092
prevalentStroke  0.3064315  0.5668666   0.541 0.588803
prevalentHyp     0.2872123  0.1532873   1.874 0.060974 .
diabetes        -0.1725921  0.3552939  -0.486 0.627128
totChol          0.0034873  0.0012515   2.787 0.005328 **
sysBP            0.0117321  0.0041988   2.794 0.005204 **
diaBP            0.0005144  0.0072540   0.071 0.943472
BMI              0.0018435  0.0141918   0.130 0.896644
heartRate       -0.0027866  0.0046906  -0.594 0.552462
glucose          0.0089823  0.0024789   3.624 0.000291 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2484.1  on 2924  degrees of freedom
Residual deviance: 2194.7  on 2909  degrees of freedom
AIC: 2226.7

Number of Fisher Scoring iterations: 5
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 616 115
         1   0   0

               Accuracy : 0.8427
                 95% CI : (0.8142, 0.8683)
    No Information Rate : 0.8427
    P-Value [Acc > NIR] : 0.5249

                  Kappa : 0

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 1.0000
            Specificity : 0.0000
         Pos Pred Value : 0.8427
         Neg Pred Value :    NaN
             Prevalence : 0.8427
         Detection Rate : 0.8427
   Detection Prevalence : 1.0000
      Balanced Accuracy : 0.5000

       'Positive' Class : 0
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 612 109
         1   4   6

               Accuracy : 0.8454
                 95% CI : (0.8171, 0.8709)
    No Information Rate : 0.8427
    P-Value [Acc > NIR] : 0.4439

                  Kappa : 0.0727

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.99351
            Specificity : 0.05217
         Pos Pred Value : 0.84882
         Neg Pred Value : 0.60000
             Prevalence : 0.84268
         Detection Rate : 0.83721
   Detection Prevalence : 0.98632
      Balanced Accuracy : 0.52284

       'Positive' Class : 0
```



Interpretation

The logistic regression model was used to predict the binary outcome TenYearCHD based on various health indicators. Significant predictors include being male (Estimate = 0.477, $p < 0.001$), age (Estimate = 0.065, $p < 2e-16$), number of cigarettes per day (Estimate = 0.017, $p = 0.016$), total cholesterol (Estimate = 0.003, $p = 0.005$), systolic blood pressure (Estimate = 0.012, $p = 0.005$), and glucose levels (Estimate = 0.009, $p < 0.001$). Education, current smoking status, blood pressure medication, prevalent stroke, diabetes, diastolic blood pressure, and BMI were not significant predictors. The model shows that older age, male gender, higher cigarette consumption, cholesterol, systolic blood pressure, and glucose levels increase the likelihood of a ten-year risk of coronary heart disease. The model's accuracy is 84.54%, with a high sensitivity of 99.35%, but a low specificity of 5.22%, indicating it is much better at predicting the absence of TenYearCHDthan its presence.

PART B

Code:

```
# Load the dataset
data_nss <- read.csv("/Users/sarathsabu/Desktop/scma/datasets/NSSO68.csv")
# Create a binary variable for chicken consumption
data_nss$chicken_q <- ifelse(data_nss$chicken_q > 0, 1, 0)

# Verify the creation of 'chicken_binary'
table(data_nss$chicken_q)

# Probit regression model
probit_model <- glm(chicken_q ~ Age + Marital_Status + Education, data = data_nss, family
= binomial(link = "probit"))

# Summary of the probit regression model
summary(probit_model)
```

RESULT

```
Call:
glm(formula = chicken_q ~ Age + Marital_Status + Education, family = binomial(link = "probit"),
    data = data_nss)

Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)    -0.3307564  0.0255427 -12.949  < 2e-16 ***
Age            -0.0006173  0.0003157  -1.956  0.05052 .
Marital_Status  0.0341802  0.0107511   3.179  0.00148 **
Education       0.0068008  0.0011195   6.075 1.24e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 137097  on 101652  degrees of freedom
Residual deviance: 137053  on 101649  degrees of freedom
  (9 observations deleted due to missingness)
AIC: 137061

