Logistic Regression and Decision Tree Predictive Modeling: A Fit Function Comparison

Randall E. Schumacker
University of Alabama

Todd Sherron
Texas State University

Decision Tree is a predictive modeling technique that is used in classification problems. There are two main types of Decision Tree predictive models: classification or regression. The classification model has a categorical outcome variable. The regression model has a continuous outcome variable. We focused on the classification model approach in our comparison because it parallels the logistic regression binary outcome. The Decision Tree and Logistic Regression predictive methods did not have the same classification accuracy because of the "cut value" used for the first node in the Decision Tree. They are both classification methods which differ in their selection of the first "*most*" important variable in the classification procedure. Decision Tree predictive modeling does not permit the selection of the "*first*" node because it is based on the highest variable Information Gain while Logistic Regression has the advantage of determining and selecting variable entry from predictor significance. Logistic Regression slightly outperformed the Decision Tree classification method.

Logistic regression is a popular predictive modeling technique that tests the relation of several independent variables to an outcome variable. The outcome variable is typically coded as a dichotomous outcome with 0 representing no disease and 1 indicating diseased in epidemiologic data (Kleinbaum, 1994; Agresti, 1996). Decision Tree is also a predictive modeling technique that is used in regression classification problems (R-Blogger, 2021). There are two main types of Decision Tree predictive models: classification or regression. The classification model has a categorical outcome variable. The regression model has a continuous outcome variable. We therefore focused on the classification model approach that parallels the logistic regression binary outcome.

The *"fit"* function for the logistic regression equation that estimates the unknown parameters for the independent variables is AIC, Deviance, Confusion Matrix, and ROC. The mean squared error is sometimes reported as the average squared difference between the predicted result and the actual result. Logistic regression permits assessing the classification accuracy of the modeled parameters for the predicted and actual result in a *confusion matrix*. For our purpose, the "*fit*" function will be the classification accuracy percent in the confusion matrix.

The "*fit"* function for the Decision Tree predictive modeling is called, Entropy, or the measure of uncertainty or randomness in the data related to the predictability of a certain binary outcome. Basically, lower values indicate less uncertainty, and higher values imply more uncertainty. An information gain is computed using Entropy values to indicate the variable importance in binary outcome classification. Information gain values therefore are used to decide which independent variables enhance the prediction of the outcome. The overall *"fit"* function is the classification accuracy percent in what is called a *confusion matrix*.

Variable selection is an important step in predictive modeling techniques. If either method is to be effective, the selection of the first independent variable is critical. Researchers by default generally use the results of the analysis to select variables, for example in regression using all possible subset and the statistical significance of the independent variables. In Decision Tree applications, the lowest Entropy and thus the highest information gain of a variable is computed which automatically selects the most important first independent variable to begin the decision tree structure, basically the first node in the tree.

If an effective set of independent variables are *not* chosen for Logistic Regression or Decision Tree predictive modeling, then the fit function (classification accuracy) would not yield better accurate results. A comparison of the two predictive modeling techniques using the same set of independent variables with a binary outcome fit function (classification accuracy) is presented.

Background

It is important to provide an understanding of the Logistic Regression and Decision Tree predictive modeling application and interpretation, especially their classification fit function. Logistic Regression is a mathematical model with a logistic function expressed as:

$$f(z) = \frac{1}{1 + e^{-z}}$$

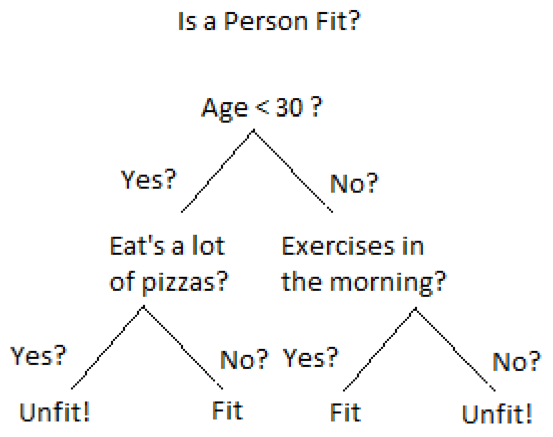
The range of f(z) is between 0 and 1 regardless of the value of z. The logistic model is designed to describe the probability or risk of an individual getting a disease, or other type of binary outcome (Klienbaum, 1994; Agresti, 1996). The logistic model probability will always be between 0 and 1, which is not true for other multiple regression type models. The monotonic S-shaped curve displays an interpretable result that combines several independent variables (risk factors) for f(z), the risk for a given z value. The logistic regression model with k independent variables is expressed as:

