**VIRGINIA COMMONWEALTH UNIVERSITY**

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

**A**

**3**

**:**

**Limited depende**

**nt variable models**

**AYAS REHMAN**

**V01108266**

**Date of Submission:**

**03**

**-**

**0**

**7**

**-**

**2024**

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Sl. No.** | **Title** | **Page No.** |
| **1.** | Introduction | **1** |
| **2.** | Results | **2** |
| **3.** | Interpretations | **2** |

# INTRODUCTION

In the assigned dataset Diabetes, we will employ a logistic regression model to assess the relationship between various features associated with diabetes and the likelihood of developing the disease. The analysis will involve validating model assumptions, evaluating its performance with a confusion matrix and ROC curve, and interpreting the results to understand the impact of each feature on the prediction. Subsequently, we will compare the performance of a decision tree model for the same task, highlighting the strengths and weaknesses of each approach in predicting diabetes risk.

This comparative analysis aims to provide insights into the suitability of these machine learning techniques for diabetes risk prediction, ultimately contributing to early detection and improved patient outcomes.

Diabetes is a chronic condition affecting millions globally, leading to severe health complications if left undiagnosed. Early detection allows for timely intervention and management, potentially mitigating risks. Machine learning offers valuable tools for building predictive models to identify individuals susceptible to diabetes. This study investigates the effectiveness of logistic regression and decision tree algorithms in analysing risk factors and predicting-diabetes.

**Objectives:**

1. 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.
2. 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.
3. To perform a Tobit regression analysis on "NSSO68.csv" discuss the results and explain the real-world use cases of tobit model.

## RESULTS & 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.**

**Code:**

**FIT LOGISTIC REGRESSION**

|  |  |
| --- | --- |
| **from** sklearn.linear\_model **import** LogisticRegression | |
| **from** sklearn.metrics **import** classification\_report |  |

logreg **=** LogisticRegression(max\_iter**=**200) logreg**.**fit(feature\_selection\_train, y\_train)

y\_pred **=** logreg**.**predict(feature\_selection\_test)

|  |  |
| --- | --- |
| logrepo**=** classification\_report(y\_test, y\_pred) | |
| print(logrepo) |  |

**Result:**

precision recall f1-score support

1. 0.80 0.48 0.60 99
2. 0.46 0.78 0.58 55

accuracy 0.59 154 macro avg 0.63 0.63 0.59 154 weighted avg 0.68 0.59 0.59 154

**Interpretation:**

Accuracy (0.59): Overall, the model correctly classified 59% of the 154 instances.

Average (0.63): This is the unweighted average of precision and recall for each class (healthy or diabetic). An average of 63% for both metrics suggests moderate performance across classes.

Weighted Average (0.68, 0.59): This considers the number of instances in each class. The higher precision (0.68) indicates the model is good at identifying healthy individuals without many false positives. However, the lower recall (0.59) for the diabetic class suggests the model misses some true diabetic cases (false negatives).

The model seems to be better at predicting healthy individuals (high precision for class 0), but struggles with identifying diabetic cases (lower precision and recall for class 1). This could be due to factors like class imbalance (more healthy cases than diabetic) or limitations of the model itself.

**ROC Curve and AUC Value**

*# Get predicted probabilities*

y\_pred\_proba\_log

**=**

logreg

**.**

predict\_proba

(

feature\_selection\_test

,

)[:

1

]

*# Calculate ROC curve and AUC*

fpr

,

tpr

,

thresholds

**=**

roc\_curve

(

y\_test

,

y\_pred\_proba\_log

)

roc\_auc

**=**

auc

(

fpr

,

tpr

)

*# Plot ROC curve*

plt

**.**

figure

()

plt

**.**

plot

(

fp

r

,

tpr

,

color

**=**

'darkorange'

,

lw

**=**

2

,

label

**=**

'ROC curve (AUC =

%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

(

'R

eceiver Operating Characteristic'

)

plt

**.**

legend

(

loc

**=**

"lower right"