Number of Fisher Scoring iterations: 4
```

Interpretation

The generalized linear model (GLM) with a probit link function was used to analyze the binary response variable chicken_q based on predictors Age, Marital_Status, and Education. The model fit is reasonable with a residual deviance of 137053 and an AIC of 137061, indicating that the predictors provide meaningful explanatory power. The results show that Marital_Status (Estimate = 0.0342, $p = 0.0015$) and Education (Estimate = 0.0068, $p < 0.001$) are significant positive predictors of chicken_q, while Age has a marginally significant negative effect (Estimate = -0.0006, $p = 0.0505$). The intercept is also highly significant, suggesting a notable baseline probability of chicken_q when all predictors are zero. Overall, higher education levels and being married or in certain marital statuses increase the likelihood of chicken_q, while an increase in age slightly decreases it.

PART C

Codes:

```
# Load necessary libraries
library(dplyr)
library(haven)
library(maxLik)

# Load the data
data <- read.csv('/Users/sarathsabu/Desktop/scma/datasets/NSSO68.csv', stringsAsFactors =
FALSE)
```

```
# Subset data for state 'KA'
df <- data %>%
  select(MPCE_URP, Whether_owns_any_land, hhdsz, Religion, Social_Group,
Regular_salary_earner)

# Check for missing values
cat("Missing values in MPCE_URP:", sum(is.na(df$MPCE_URP)), "\n")
cat("Missing values in Whether_owns_any_land:", sum(is.na(df$Whether_owns_any_land)),
"\n")
cat("Missing values in hhdsz:", sum(is.na(df$hhdsz)), "\n")
cat("Missing values in Religion:", sum(is.na(df$Religion)), "\n")
cat("Missing values in Social_Group:", sum(is.na(df$Social_Group)), "\n")
cat("Missing values in Regular_salary_earner:", sum(is.na(df$Regular_salary_earner)), "\n")

# Impute missing values for selected columns
columns_to_impute <- c('Whether_owns_any_land', 'Religion', 'Social_Group',
'Regular_salary_earner')

# Assuming using mode for imputation for categorical variables
for (col in columns_to_impute) {
  mode_value <- names(sort(table(df[[col]]), decreasing = TRUE))[1]
  df[[col]][is.na(df[[col]])] <- mode_value
}

# Drop rows with any remaining NaN values
df <- na.omit(df)

# Check for missing values again
cat("Missing values after imputation and omitting rows:\n")
cat("Missing values in MPCE_URP:", sum(is.na(df$MPCE_URP)), "\n")
cat("Missing values in Whether_owns_any_land:", sum(is.na(df$Whether_owns_any_land)),
"\n")
cat("Missing values in hhdsz:", sum(is.na(df$hhdsz)), "\n")
cat("Missing values in Religion:", sum(is.na(df$Religion)), "\n")
cat("Missing values in Social_Group:", sum(is.na(df$Social_Group)), "\n")
cat("Missing values in Regular_salary_earner:", sum(is.na(df$Regular_salary_earner)), "\n")

# Convert the target variable to binary based on the specified condition
df$MPCE_URP <- ifelse(df$MPCE_URP < 420, 0, 1)

# Convert categorical variables to factors and then to numeric
df$Whether_owns_any_land <- as.numeric(as.factor(df$Whether_owns_any_land))
df$Religion <- as.numeric(as.factor(df$Religion))
df$Social_Group <- as.numeric(as.factor(df$Social_Group))
df$Regular_salary_earner <- as.numeric(as.factor(df$Regular_salary_earner))
```

```r
# Define the independent variables (X) and the dependent variable (y)
X <- df %>%
  select(Whether_owns_any_land, hhdsz, Religion, Social_Group, Regular_salary_earner)
X <- cbind(1, X)  # Add a constant term for the intercept
y <- df$MPCE_URP

