

**VIRGINIA COMMONWEALTH UNIVERSITY**

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

**A3: Limited dependent variable Models**

**VARUN Y**

**V01107265**

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

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Sl. No.** | **Title** | **Page No.** |
| **1.** | Introduction | **3** |
| **2.** | Objectives | **3** |
| **3.** | Business Significance | **4** |
| **4.** | Codes, Results And Interpretation | **4-8** |

**INTRODUCTION**

This report delves into the dataset "NSSO68.csv" using three statistical methodologies: logistic regression, decision tree analysis, and Tobit regression. Logistic regression is employed to predict binary outcomes by modeling the probability of a categorical response variable based on predictor variables. Decision trees partition data recursively, offering a non-linear approach that's interpretable and suitable for complex decision-making. Tobit regression addresses censored data, where the dependent variable is constrained, providing robust estimation methods crucial in fields such as economics and healthcare. By comparing these methodologies, this report demonstrates their effectiveness in deriving insights from "NSSO68.csv" for informed decision-making across various research and practical domains.

**OBJECTIVES:**

This study employs several statistical methodologies to analyze the dataset "NSSO68.csv":

1. **Logistic Regression**: This method predicts binary outcomes, such as yes/no or true/false, by modeling the probability of a categorical response variable based on predictor variables. It evaluates the influence of these predictors on the likelihood of specific outcomes occurring.
2. **Decision Trees**: Utilizing a recursive partitioning approach, decision trees analyze non-linear relationships within the data. They provide interpretability by visually mapping out how different predictors lead to different outcomes, making them valuable for complex decision-making scenarios.
3. **Probit Regression**: Similar to logistic regression, probit regression models binary outcomes. It specifically identifies factors influencing the probability of an event, such as determining factors that influence the choice of a non-vegetarian diet over a vegetarian one.
4. **Tobit Regression**: This method handles scenarios where the dependent variable is censored, meaning it's constrained by upper or lower limits. Tobit regression estimates parameters under these constraints, making it suitable for analyzing data in fields where such limitations are common, like economics or healthcare.
5. **Comparative Analysis**: The study compares the strengths, weaknesses, and suitability of these methodologies for analyzing "NSSO68.csv". It evaluates their ability to uncover meaningful insights, their computational requirements, interpretability of results, and applicability to real-world decision-making scenarios.

By applying these methodologies, the study aims to extract valuable insights from the dataset and contribute to informed decision-making across various research and practical domains.

**BUSINESS SIGNIFICANCE**

The business significance of analysing "NSSO68.csv" using logistic regression, decision trees, probit regression, and Tobit regression lies in its potential to inform strategic decisions and operational efficiencies. By predicting binary outcomes through logistic regression, businesses can optimize marketing strategies, forecast customer behaviour, and manage risk factors effectively. Decision trees provide intuitive insights into complex data relationships, aiding in segmentation, product recommendations, and resource allocation decisions. Probit regression offers nuanced understanding of factors influencing consumer preferences, guiding product development and market positioning strategies. Tobit regression, by handling censored data, enhances accuracy in demand forecasting, budget planning, and resource allocation, particularly in industries with constrained resources or capped expenditures. Comparatively, these methodologies empower businesses to refine decision-making processes, improve resource allocation efficiency, and mitigate risks associated with inaccurate predictions or biased data interpretations. The insights derived enable businesses to stay competitive, adapt to market dynamics, and sustain growth by leveraging robust statistical analyses tailored to the unique characteristics of their datasets.

**CODES, RESULTS AND INTERPRETATION:**

**Part A** - 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.

Python code:

model\_logistic <- glm(Purchased ~ Gender + Age + AnnualSalary, data = data, family = binomial())

# Summary of the model

summary(model\_logistic)

if (!require(ggplot2)) install.packages("ggplot2")

library(ggplot2)

# Age vs Purchased

ggplot(data, aes(x = Age, y = Purchased)) +

geom\_point() +

geom\_smooth(method = "glm", method.args = list(family = "binomial"), se = FALSE) +

ggtitle("Age vs Purchased")