$$z = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k$$

The dependent variable, z, is an index that combines the X's. The X variables are typically selected based on statistical significance. The probability of developing a disease, P(X), given the significant parameters in the logistic regression equation would then be expressed as:

$$P(X) = \frac{1}{1 + e^{-(\beta_0 + \Sigma \beta_k X_k)}}$$

The Decision Tree classification approach has a tree-like structure with typical yes-no decisions that represent decision rules leading to an eventual categorical outcome. Decision trees are popular because of their display simplicity and interpretability. A simple binary Decision Tree (Source: https://www.xoriant.com) can be displayed as:



The decision rules were the questions: What is the age?, Does the person exercise ?, Does the person eat a lot of pizza? The eventual outcome is either fit or unfit. This is a binary Decision Tree classification problem (yes-no). Probabilities are given at each node.

A "*fit*" function algorithm in Decision Tree predictive modeling is called, Entropy, or the measure of uncertainty or randomness in the data related to the predictability of a certain binary outcome. Entropy, S, is expressed as:

$$Entropy(S) = \sum \mathrm{p(x)} log_2 \frac{1}{p(x)}$$

Entropy is 0 if all members belong to the same group, and 1 if half of them belong to one group and the other half belong to the other group (50/50 split) which would indicate perfect randomness. An information gain [IG (S, A)] algorithm is computed that indicates the change in Entropy, S, for a given set of independent variables on a particular outcome, A. It is expressed as:

$$IG(S, A) = Entropy(S) - \sum \mathrm{P(x)} Entropy(S)$$

The value S is the Entropy for the set of independent variables and the second term is Entropy after applying the particular outcome, A, where P(x) is the probability of the outcome event. Information gain values are used to decide which independent variables enhance the prediction of the outcome. A useful "*fit*" function is the classification accuracy percent in what is called a *confusion matrix*. The Decision Tree approach typically employs a split sample, one for training and the other for testing. The selection of the first important variable for the initial node in a Decision Tree is based on the highest information gain.

Logistic regression and Decision Tree predictive methods have been shown to be comparable in data mining applications (Raju & Schumacker, 2016). Logistic regression and Decision Tree today are considered supervised machine learning techniques that use either a loss function or an optimizing function to improve the model results. The loss function measures the averaged squared difference between predicted and actual results. The optimizing function selects independent variables with significant parameters to improve prediction results. In both scenarios, a set of independent variables are selected and then a determination is made as to the efficacy of the independent variables to adequately predict the binary outcome. An easy way to compare both methods is via the classification accuracy table.

Methodology

The R version 4.4.1 software (https://cran.r-project.org/) was used to run the Logistic Regression and Decision Tree comparison. The R program script files for both Logistic Regression and Decision Tree predictive models are in the Appendix. The regression equation parameters were estimated using the *glm* function with predicted values using the *predict* function. The Decision Tree was created using the *rpart* function with predicted values using the *predict* function. Confusion matrices (classification accuracy tables) were created for both predictive models. The classification accuracy was computed based on the sum of the diagonal values divided by the total.

Data Source

The **iris** data set in the base R version 4.4.1 software was used for comparative analyses. The data set included 150 flowers with 3 species of which only 2 species were used in our analysis. The reduced data set, **iris_small**, had 50 *versicolor* and 50 *virginica* species type flowers. The independent variables were *sepal.width*, *sepal.length*, *petal.width*, and *pedal.length.*

Results

The Logistic Regression output and the Decision Tree output are presented separately. A comparison of their classification accuracy is presented and discussed.

Logistic Regression

The logistic regression predictive modeling output using the Deviance criteria for the I**ris** small data set, training data set, and the test data set are in Table 1.

