R Notebook: Binary Choice Modelling (Logistic Regression)

Dr. Ashutosh Singh

Winter 2024

Contents

The following packages are needed.

- ggplot2 # very popular plotting library ggplot2
- ggthemes # themes for ggplot2
- xtable # processing of regression output
- knitr # used for report compilation and table display
- caret # confusion matrix
- pROC # confusion matrix

```r
library("ggplot2") # very popular plotting library ggplot2
library("ggthemes") # themes for ggplot2
library("xtable") # processing of regression output
library("knitr") # used for report compilation and table display
library("caret") # confusion matrix
```

```
## Loading required package: lattice
```

```r
library("pROC") # confusion matrix
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

Marketers often observe yes/no outcomes:

- Did a customer purchase a product?
- Did a customer take a test drive?
- Did a customer sign up for a credit card, renew her subscription, or respond to a promotion?

All of these kinds of outcomes are *binary* because they have only two possible overserved states: *yes* or *no*. A *logistic* model is used to fit such outcomes.

1 Basics of logistic regression

The core feature of a logistic model is that it relates the *probability* of an outcome to an *exponential function* of a predictor variable.

By modelling the *probability* of an outcome, a logistic model accomplishes two things.

- First, it more directly models what we are interested in, which is a probability or proportion, such as the likelihood of a given customer to purchase a product or the expected proportion of a segment who will respond to a promotion.

- Second, it limits the model to the appropriate range for a proportion, which is [0, 1]. A basic linear model, as generated with *lm()*, does not have such a limit. The equation for the logistic function is:

$$logistic : p(y) = \frac{e^{v_x}}{e^{v_x} + 1}$$

In this equation, the outcome of interest is y, and we compute its likelihood *p(y)* as a function of v_x. We typically estimate v_x as a function of the features *(x)* of a product, such as price. v_x can take any real value, so we are able to treat it as a continuous function in a linear model. In that case, v_x is composed of one or more coefficients of the model and indicates the importance of the corresponding features of the product.

The formula gives a value between [0, 1]. The likelihood of y is less than 50% when v_x is negative, is 50% when $v_x = 0$ and is above 50% when v_x is positive. We compute this first by hand and then switch to the equivalent *plogis()* function:

```
exp(0) / exp(0)+1 # computing logistic by hand, or using plogis()
```

```
## [1] 2
```

```
plogis(-Inf)       #infinitely low = likelihood 0
```

```
## [1] 0
```

```
plogis(2)          #moderate probability = 88% chance of outcome
```

```
## [1] 0.8807971
```

```
plogis(-0.2)       # weak likelihood
```

```
## [1] 0.450166
```

Such a model is known as a *logit* model, which determines the value of v_x from the logarithm of the relative probability of occurence of y:

$$logit : v_x = log(\frac{p(y)}{1 - p(y)})$$

R includes a built-in function *qlogis()* for the logit function:

```
log(0.88 / (1-0.88))   # moderate high likelihood
```

```
## [1] 1.99243
```

```
qlogis(0.88)           # equivalent to hand computation
```

```
## [1] 1.99243
```

In practice, the expressions of *logit model* and *logistic regression* are used interchangeably.

2 Generalised linear model (GLM)

A logistic regression model in R is fitted as a *generalised linear model (GLM)* using a process similar to linear regression with *lm()*, but with the difference that a GLM can handle dependent variables that are not normally distributed. Thus, GLM can be used to model *data counts* (such as the number of purchases), *time intervals* (such as time spent on a website), or *binary variables* (e.g., did/didn't purchase). The common feature of all GLM models is that they relate normally distributed predictors to a non-normal outcome using a function known as a *link*. This means that they are able to fit models for many different distributions using a single, consistent framework.

3 RFM (recency, frequency, monetary)

RFM is a method used for analyzing customer value. RFM stands for the three dimensions: Recency: How recently did the customer purchase? Frequency: How often do they purchase? Monetary Value: How much do they spend?

Let us load the data first.