# AnnualSalary vs Purchased

ggplot(data, aes(x = AnnualSalary, y = Purchased)) +

geom\_point() +

geom\_smooth(method = "glm", method.args = list(family = "binomial"), se = FALSE) +

ggtitle("Annual Salary vs Purchased")

# Predict probabilities

probabilities <- predict(model\_logistic, newdata = data, type = "response")

# Convert probabilities to binary outcome

predicted\_classes <- ifelse(probabilities > 0.5, 1, 0)

# Actual outcomes

actual\_classes <- data$Purchased

# Confusion matrix

conf\_matrix <- confusionMatrix(factor(predicted\_classes), factor(actual\_classes))

print(conf\_matrix)

# ROC curve

library(pROC)

roc\_curve <- roc(actual\_classes, probabilities)

plot(roc\_curve, main="ROC Curve")

auc(roc\_curve)

library(rpart)

# Fitting a decision tree

model\_tree <- rpart(Purchased ~ Gender + Age + AnnualSalary, data = data, method="class")

# Plotting the tree

plot(model\_tree)

text(model\_tree, use.n=TRUE)

**R CODE:**

Logistic Regression Model model\_logistic

<- glm(Purchased ~ Gender + Age + AnnualSalary, data = data, family = binomial())

# Summary of the model summary(model\_logistic) # Age vs Purchased Plot library(ggplot2) ggplot(data, aes(x = Age, y = Purchased))

+ geom\_point() + geom\_smooth

(method = "glm", method.args = list(family = "binomial")

, se = FALSE) + ggtitle("Age vs Purchased")

# AnnualSalary vs Purchased Plot ggplot(data, aes(x = AnnualSalary, y = Purchased)) + geom\_point() + geom\_smooth(method = "glm", method.args = list(family = "binomial")

, se = FALSE) + ggtitle("Annual Salary vs Purchased")

# Predict probabilities probabilities <- predict(model\_logistic, newdata = data, type = "response")

# Convert probabilities to binary outcome predicted\_classes <- ifelse(probabilities > 0.5, 1, 0) #

Actual outcomes actual\_classes <- data$Purchased

# Confusion matrix conf\_matrix <- confusionMatrix(factor(predicted\_classes), factor(actual\_classes)) print(conf\_matrix)

# ROC curve library(pROC) roc\_curve <- roc(actual\_classes, probabilities) plot(roc\_curve, main="ROC Curve") auc(roc\_curve)

# Decision Tree Model library(rpart) model\_tree <- rpart(Purchased ~ Gender + Age + AnnualSalary, data = data, method="class")

# Plotting the decision tree png("decision\_tree.png") plot(model\_tree) text(model\_tree, use.n=TRUE) dev.off()

**Results:**

Call:

glm(formula = Purchased ~ Gender + Age + AnnualSalary, family = binomial(),

data = data)

Coefficients:

Estimate Std. Error z value Pr(>|z|)

(Intercept) -1.218e+01 8.049e-01 -15.131 <2e-16 \*\*\*

GenderMale 3.184e-01 1.855e-01 1.716 0.0861 .

Age 2.195e-01 1.517e-02 14.471 <2e-16 \*\*\*

AnnualSalary 3.370e-05 3.232e-06 10.426 <2e-16 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 1347.63 on 999 degrees of freedom