Table 1. Logistic Regression Deviance Criteria

```
Iris Small Data Set (Only 2 Flowers)
            Df Deviance Resid. Df Resid. Dev              Pr(>Chi)
NULL                           99    138.629
Sepal.Width   1   10.049       98    128.581             0.0015246 **
Sepal.Length  1   18.255       97    110.326            0.00001932 ***
Petal.Width   1   84.424       96     25.902 < 0.00000000000000022 ***
Petal.Length  1   14.003       95     11.899             0.0001825 ***

Iris Train Data Set (70%)
             Df Deviance Resid. Df Resid. Dev              Pr(>Chi)
NULL                            69     96.983
Sepal.Width   1    2.151        68     94.832              0.142434
Sepal.Length  1    9.386        67     85.446              0.002186 **
Petal.Width   1   67.781        66     17.664 < 0.00000000000000022 ***
Petal.Length  1    7.134        65     10.531              0.007564 **

Iris Test Data set (30%)
             Df Deviance Resid. Df Resid. Dev   Pr(>Chi)
NULL                            29     41.455
Sepal.Width   1   12.915        28     28.540  0.0003260 ***
Sepal.Length  1   16.226        27     12.314  0.00005622 ***
Petal.Width   1   12.314        26      0.000  0.0004495 ***
Petal.Length  1    0.000        25      0.000  0.9999746
```

The resulting classification table using the test data was:

```
               Predicted       Predicted
               Versicolor      Virginica        Sum
  versicolor    14                 2             16
  virginica      0                14             14
  Sum           14                16             30
```

The classification accuracy was 93 % (14 + 14 = 28 / 30 = 0.93).

Decision Tree

The Decision Tree predictive modeling output using the Entropy and Information Gain criteria for the **Iris** small data set, training data set, and the test data set are in Table 2.

Table 2. Decision Tree Information Gain Criteria

Iris Small Data Set (Only 2 flowers)

```
    attributes importance
1 Sepal.Length  0.1112501
2  Sepal.Width  0.0000000
3 Petal.Length  0.4556568
4  Petal.Width  0.4783827
```

Iris Train Data Set (70%)

```
   attributes importance
1 Sepal.Length  0.0000000
2  Sepal.Width  0.0000000
3 Petal.Length  0.4792361
4  Petal.Width  0.4476642
```

Iris Test Data Set (30%)

```
    attributes importance
1 Sepal.Length  0.3730996
2  Sepal.Width  0.2836804
3 Petal.Length  0.6909233
4  Petal.Width  0.5641497
```

The *rpart* function automatically selected *Petal.Length* based on the Information Gain result in the test data set. It also automatically choose the split criteria for the binary classification (yes-no). The Decision Tree displayed the actual (0.49; 0.51) and predicted percent (0.46; 0.54) based on the *Petal.Length* split. These percent values are related to the predictive model accuracy. There were 16 *versicolor* flowers and 14 *virginica* flowers in the test data set. The resulting Decision Tree is displayed in Figure 1.

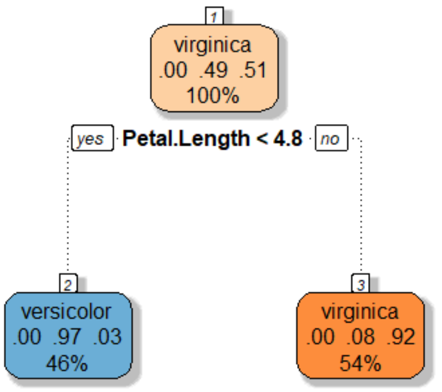


Figure 1. Decision Tree for Iris Test Data

The test set predictions showed 3 misclassifications:

```
                 Predicted  Predicted
                 versicolor virginica
  versicolor          13         0
  virginica            3        14
```

The results indicated that 3 flowers were originally *versicolor* but misclassified as *virginica*. The sum of the diagonal values divided by the total yields the classification accuracy percent of 90% (13 + 14 = 27 divided by 30 = 0.90).

Summary and Conclusion

The Logistic Regression analysis picked *Petal.Width* (small data set), *Petal.Width* (train data set), *and Sepal.Length* (test data set). Decision Tree picked *Petal.Width* (small data set), *Petal.Length* (train data set), *and Petal.Length* (test data set). The Logistic Regression and Decision Tree agreed on their selection of which variable was the most important in the full **Iris** data set, but not the subsequent training and test data sets. Logistic regression indicated only 2 misclassifications, hence classification accuracy = 93%. Decision Tree indicated a total of 3 misclassifications, hence classification accuracy = 90%. The two predictive methods didn't have the same classification accuracy because of the "cut value" used for the first node in the Decision Tree. Another explanation for the variable selection difference in the subsequent data sets is the **Simpson's paradox** which is a phenomenon in probability and statistics where a trend appears in subsets of data but disappears or reverses when the groups are combined (Wikipedia, 2024).