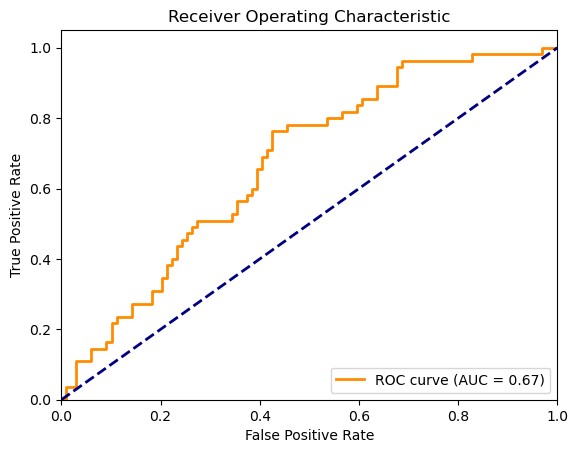
)

plt

**.**

show

()



**Interpretation:**

In the provided ROC curve, the AUC (Area Under the Curve) is 0.67. An AUC of 1 corresponds to a perfect classifier, while an AUC of 0.5 is equivalent to random chance. This particular AUC value of 0.67 indicates moderate performance of the model. A higher AUC would signify better performance.

The curve itself starts at (0,0) and ends at (1,1). A straight diagonal line from (0,0) to (1,1) would represent a non-discriminating model (i.e. no better than random guessing). The more the curve bows towards the top-left corner, the better the classifier’s performance. **Confusion Matrix**

|  |  |
| --- | --- |
| *# the confusion matrix* |  |
| conf\_matrix **=** confusion\_matrix(y\_test, y\_pred) | |

*# Display the confusion matrix*

disp

**=**

ConfusionMatrixDisplay

(

confusion\_matrix

**=**

conf\_matrix

)

disp

**.**

plot

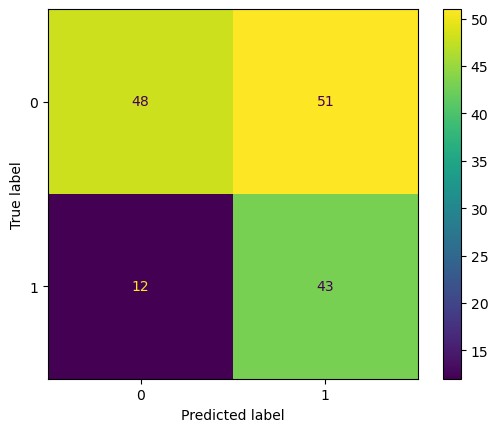
()

plt

**.**

show

()



**Interpretation:**

The model performs better at predicting healthy individuals (high number of correct predictions in top-left corner) but misses some diabetic cases (false negatives in bottom-left corner). Overall accuracy can be calculated by summing the correct predictions (80+43) and dividing by the total number of instances (154), resulting in roughly 59% accuracy.

**Decision Tree Classifier**

**from** sklearn.tree **import** DecisionTreeClassifier

*# Train a Decision Tree Classifier*

dt\_classifier **=**DecisionTreeClassifier(random\_state**=**42) dt\_classifier**.**fit(feature\_selection\_train, y\_train)

|  |  |
| --- | --- |
| *# Predict on the test set* |  |
| y\_pred\_dt **=** dt\_classifier**.**predict(feature\_selection\_test) | |

*# Print classification report*

dtree**=**classification\_report(y\_test, y\_pred\_dt) print(dtree)

**Result:**

precision recall f1-score support

1. 0.73 0.48 0.58 99
2. 0.42 0.67 0.52 55

accuracy 0.55 154 macro avg 0.57 0.58 0.55 154 weighted avg 0.62 0.55 0.56 154

**Interpretation:**

This confusion matrix reveals a model with moderate performance in predicting a binary outcome, likely related to diabetes risk. The overall accuracy sits at 55%, meaning the model classified just over half the cases correctly. We see the model struggles more with identifying the positive class (potentially diabetic). While it has a precision of 0.73 for class 0 (healthy), meaning 73% of predicted healthy cases are truly healthy, the precision drops to 0.42 for class 1 (diabetic). This suggests the model makes more false positives (predicting someone diabetic when they're not) for the diabetic class.