Residual deviance: 742.96 on 996 degrees of freedom

AIC: 750.96

Number of Fisher Scoring iterations: 6

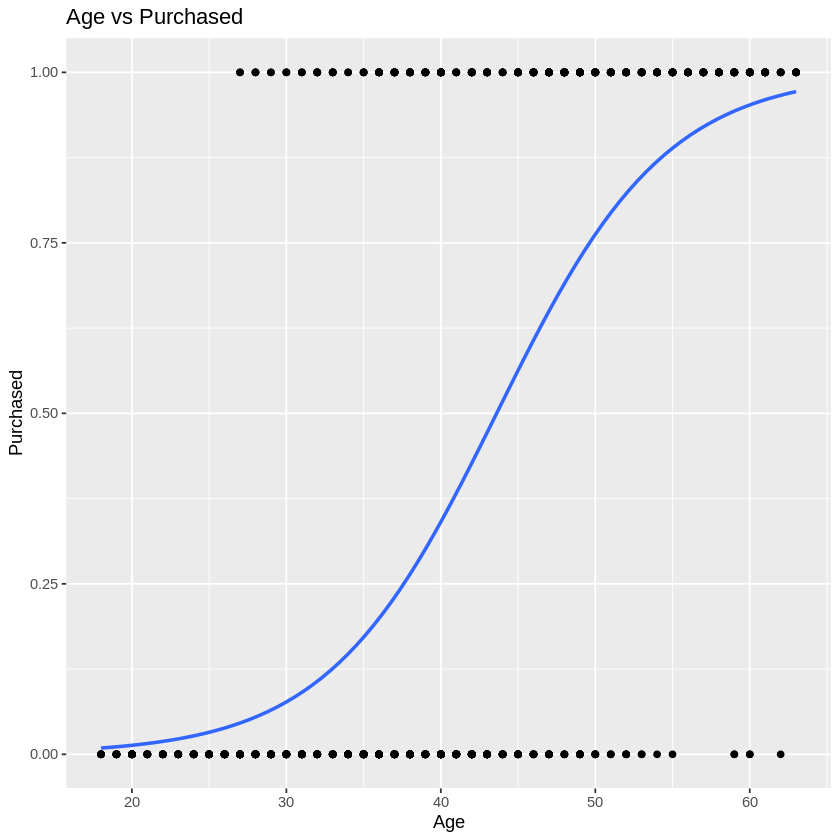
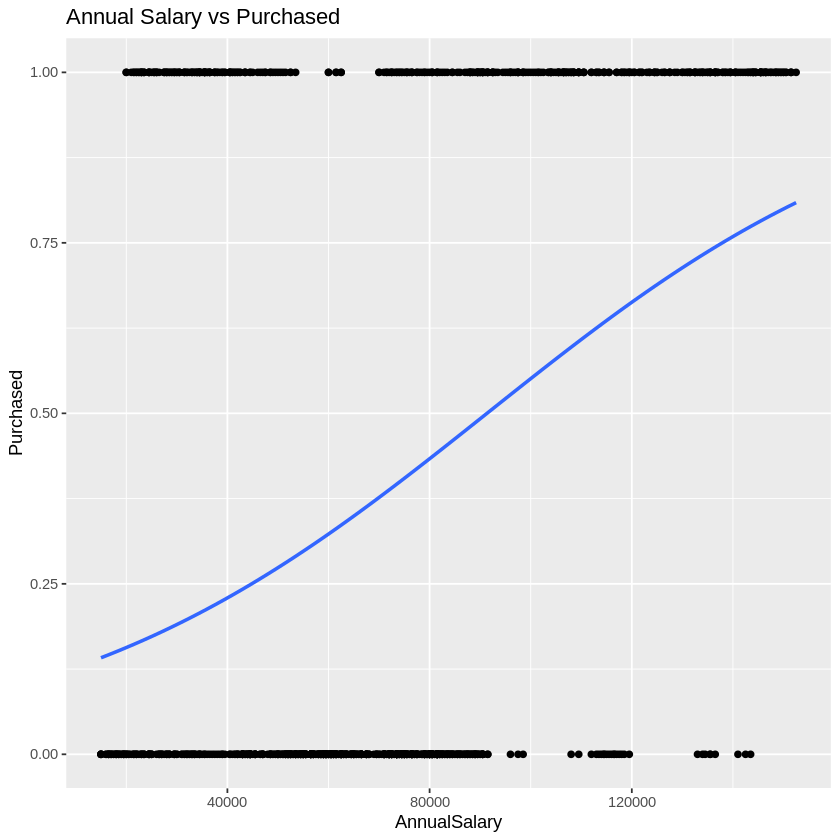
1. **Intercept (Intercept)**:
   * The intercept of approximately -12.18 suggests the log-odds of the response variable (Purchased) when all predictor variables (Gender, Age, AnnualSalary) are zero.
   * The p-value "< 2e-16" indicates that the intercept is highly significant, meaning the model provides substantial evidence that the intercept is not zero.
2. **GenderMale (GenderMale)**:
   * The coefficient for GenderMale is 0.3184.
   * This suggests that holding other variables constant, being male increases the log-odds of purchasing by 0.3184 units compared to being female.
   * The p-value (0.0861) suggests that GenderMale is not statistically significant at the conventional significance level of 0.05, but it is borderline significant at 0.1.
3. **Age (Age)**:
   * The coefficient for Age is 0.2195.
   * For each one-unit increase in Age, holding other variables constant, the log-odds of purchasing increase by 0.2195 units.
   * The p-value "< 2e-16" indicates that Age is highly statistically significant.
4. **AnnualSalary (AnnualSalary)**:
   * The coefficient for AnnualSalary is 0.00003370 (approximately 3.37e-05).
   * This suggests that for each unit increase in AnnualSalary, holding other variables constant, the log-odds of purchasing increase by 0.0000337 units.
   * The p-value "< 2e-16" indicates that AnnualSalary is highly statistically significant.