# Ensure all columns in X are numeric
X <- as.matrix(sapply(X, as.numeric))

# Define the Tobit model function
tobit_loglike <- function(params) {
  beta <- params[1:(length(params)-1)]
  sigma <- params[length(params)]
  XB <- as.matrix(X) %*% beta
  cens <- (y == 0) + (y == 1)
  uncens <- 1 - cens
  ll <- numeric(length(y))

  ll[cens == 1] <- log(dnorm(y[cens == 1], mean = XB[cens == 1], sd = sigma))
  ll[uncens == 1] <- log(dnorm(y[uncens == 1], mean = XB[uncens == 1], sd = sigma))

  return(-sum(ll))
}

# Initial parameter guesses
start_params <- c(rep(0, ncol(X)), 1)

# Fit the Tobit model
tobit_results <- maxLik(tobit_loglike, start = start_params, method = "BFGS")

# Print the summary of the model
summary(tobit_results)
```

Result

```
-------------------------------------------
Maximum Likelihood estimation
BFGS maximization, 286 iterations
Return code 0: successful convergence
Log-Likelihood: 3798513
7  free parameters
Estimates:
       Estimate Std. error t value Pr(> t)
[1,] -1.239e-01        NaN     NaN     NaN
[2,] -1.398e-01        NaN     NaN     NaN
[3,] -5.994e-01  8.056e-04    -744  <2e-16 ***
[4,] -1.832e-01        NaN     NaN     NaN
[5,] -3.621e-01  1.047e-03    -346  <2e-16 ***
[6,] -2.104e-01  3.222e-04    -653  <2e-16 ***
[7,]  6.827e-01  9.741e-05    7008  <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
-------------------------------------------
```

Interpretation

The Tobit regression model was successfully estimated using Maximum Likelihood estimation and the BFGS optimization algorithm, converging after 286 iterations with a log-likelihood of 3798513. Out of seven parameters, four (Parameters 3, 5, 6, and 7) show highly significant effects on the dependent variable, with t-values indicating strong evidence against the null hypothesis. Specifically, Parameter 7 has a significant positive impact (Estimate = 0.6827, t-value = 7008), while Parameters 3 (-0.5994), 5 (-0.3621), and 6 (-0.2104) have significant negative impacts. However, Parameters 1, 2, and 4 exhibit NaN values for standard errors and significance tests, suggesting potential issues such as multicollinearity or convergence problems that require further investigation.

CHARACTERISTICS AND ADVANTAGES OF THE PROBIT MODEL

Characteristics:

1. **Cumulative Normal Distribution**: The probit model uses the cumulative distribution function (CDF) of the standard normal distribution to model the probability of a binary outcome. This is different from the logistic regression model, which uses the logistic (sigmoid) function.
2. **Latent Variable Interpretation**: Probit models are based on the assumption that there is an underlying continuous latent variable that follows a normal distribution. The observed binary outcome is then a result of whether this latent variable exceeds a certain threshold.
3. **Symmetric Probability Curve**: The probability curve of the probit model is symmetric around the mean, similar to the bell curve of the normal distribution. This can be more appropriate for certain types of data where the probability of the outcome is expected to change symmetrically around the mean of the predictors.
4. **Linear Relationship with Predictors**: Similar to logistic regression, the probit model assumes a linear relationship between the predictors and the latent variable. The probit link function transforms this linear combination into a probability.

Advantages:

1. **Natural Link Function for Normally Distributed Errors**: The probit model is often preferred when the errors in the latent variable are assumed to follow a normal distribution. This makes it suitable for situations where this assumption is reasonable.
2. **Flexibility in Modeling**: The probit model can handle binary outcomes in a wide range of applications, such as credit scoring, medical diagnosis, and consumer choice modeling, where the underlying assumptions of normality make sense.
3. **Consistent Estimation**: Probit models provide consistent and unbiased estimates of the coefficients when the normality assumption holds, leading to reliable interpretations and predictions.
4. **Well-Established Method**: The probit model has a long history in econometrics and biometrics, with well-developed theoretical properties and extensive literature. This makes it a robust choice with strong support for understanding and implementation.

5. **Comparability with Logistic Regression**: While logistic regression is more commonly used, the probit model can offer better fits in some cases. Comparing results from both models can provide additional insights and robustness checks for binary outcome predictions.