It is important to understand the variable selection in Logistic Regression and Decision Tree predictive modeling. They are both classification methods which can differ in their selection of the first *"most"* important variable given subsets of data. Decision Tree predictive modeling does not permit the selection of the *"first"* node because it is based on the highest variable Information Gain.

Discussion

How can you evaluate Logistic Regression model fit and accuracy ?

The generally recognized criteria are:

1. Akaike Information Criteria (AIC). AIC is counterpart to Adjusted R Squared.
2. Deviance
3. Receiver Operator Characteristic (ROC)
4. Confusion Matrix

How can you evaluate Decision Tree model fit and accuracy?

In Decision Tree algorithms, the initial root node is selected by default as the variable with the highest Information Gain. Afterwards, on each iteration of the algorithm, it selects any unused variable and calculates Entropy and Information Gain. It selects the next variable based on the highest Information Gain. The generally recognized criteria are:

1. Entropy
2. Information Gain
3. Confusion Matrix

We chose the Confusion Matrix, also known as the Classification Accuracy table. Ultimately, this is what classification methods are all about – **How well can we predict group classification (membership).**

References

Agresti, A. (1996). *An Introduction to Categorical Data Analysis*. NY: John Wiley & Sons.

Kleinbaum, D.G. (1994). *Logistic Regression: A Self-Learning Text*. NY: Springer-Verlag.

Raju, D. &, Schumacker, R. (2016). Comparing Data Mining Models in Academic Analytics. *International Journal of Knowledge-Based Organizations, 6(2), 1-15.*

R-Blogger (2021). Machine Learning with R: A Complete Guide to Decision Trees. Retrieved from: https://www.r-bloggers.com/2021/02/machine-learning-with-r-a-complete-guide-to-decision-trees/

Wikipedia (2024). Simpson's paradox. Retrieved from:
https://en.wikipedia.org/wiki/Simpson%27s_paradox

Send correspondence to: Randall E. Schumacker
University of Alabama
Email: rschumacker@ua.edu

Appendix

Logistic Regression R Script

```
# Logistic Regression R Version 4.4.1
# Iris data set in R n = 150 iris flowers with 3 different species
# Drop setosa type species, leaving 100 flowers

data(iris)
options(scipen=999) # remove scientific notation

# make a reduced iris data set that only contains virginica and versicolor
species
iris_small = iris[51:150,]
iris_small

# Random split:  70% Train (70) and 30% Test (30)
set.seed(20)
trainlr = sample(1:nrow(iris_small), nrow(iris_small)*0.7)
trainlr

# Train data set
train_set = iris_small[trainlr,]
train_set

# Test data set
test_set = iris_small[-trainlr, ]
test_set

# Logisitic Regression on training data
modellr = glm(Species ~ Sepal.Width + Sepal.Length + Petal.Width +
Petal.Length, data = train_set,
    +  family = binomial)
summary(modellr)
anova(modellr)

# Use Test data to evaluate the logistic regression equation from Train data
predlr = predict(modellr, newdata = test_set ,type = "response")
predlr

# Confusion Matrix = Classification Accuracy
predlr.class = ifelse(predlr >= .5,1,0) # .5 is the cut-off value selected by
Researcher
table(predlr.class)
new = droplevels(test_set$Species)
new

conf.table = table(new,predlr.class)
colnames(conf.table) = c("Predicted Versicolor", "Predicted Virginica")
addmargins(conf.table)
conf.table

# Model Classification Accuracy
accuracy = (conf.table[1,1] + conf.table [2,2]) /30
cat("Prediction Accuracy: ",accuracy)
```