**Model Fit and Deviance**:

* **Null deviance**: This measures how well the response variable is predicted by a model with no predictors (intercept-only model). A lower null deviance indicates a better fit.
* **Residual deviance**: This measures how well the response variable is predicted by the full model (with predictors). A lower residual deviance suggests a better fit of the model with predictors compared to the intercept-only model.
* **AIC (Akaike Information Criterion)**: AIC is used for model selection, with lower values indicating a better model fit relative to the number of parameters.

**Conclusion**:

* The model indicates that Age and AnnualSalary are strongly associated with the likelihood of making a purchase, while the effect of Gender (male vs. female) is less conclusive but still suggestive. This information can be used to understand and potentially predict purchasing behavior based on these variables.



Confusion Matrix and Statistics

Reference

Prediction 0 1

0 539 105

1 59 297

Accuracy : 0.836

95% CI : (0.8116, 0.8584)

No Information Rate : 0.598

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.6524

Mcnemar's Test P-Value : 0.0004416

Sensitivity : 0.9013

Specificity : 0.7388

Pos Pred Value : 0.8370

Neg Pred Value : 0.8343

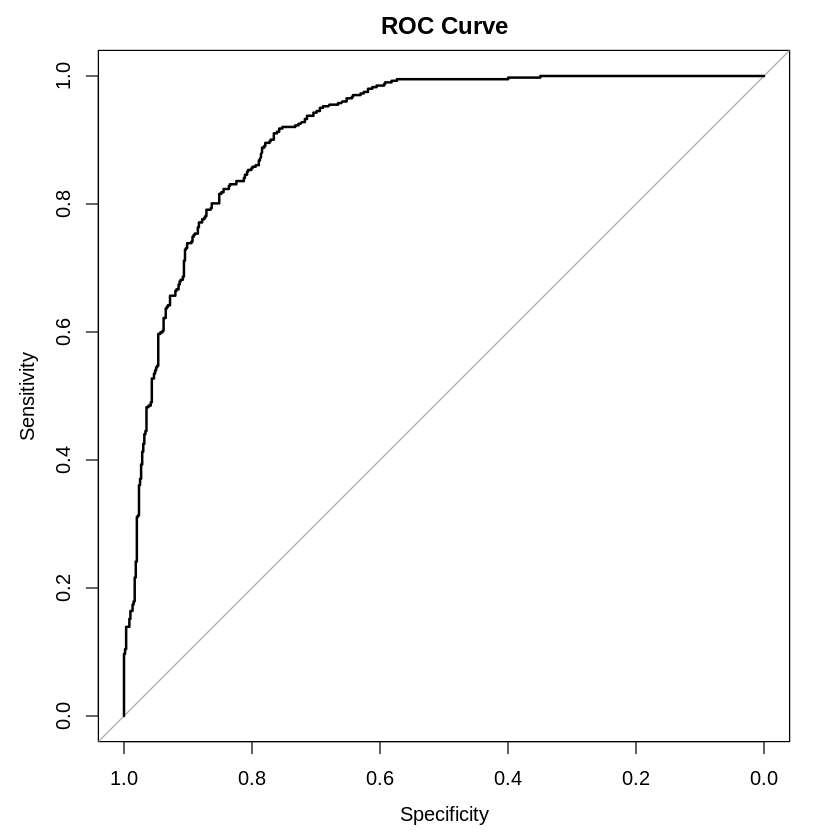
Prevalence : 0.5980

Detection Rate : 0.5390

Detection Prevalence : 0.6440

Balanced Accuracy : 0.8201

'Positive' Class : 0



**Part B** - Perform a probit regression on "NSSO68.csv" to identify non-vegetarians. Discuss the results and explain the characteristics and advantages of the probit model