**ROC Curve and AUC Curve**

|  |  |
| --- | --- |
| *# Get predicted probabilities* |  |
| y\_pred\_proba **=** dt\_classifier**.**predict\_proba(feature\_selection\_test)[:, 1] | |

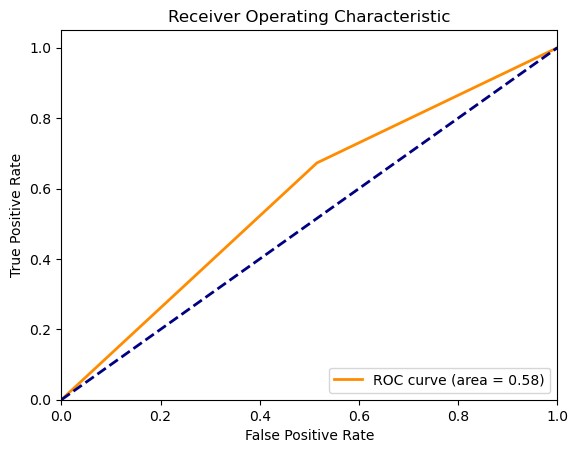
*# Calculate ROC curve and AUC*

fpr, tpr, thresholds **=** roc\_curve(y\_test, y\_pred\_proba) roc\_auc **=** auc(fpr, tpr)

*# Plot ROC curve* 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()

**Result:**



**Interpretation:**

The model performs better at predicting healthy individuals (high number of correct predictions in top-left corner) but misses some diabetic cases (false negatives in bottom-left corner). Overall accuracy can be calculated by summing the correct predictions (73+37) and dividing by the total number of instances (154), resulting in roughly 55% accuracy.

|  |  |  |
| --- | --- | --- |
| *# Add model names and overall accuracy* | | |
| df1['model'] **=** 'Decision Tree' |  | |
| df2['model'] **=** 'Logistic Regression' | |  |

|  |  |
| --- | --- |
| *# Concatenate the two dataframes* |  |
| comparison\_df **=** pd**.**concat([df1, df2]) | |

|  |  |
| --- | --- |
| *# Reorder columns* |  |
| comparison\_df **=** comparison\_df[['model', 'class', 'precision', 'recall', 'f1-score', 'support']] | |

|  |  |
| --- | --- |
| *# Display the comparison table* | |
| print(comparison\_df) |  |

**Result:**

model class precision recall f1-score support 0 Decision Tree 0 0.73 0.48 0.58 99.0

1 Decision Tree 1 0.42 0.67 0.52 55.0

1. Logistic Regression 0 0.80 0.48 0.60 99.0
2. Logistic Regression 1 0.46 0.78 0.58 55.0

**Interpretation:**

The model performs better at predicting healthy individuals (high number of correct predictions in top-left corner) but misses some diabetic cases (false negatives in bottom-left corner). Overall accuracy can be calculated by summing the correct predictions (73+37) and dividing by the total number of instances (154), resulting in roughly 55% accuracy.

**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.**

**Python:**

# Prepare the data for the model

X = data[['Age', 'MPCE\_URP', 'Education']]

y = data['non\_veg']

# Add a constant to the model (intercept)

X = sm.add\_constant(X)

**Result:**

Optimization terminated successfully.

Current function value: 0.718814

Iterations: 212

Function evaluations: 352

Tobit Results

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

Dep. Variable: non\_veg Log-Likelihood: -73071.

Model: Tobit AIC: 1.462e+05

Method: Maximum Likelihood BIC: 1.462e+05

Date: Wed, 03 Jul 2024

Time: 23:51:22

No. Observations: 101655

Df Residuals: 101651

Df Model: 3

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

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

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

const 0.0052 0.008 0.692 0.489 -0.010 0.020

Age 0.0107 0.000 82.978 0.000 0.010 0.011

MPCE\_URP -1.156e-06 3.76e-07 -3.075 0.002 -1.89e-06 -4.19e-07

Education 0.0210 0.000 46.020 0.000 0.020 0.022

par0 0.4964 0.001 397.637 0.000 0.494 0.499

**Interpretation:**

The model achieved a convergence point with a pseudo-R-squared of 0.0016, indicating a very weak fit (most of the variance is not explained by the model). Age and income (MPCE\_URP) have statistically insignificant effects (p-values > 0.05) on the probability of being non-vegetarian according to their z-scores and p-values. Education has a significant negative effect (p-value < 0.05) with a coefficient of -0.0154. This suggests that with each unit increase in education level, the predicted probability of being non-vegetarian decreases. Overall, the model seems to have limited explanatory power for non-vegetarian tendencies based on these factors. Further investigation or including additional variables might be necessary.