Decision Tree R Script

```
# Decision Tree
# Package rpart and function rpart

install.packages("rpart"); library(rpart)
install.packages("rpart.plot"); library(rpart.plot)
install.packages("caret"); library(caret)

# make a reduced iris data set that only contains virginica and versicolor
species
data(iris)
iris_small = iris[51:150,]
iris_small

# Random split 70% Train (70) and 30% Test (30)
set.seed(20)
trainlr = sample(1:nrow(iris_small),nrow(iris_small)*0.7)
trainlr

# Train data set
train_set = iris_small[trainlr,]
train_set

# Test data set
test_set = iris_small[-trainlr, ]
test_set

iris_tree = rpart(Species ~ Sepal.Width + Sepal.Length + Petal.Width +
Petal.Length,data = train_set,
    +  method="class")
iris_tree

# Plot iris decision tree
install.packages("rattle")
library(rattle)

fancyRpartPlot(iris_tree, main = "Decision Tree for Iris Dataset",
caption=NULL)

# Confusion Matrix for predictions
predictions = predict(iris_tree, test_set, type = "class")
predictions

testpred = table(predictions, test_set$Species, exclude = "setosa")
testpred

# Model Classification Accuracy
accuracy = ((testpred[1,1] + testpred [2,2])/30)
cat("Decision Tree Prediction Accuracy: ",accuracy)

cm <- confusionMatrix(predictions, test_set$Species)
cm
```

Entropy

```
install.packages("entropy")
library(entropy)
install.packages("FSelectorRcpp")
library(FSelectorRcpp)

# make a reduced iris data set that only contains virginica and versicolor
species
data(iris)
iris_small = iris[51:150,]
irisXsmall = iris[51:150,]
irisXsmall = iris_small[-5]

SL = entropy(irisXsmall$Sepal.Length,method="ML")
cat(SL, "Sepal.Length")
SW = entropy(irisXsmall$Sepal.Width,method="ML")
cat(SW, "Sepal.Width")
PL = entropy(irisXsmall$Petal.Length,method="ML")
cat(PL,"Petal.Length")
PW = entropy(irisXsmall$Petal.Width,method="ML")
cat(PW, "Petal.Width")

# Run Entropy on Train data set
# Train data set
set.seed(20)
trainlr = sample(1:nrow(iris_small),nrow(iris_small)*0.7)
trainlr

train_set = iris_small[trainlr,]
trainX = train_set[-5]

SLT = entropy(trainX$Sepal.Length,method="ML")
cat(SLT, "Sepal.Length")
SWT = entropy(trainX$Sepal.Width,method="ML")
cat(SWT, "Sepal.Width")
PLT = entropy(trainX$Petal.Length,method="ML")
cat(PLT,"Petal.Length")
PWT = entropy(trainX$Petal.Width,method="ML")
cat(PWT, "Petal.Width")

# Run Information Gain on Test data set
# Test data set
test_set = iris_small[-trainlr,]
testX = test_set[-5]


TSLT = entropy(testX$Sepal.Length,method="ML")
cat(TSLT, "Sepal.Length")
TSWT = entropy(testX$Sepal.Width,method="ML")
cat(TSWT, "Sepal.Width")
TPLT = entropy(testX$Petal.Length,method="ML")
cat(TPLT,"Petal.Length")
TPWT = entropy(testX$Petal.Width,method="ML")
cat(TPWT, "Petal.Width")
```

Information Gain

```
# Calculate Information Gain

install.packages("FSelectorRcpp")
library(FSelectorRcpp)

data(iris)

# Calculate Information Gain
irisX <- iris[-5]
y <- iris$Species

# data.frame interface
information_gain(x = irisX, y = y)

# formula
information_gain(formula = Species ~ ., data = iris)

# make a reduced iris data set only contains virginica and versicolor species

iris_small = iris[51:150,]
irisXsmall = iris[51:150,]
irisXsmall = iris_small[-5]

y = iris_small$Species

information_gain(x = irisXsmall, y = y)

# Run Information Gain on Train data set
# Train data set

set.seed(20)
trainlr = sample(1:nrow(iris_small),nrow(iris_small)*0.7)
trainlr

train_set = iris_small[trainlr,]
trainX = train_set[-5]
y = train_set$Species

information_gain(x = trainX, y = y)
information_gain(formula = Species ~ ., data = train_set)


# Run Information Gain on Test data set
# Test data set

test_set = iris_small[-trainlr,]
testX = test_set[-5]
y = test_set$Species

information_gain(x = testX, y = y)
information_gain(formula = Species ~ ., data = test_set)
```