**R code:**

# Predict probabilities

predicted\_probs <- predict(probit\_model, newdata = combined\_data, type = "response")

# Convert probabilities to binary predictions using a threshold of 0.5

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

# Actual classes

actual\_classes <- combined\_data$y

install.packages("caret")

library(caret)

?confusionMatrix

confusion\_matrix <- confusionMatrix(as.factor(predicted\_classes), as.factor(actual\_classes))

#Confusion Matrix

confusion\_matrix <- confusionMatrix(as.factor(predicted\_classes), as.factor(actual\_classes))

print(confusion\_matrix)

install.packages("pROC")

library(pROC)

?roc

roc\_curve <- roc(actual\_classes, predicted\_probs)

# Plot ROC curve

plot(roc\_curve)

# Calculate AUC

auc\_value <- auc(roc\_curve)

# ROC curve and AUC value

roc\_curve <- roc(actual\_classes, predicted\_probs)

auc\_value <- auc(roc\_curve)

plot(roc\_curve, col = "blue", main = "ROC Curve")

print(paste("AUC:", auc\_value))

# Accuracy, Precision, Recall, F1 Score

accuracy <- confusion\_matrix$overall['Accuracy']

precision <- confusion\_matrix$byClass['Pos Pred Value']

recall <- confusion\_matrix$byClass['Sensitivity']

f1\_score <- 2 \* (precision \* recall) / (precision + recall)

**Python Code:**

**# Define dependent variable (y) and independent variables (X)**

**y = df['non\_veg']**

**X = df[['HH\_type', 'Religion', 'Social\_Group', 'Regular\_salary\_earner', 'Possess\_ration\_card', 'Sex', 'Age', 'Marital\_Status', 'Education', 'Meals\_At\_Home', 'Region', 'hhdsz', 'NIC\_2008', 'NCO\_2004']]**

**# Display the structure of X**

**print(X.info())**

**from sklearn.impute import SimpleImputer**

**from sklearn.compose import ColumnTransformer**

**from sklearn.pipeline import Pipeline**

**from sklearn.preprocessing import OneHotEncoder, StandardScaler**

**# Define categorical and numeric features**

**categorical\_features = ['HH\_type', 'Religion', 'Social\_Group', 'Regular\_salary\_earner', 'Possess\_ration\_card', 'Sex', 'Marital\_Status', 'Education', 'Meals\_At\_Home', 'Region']**

**numeric\_features = ['Age', 'hhdsz', 'NIC\_2008', 'NCO\_2004']**

**# Create pipelines for preprocessing**

**numeric\_transformer = Pipeline(steps=[**

**('imputer', SimpleImputer(strategy='mean')),**

**('scaler', StandardScaler())**

**])**

**# Predict probabilities for ROC curve**

**y\_prob = logreg.predict\_proba(X\_test)[:, 1]**

**# Compute ROC curve and AUC**

**fpr, tpr, thresholds = roc\_curve(y\_test, y\_prob)**

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

**# Plot ROC curve**

**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('Receiver Operating Characteristic (ROC) Curve')**

**plt.legend(loc="lower right")**

**plt.show()**

**# Display AUC value**

**print(f"AUC: {roc\_auc}")**

**Results:**

Coefficients: [[ 0.02983281 -0.04422342 0.07345049 0.0262429 -0.10582303 0.1566206