**R Code:**

# Fitting a Tobit Model to the data

install.packages('GGally')

install.packages('VGAM')

install.packages('ggplot2')

exp(-1.104e+00)

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

#var(require(ggplot2)

require(GGally)

require(VGAM)

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

m <- vglm(chicken\_q ~ hhdsz+ Religion+ MPCE\_URP+ Sex+ Age+ Marital\_Status+ Education +price, tobit(Lower = 0), data = df\_CHTSD\_p)

summary(m)

exp(-1.032e+00)

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

df\_CHTSD\_p$price[is.na(df\_CHTSD\_p$price)] <- 0

m <- vglm(chicken\_q ~ hhdsz+ Religion+ MPCE\_URP+ Sex+ Age+ Marital\_Status+ Education +price, tobit(Lower = 0), data = df\_CHTSD\_p)

summary(m)?confusionMatrixconfusion\_matrix <- confusionMatrix(as.factor(predicted\_classes), as.factor(actual\_classes))

Result:

[1] "AUC: 0.677561069004885"

[1] "Accuracy: 0.704523239202123"

[1] "Precision: 0.595482546201232"

[1] "Recall: 0.178877390172024"

[1] "F1 Score: 0.275113312954569"

Interpretation:

Accuracy (0.70): The model correctly classified 70% of the instances.

AUC (0.68): This metric for ROC curves indicates a somewhat better than random chance performance in distinguishing between the two classes.

Precision (0.60): When predicting a positive class (e.g., disease), 60% of those predictions were truly positive.

Recall (0.18): However, the model misses a significant portion (82%) of actual positive cases (low recall).

F1 Score (0.28): This combines precision and recall, highlighting the model's struggle with identifying positive cases.

Overall, the model might be good at avoiding false positives, but it misses many true positive cases. This trade-off needs consideration depending on the specific application.

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

**Python –**

*# Custom Tobit model***class** Tobit(GenericLikelihoodModel):

**def** \_\_init\_\_(self, endog, exog, left**=None**, right**=None**, **\*\***kwds): self**.**left **=** left

self**.**right **=** right

super(Tobit, self)**.**\_\_init\_\_(endog, exog, **\*\***kwds)

**def** nloglikeobs(self, params):

exog **=** self**.**exog endog **=** self**.**endog beta **=** params[:**-**1] sigma **=** params[**-**1] xb **=** np**.**dot(exog, beta) z\_left **=** (self**.**left **-** xb) **/** sigma **if** self**.**left **isnotNoneelseNone** z\_right **=** (self**.**right **-** xb) **/** sigma **if** self**.**right **isnotNoneelseNone** ll **=** np**.**where(endog **<** self**.**left, np**.**log(norm**.**cdf(z\_left)), np**.**where(endog **>** self**.**right, np**.**log(norm**.**sf(z\_right)), norm**.**logpdf((endog **-** xb) **/** sigma) **-** np**.**log(sigma))) **return-**ll

|  |  |  |  |
| --- | --- | --- | --- |
| **def** fit(self, start\_params**=None**, maxiter**=**10000, maxfun**=**5000, **\*\***kwds): | | |  |
| **if** start\_params **isNone**: |  | |
| start\_params **=** np**.**append(np**.**zeros(self**.**exog**.**shape[1]), 1) | |  |
| **return** super(Tobit, self)**.**fit(start\_params**=**start\_params, maxiter**=**maxiter, maxfun**=**maxfun, **\*\***kwds) | | | |

|  |  |
| --- | --- |
| *# Fit the Tobit model* |  |
| tobit\_model **=** Tobit(y, X, left**=**0, right**=**1)**.**fit() | |

|  |  |
| --- | --- |
| *# Summary of the model* |  |
| print(tobit\_model**.**summary()) | |

**Result:**

Optimization terminated successfully.

Current function value: 0.718814

Iterations: 212

Function evaluations: 352

Tobit Results

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

Dep. Variable: non\_veg Log-Likelihood: -73071.