```
RFMdata <- read.csv(file = "RFMData.csv",row.names=1)
head(RFMdata,5)
```

Each row (observation) is a separate customer who has transacted at least once before. The columns (variables) are:

1. Recency – how many days since last purchase
2. Frequency – how many times the consumer buys per year
3. Monetary – total $ amount spent per year
4. Purchase - (yes/no) whether purchase occurred

3.1 The Logit Model

The logit model restricts the output values to lie in $[0, 1]$ intervals.

Specifically, it expresses the probability of purchase by customer i as a function of coefficients $\beta_{0:3}$ and variables in the following manner:

$$P(Purchase_i) = \frac{\exp(\beta_0 + \beta_1 Recency_i + \beta_2 Frequency_i + \beta_3 Monetary_i)}{\exp(\beta_0 + \beta_1 Recency_i + \beta_2 Frequency_i + \beta_3 Monetary_i) + 1}$$

Intuitively, the utility of *choosing to buy* is

$$V_{bi} = \beta_0 + \beta_1 Recency_i + \beta_2 Frequency_i + \beta_3 Monetary_i$$

whereas utility of *choosing **not** to buy* is normalized to zero $V_{ni} = 0$, so $exp(V_n) = exp(0) = 1$ in the fraction above.

With the given formulation, we can estimate values $\beta_{0:3}$ that fit the data best. We use glm() of family="binomial".

```r
model <- glm(Purchase~Recency+Frequency+Monetary, data=RFMdata, family = "binomial")
output <- cbind(coef(summary(model))[, 1:4],exp(coef(model)))
colnames(output) <- c("beta","SE","z val.","Pr(>|z|)",'exp(beta)')
kable(output,caption = "Logistic regression estimates")
```

Table 1: Logistic regression estimates

	beta	SE	z val.	Pr(>|z|)	exp(beta)
(Intercept)	-30.2976692	8.5522913	-3.542638	0.0003961	0.000000
Recency	0.1114175	0.0309797	3.596464	0.0003226	1.117862
Frequency	0.5941268	0.2429393	2.445577	0.0144620	1.811448
Monetary	0.1677054	0.0465645	3.601572	0.0003163	1.182588

We also run the likelihood ratio test with $H_0 : \beta_1 = \beta_2 = \beta_3 = 0$ – to make sure our full logit model offers a significantly better fit than the model with just an intercept. We find that $\chi^2 = 107.14$ and $P(> |Chi|) \approx 0$, so we reject H_0.

```r
# likelihood ratio test
reduced.model <- glm(Purchase ~ 1, data=RFMdata, family = "binomial")
kable(xtable(anova(reduced.model, model, test = "Chisq")),caption = "Likelihood ratio test")
```

Table 2: Likelihood ratio test

Resid. Df	Resid. Dev	Df	Deviance	Pr(>Chi)
99	137.62776	NA	NA	NA
96	30.48715	3	107.1406	0

3.2 Predicting probabilities

Now we calculate $P(Purchase_i)$ for each individual in the data set.

```
# calculate logit probabilities
RFMdata$Base.Probability <- predict(model, RFMdata, type="response")
kable(head(RFMdata,5),row.names = TRUE)
```

	Recency	Frequency	Monetary	Purchase	Base.Probability
1	120	7	41.66	0	0.0030728
2	90	9	46.71	0	0.0008332
3	120	6	103.99	1	0.9833225
4	270	17	37.13	1	0.9999999
5	60	5	88.92	0	0.0032378

3.3 Predicting behaviour

We also calculate an indicator variable for whether individuals will purchase or not based on their predicted probabilities

$$\mathbb{1}[P(Purchase_i) \geq 0.5]$$

If individual's predicted probability is greater or equal to 0.5, we predict he will make a purchase.

```
# purchase vs. no purchase <-> p>0.5 or p<0.5
RFMdata$Predicted.Purchase <- 1*(RFMdata$Base.Probability>=0.5)
kable(head(RFMdata,5),row.names = TRUE)
```

	Recency	Frequency	Monetary	Purchase	Base.Probability	Predicted.Purchase
1	120	7	41.66	0	0.0030728	0
2	90	9	46.71	0	0.0008332	0
3	120	6	103.99	1	0.9833225	1
4	270	17	37.13	1	0.9999999	1
5	60	5	88.92	0	0.0032378	0

3.4 Evaluating the model

Now, we compute a *confusion matrix* between predicted purchases and actual purchase behaviour.