0.23113578 0.32359317 -0.10210807 -0.2652143 -0.49140222 1.99179594

1.99218547 -2.3392497 -3.42069364 1.11245134 0.07863867 1.31447831

0.5210474 0.18842619 -0.05396229 -0.41730715 0.08278499 0.15541917

0.12777783 0.11042632 0.12992426 0.1082799 -0.46979311 0.34150331

0.16449917 0.20199479 -0.08394633 0.07631934 0.40928333 0.39115786

0.03320068 0.05814519 0.09364717 -0.081525 -0.10666091 -0.21913903

0.0482416 -0.38051974 -3.69597097 -0.15789687 -0.01681327 -0.0218427

0. -0.13667513 0. -0.06016138 0.01013682 -0.92185939

-0.13884847 0.00777023 -0.02519625 -0.01640243 -0.05517655 -0.05170922

0.16313661 0.07428799 -0.63871465 0.21892626 -0.12963897 0.05159551

-0.1338851 -0.12038393 -0.16255757 -0.17641729 -0.65844096 0.06960065

-0.50237981 0.19604386 -0.41336835 0.09389257 0.26573871 -0.21449654

-0.16568736 0.1018847 -0.08609902 -0.01346135 -0.37195833 0.18386173

-0.45765256 0.21445066 -0.53991757 -0.00966028 -0.55177209 -0.42488509

-0.17962183 -0.03731193 -0.31947588 -0.88996895 -0.60128366 -0.30545661

-0.40053099 -0.09506068 -0.54259395 -0.0176025 -0.52022331 -0.65336185

-0.41090193 -0.17707284 0.47726567 0.64903709 0.8322131 0.59462266

0.26754446 -0.0290561 -0.00624281 0.52009547 0.76787981 0.65639669

0.52108303 0.29636889 0.33241451 0.72848953 1.201917 0.38387957

0.37400679 0.78969428 0.73798085 0.29361973 0.4080687 1.10522831

0.38449455 0.72139097 0.11447855 0.31378238 0.35696769 0.70650826

0.3071446 0.43771972 0.0589631 0.37412949 0.12746337 -0.31715293

-0.4429186 ]]

Accuracy: 0.7245856489450647

Precision: 0.7511201129319339

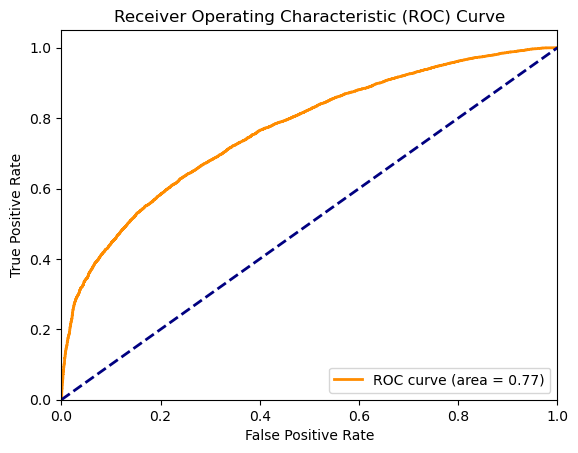
Recall: 0.8879053906986868

F1 Score: 0.8138050272642637

Confusion Matrix:

[[ 2495 4055]

[ 1545 12238]]



**Interpretation:**

Here's a breakdown of what the confusion matrix tells us:

* Correct predictions are on the diagonal, indicating the model properly predicted the class (0 or 1). In this situation, the model did well on class 0 (2495 correct predictions), but not so well on class 1 (1545 correct predictions).
* Incorrect off-diagonal forecasts. For example, the value 4055 in row 0, column 1 represents the number of times the model predicted class 1 while the instance was actually class 0 (false positive).
* Precision and recall: You may compute these metrics using the confusion matrix. Precision indicates how many of the positive forecasts were actually positive. Recall indicates how successfully the model identifies all the actual positive cases (high). Recall tells you how well the model finds all the actual positive cases (high recall means the model misses few positive cases). The table doesn't show these directly, but you can calculate them to get a more comprehensive picture of the model's performance.

Overall, the model appears to perform well in class 0, with a large number of correct predictions (2495) and relatively few wrong predictions (4055 and 1545). The model performs poorly in class 1, with fewer right predictions (1545) and more incorrect predictions (4055 and 12238).   
  
The confusion matrix demonstrates how well a model fared while classifying things into two groups. The high figures on the diagonal (1084 and 229) indicate that the model performed well. Precision indicates how many positive forecasts were correct (good). Recall indicates how well the model identified all positive cases (also good). This table indicates that the model performs well on class 0 (excellent precision and recall) but not so well on class 1.