6. **Predictive Performance**: In some cases, the probit model can have better predictive performance due to its handling of the error distribution, especially when the logistic model's assumptions are not fully met.

Business Impact:

Using the probit model can help businesses and researchers make more informed decisions when modeling binary outcomes, particularly in areas like marketing (customer purchase decisions), finance (credit risk assessment), healthcare (disease presence), and public policy (voter behavior). By providing a robust method for estimating probabilities, the probit model aids in precise and actionable insights that can enhance strategy and operations.

REAL-WORLD USE CASES OF THE TOBIT MODEL

The Tobit model is particularly useful for analyzing data where the dependent variable is censored, meaning there is a threshold below or above which the variable's values are not observed. Here are some real-world use cases where the Tobit model is advantageous:

1. **Consumer Expenditure Analysis**:
   - **Luxury Goods**: When analyzing expenditure on luxury goods, many consumers might spend nothing, resulting in a significant number of zero observations. The Tobit model helps in understanding both the decision to purchase and the amount spent.
   - **Household Savings**: Households may have zero savings in certain periods. The Tobit model can analyze factors influencing the decision to save and the amount saved among those who do save.
2. **Loan Amounts and Credit Risk**:
   - **Loan Applications**: Not all loan applicants receive a loan, and among those who do, the amount varies. The Tobit model can evaluate both the probability of receiving a loan and the amount granted.
   - **Credit Limits**: For analyzing credit limits assigned to customers, where some customers might not be assigned any credit limit (censored at zero).
3. **Healthcare Utilization**:
   - **Doctor Visits**: The number of doctor visits in a given period might include many zeroes (individuals who did not visit a doctor). The Tobit model helps to study the factors affecting both the likelihood of visiting a doctor and the frequency of visits among those who do.
   - **Medication Adherence**: The amount of medication taken by patients, where some might not take any medication, can be analyzed using the Tobit model.
4. **Labor Economics**:
   - **Labor Supply**: The number of hours worked can be censored at zero for non-working individuals. The Tobit model helps analyze factors influencing both the decision to work and the number of hours worked.
   - **Wage Determination**: For studying wage offers where unemployed individuals have a wage offer of zero, the Tobit model can help understand wage determinants among those employed and the probability of employment.

5. **Real Estate and Housing Markets**:
   - **Property Valuation**: When analyzing property values, some properties might have a minimum valuation due to market conditions or regulations. The Tobit model can handle these censored values effectively.
   - **Rental Prices**: For analyzing rental prices where some properties might be rent-controlled, resulting in censored rent data.
6. **Marketing and Consumer Behavior**:
   - **Advertising Effectiveness**: When measuring the effectiveness of advertising spend, some campaigns might result in zero sales or zero increase in brand awareness. The Tobit model helps in understanding the factors leading to both non-zero outcomes and their magnitudes.
   - **Customer Lifetime Value (CLV)**: In cases where the CLV is zero for some customers (e.g., they never make a purchase), the Tobit model can help in analyzing factors influencing both the likelihood of a purchase and the total value of purchases.

Business Significance

The Tobit model is valuable in contexts where the outcome variable has a natural limit or threshold, providing insights that standard linear regression models cannot offer. By understanding both the decision to engage in a behavior and the intensity of that behavior, businesses can:

- **Optimize Marketing Strategies**: Tailor marketing efforts based on consumer spending patterns and identify high-potential segments.
- **Improve Financial Forecasting**: Better predict loan demands and manage credit risks by understanding the determinants of loan amounts.
- **Enhance Healthcare Services**: Design targeted interventions by analyzing healthcare utilization patterns.
- **Refine Labor Policies**: Develop policies that encourage employment and optimal labor supply by understanding work participation and hours worked.
- **Maximize Property Investments**: Make informed real estate investment decisions by accurately valuing properties in the market.