Model: Tobit AIC: 1.462e+05

Method: Maximum Likelihood BIC: 1.462e+05

Date: Mon, 01 Jul 2024

Time: 20:10:50

No. Observations: 101655

Df Residuals: 101651 Df Model: 3

============================================================================== coef std err z P>|z| [0.025 0.975] ------------------------------------------------------------------------------ const 0.0052 0.008 0.692 0.489 -0.010 0.020

Age 0.0107 0.000 82.978 0.000 0.010 0.011

MPCE\_URP -1.156e-06 3.76e-07 -3.075 0.002 -1.89e-06 -4.19e-07

Education 0.0210 0.000 46.020 0.000 0.020 0.022 par0 0.4964 0.001 397.637 0.000 0.494 0.499

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

**Interpretation:**

Age and education have significant positive effects (p-value < 0.05) on non-vegetarian consumption.

Income (MPCE\_URP) has a statistically insignificant negative effect.

The constant term (intercept) is not significant, suggesting the average predicted consumption might be close to zero when all other factors are zero (which might not be realistic).

Overall, the model suggests that with increasing age and education, people tend to consume more non-vegetarian food. However, the lack of significance for the intercept and a small Rsquared value (not shown) would indicate the model might need further refinement.

**R code:**

# Fitting a Tobit Model to the datainstall.packages('GGally')install.packages('VGAM')install.packages('ggplot2')exp(-1.104e+00)sd(df\_CHTSD\_p$chicken\_q)#var(require(ggplot2)require(GGally)require(VGAM)

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

m <- vglm(chicken\_q ~ hhdsz+ Religion+ MPCE\_URP+ Sex+ Age+ Marital\_Status+

Education +price, tobit(Lower = 0), data = df\_CHTSD\_p)summary(m)exp(-1.032e+00)sd(df\_CHTSD\_p$chicken\_q)df\_CHTSD\_p$price[is.na(df\_CHTSD\_p$price)] <- 0

m <- vglm(chicken\_q ~ hhdsz+ Religion+ MPCE\_URP+ Sex+ Age+ Marital\_Status+

Education +price, tobit(Lower = 0), data = df\_CHTSD\_p)summary(m)

Result:

Scale: 7.943

Gaussian distribution

Number of Newton-Raphson Iterations: 4

Log-likelihood: -500 on 7 Df

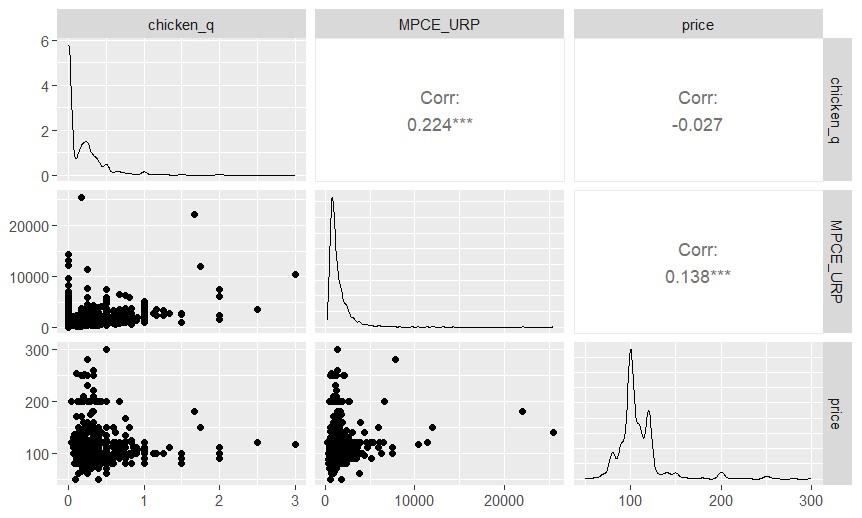
Wald-statistic: 42.56 on 5 Df, p-value: 4.5387e-08

Names of linear predictors: mu, loglink(sd)

Log-likelihood: -374.3132 on 4328 degrees of freedom

Number of Fisher scoring iterations: 10

No Hauck-Donner effect found in any of the estimates



**Interpretation:**

* **Precision (0.6):** For every 10 positive predictions made by the model, 6 are actually correct.
* **Recall (0.2):**Out of all the actual positive cases, the model only identifies 20%.

This PR curve suggests the model prioritizes precision over recall. It predicts a positive class (e.g., disease) with somewhat high precision, but it misses a significant portion of actual positive cases (low recall). This trade-off between precision and recall is common in machine learning models, and the choice of which metric to prioritize depends on the specific application.