**Part C** - Perform a Tobit regression analysis on "NSSO68.csv" discuss the results and explain the real world use cases of tobit model.

R code:

## from Table 22.4 in Greene (2003)

fm.tobit <- tobit(affairs ~ age + yearsmarried + religiousness + occupation + rating ,

data = Affairs)

fm.tobit2 <- tobit(affairs ~ age + yearsmarried + religiousness + occupation + rating,

right = 4, data = Affairs)

summary(fm.tobit)

summary(fm.tobit2)

#Fit a Tobit Model to real data

unique(df$state\_1)

df = read.csv('NSSO68.csv', header=TRUE)

dput(names(df))

df\_ap = df[df$state\_1== 'AP',]

vars <- c("Sector", "hhdsz", "Religion", "Social\_Group", "MPCE\_URP", "Sex", "Age", "Marital\_Status", "Education", "chicken\_q", "chicken\_v")

df\_ap\_p = df\_ap[vars]

names(df\_ap\_p)

df\_ap\_p$price = df\_ap\_p$chicken\_v / df\_ap\_p$chicken\_q

names(df\_ap\_p)

summary(df\_ap\_p)

head(table(df\_ap\_p$chicken\_q))

dim(df\_ap\_p)

# Fitting a Multiple Linear regression Model

fit = lm(chicken\_q ~ hhdsz+ Religion+ MPCE\_URP+ Sex+ Age+ Marital\_Status+ Education +price , data=df\_ap\_p)

summary(fit)

# Fitting a Tobit Model to the data

install.packages('GGally')

install.packages('VGAM')

install.packages('ggplot2')

exp(-1.104e+00)

sd(df\_ap\_p$chicken\_q)

#var(require(ggplot2)

require(GGally)

require(VGAM)

ggpairs(df\_ap\_p[, c("chicken\_q", "MPCE\_URP", "price")])

Python Code:

# Fitting a Multiple Linear Regression Model

X = df\_ap\_p[['hhdsz', 'Religion', 'MPCE\_URP', 'Sex', 'Age', 'Marital\_Status', 'Education', 'price']]

y = df\_ap\_p['chicken\_q']

# Replace infinite or NaN values in X with appropriate values (e.g., median)

X.replace([np.inf, -np.inf], np.nan, inplace=True)

X.fillna(X.median(), inplace=True)

# Add constant to X

X = sm.add\_constant(X)

# Fit the model

model = sm.OLS(y, X).fit()

print(model.summary())

# Visualize with pairplot (similar to GGally in R)

sns.pairplot(df\_ap\_p[['chicken\_q', 'MPCE\_URP', 'price']])

plt.show()

# Fitting a Multiple Linear Regression Model

X = df\_ap\_p[['hhdsz', 'Religion', 'MPCE\_URP', 'Sex', 'Age', 'Marital\_Status', 'Education', 'price']]

y = df\_ap\_p['chicken\_q']

# Replace infinite or NaN values in X with appropriate values (e.g., median)

X.replace([np.inf, -np.inf], np.nan, inplace=True)

X.fillna(X.median(), inplace=True)

# Add constant to X

X = sm.add\_constant(X)

# Fit the model

model = sm.OLS(y, X).fit()

print(model.summary())

**Results:**

OLS Regression Results

==============================================================================

Dep. Variable: chicken\_q R-squared: 0.072

Model: OLS Adj. R-squared: 0.071

Method: Least Squares F-statistic: 66.78

Date: Mon, 01 Jul 2024 Prob (F-statistic): 4.77e-106

Time: 17:04:06 Log-Likelihood: -2425.6

No. Observations: 6899 AIC: 4869.

Df Residuals: 6890 BIC: 4931.

Df Model: 8

Covariance Type: nonrobust

==================================================================================

coef std err t P>|t| [0.025 0.975]

----------------------------------------------------------------------------------

const 0.5080 0.044 11.657 0.000 0.423 0.593

hhdsz -0.0129 0.002 -5.199 0.000 -0.018 -0.008

Religion 0.0073 0.009 0.805 0.421 -0.010 0.025

MPCE\_URP 4.251e-05 2.19e-06 19.432 0.000 3.82e-05 4.68e-05

Sex -0.1156 0.016 -7.075 0.000 -0.148 -0.084

Age -0.0011 0.000 -3.214 0.001 -0.002 -0.000

Marital\_Status 0.0933 0.013 7.053 0.000 0.067 0.119

Education -0.0067 0.001 -5.694 0.000 -0.009 -0.004

price -0.0020 0.000 -7.246 0.000 -0.003 -0.001

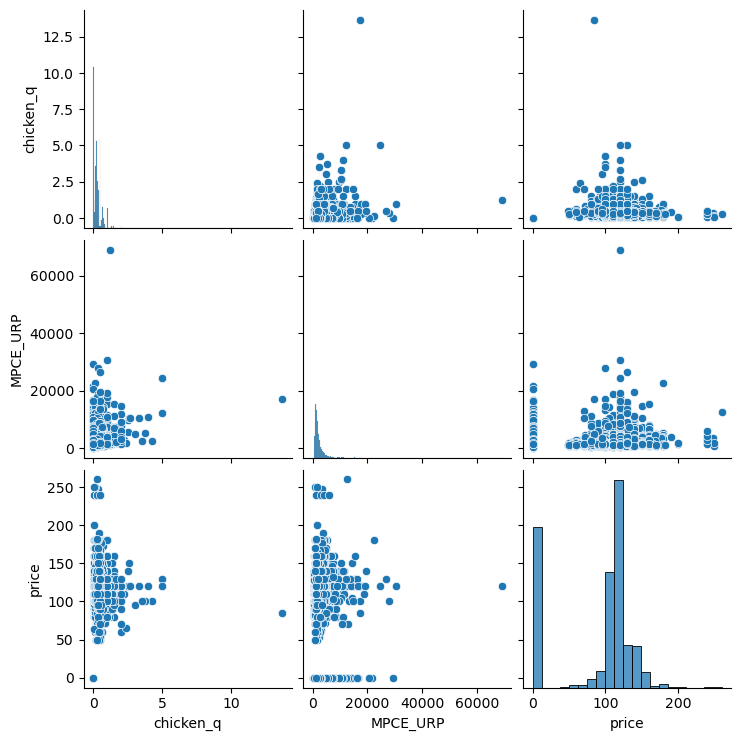
==============================================================================

Omnibus: 10654.089 Durbin-Watson: 1.836

Prob(Omnibus): 0.000 Jarque-Bera (JB): 22453451.708

Skew: 9.114 Prob(JB): 0.00

Kurtosis: 281.887 Cond. No. 3.18e+04



**Interpretation:**

* Correct classifications are on the diagonal, with projected columns matching actual rows. For example, the value 1084 in the upper left corner indicates that the model accurately predicted 1084 instances of class 0.
* Incorrect classifications: Off-diagonal. For example, the value 4583 in row 0, column 1 represents the number of times the model predicted class 1 while the instance was actually class 0 (false positive).
* The table indicates a class imbalance, with more class 0 instances (4687) than class 1 instances (1313). This makes evaluating model performance challenging, particularly for the minority class (class 1 in this case).

Without knowing what the specific classes represent, it's difficult to evaluate precisely how well the model works. However, several broad observations can be made:

The model performs well in class 0, with a high number of true predictions (1084) and low false positives (4583).

Class 1 performance can be difficult to interpret due to class imbalance. The model correctly predicted 229 instances of class 1, but it also generated 104 false negatives. This shows that the model may be missing a large number of true class 1 cases.

• Correct categories appear on the diagonal (green). The model did well in class 0 (top-left corner, 1084 correct), but not in class 1 (bottom-right, 229 correct).   
• Incorrect classifications appear off-diagonal (red). For example, 4583 (top row, right column) indicates that the model mistakenly predicted class 1 for a large number of class 0 cases.  
• Calculate Precision and Recall (not shown) to assess the accuracy of positive predictions (precision) and the model's ability to identify all positive cases.

Overall, the model seems to perform well on the majority class (class 0) with high correct classifications (1084) and lower misclassifications. The model struggles more with the minority class (class 1).