```
confusionMatrix(table(RFMdata$Predicted.Purchase,RFMdata$Purchase),positive = "1")
```

```
## Confusion Matrix and Statistics
##
##
##       0  1
##    0 51  2
##    1  4 43
##
##                Accuracy : 0.94
##                  95% CI : (0.874, 0.9777)
##     No Information Rate : 0.55
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8793
##
##  Mcnemar's Test P-Value : 0.6831
##
##             Sensitivity : 0.9556
##              Specificity : 0.9273
##           Pos Pred Value : 0.9149
##           Neg Pred Value : 0.9623
##               Prevalence : 0.4500
##           Detection Rate : 0.4300
##     Detection Prevalence : 0.4700
##        Balanced Accuracy : 0.9414
##
##         'Positive' Class : 1
##
```

We can also plot the receiver operating characteristic (ROC) curve, which illustrates the diagnostic ability of a binary logit model. It is created by plotting the true positive rate (TPR) against the false positive rate (FPR) – at various decision threshold values for prediction.

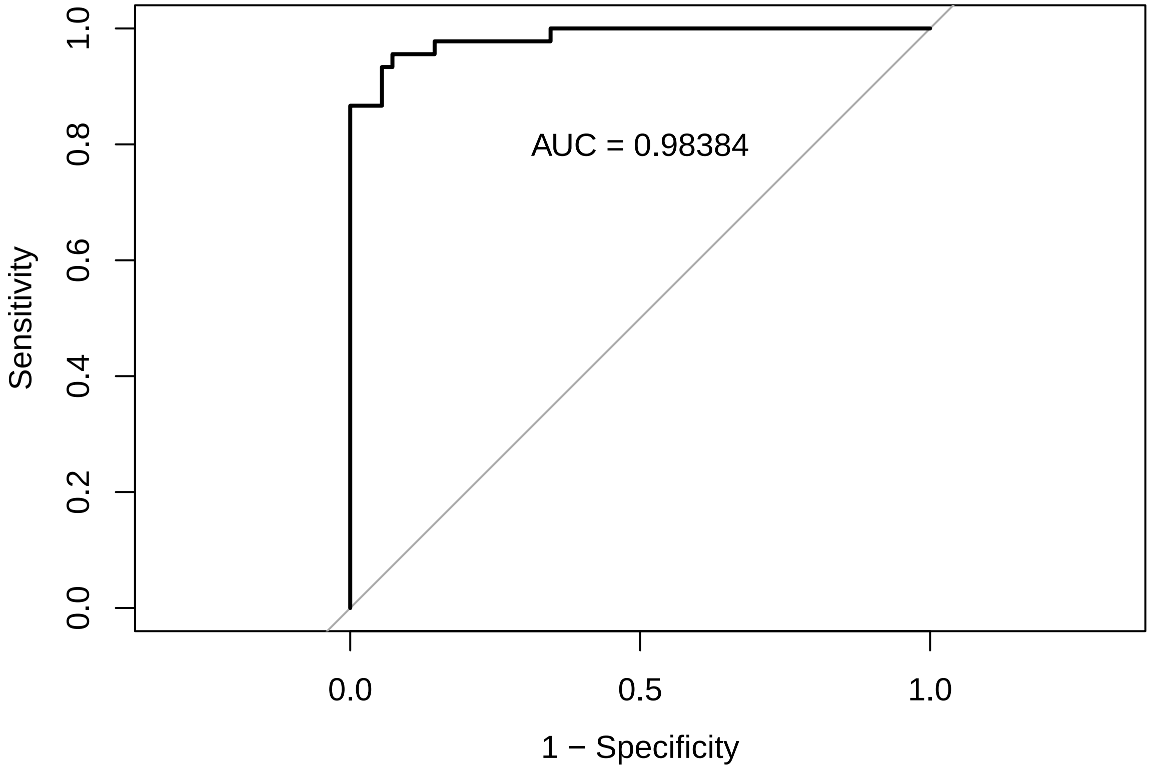
ROC curve can be quickly evaluated using the area under the curve (AUC) metric, which captures the overall quality of the classifier. The greater the AUC, the better. AUC of 1.0 represents a perfect classifier, AUC of 0.5 (diagonal line) represents a worthless classifier. As we see, the binary logit classifier does a good job of predicting purchases on the training data.

```
rocobj <- roc(RFMdata$Purchase, RFMdata$Base.Probability)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
{plot(rocobj,legacy.axes=TRUE)
text(0.5, 0.8, labels = sprintf("AUC = %.5f",rocobj$auc))}
```



Finally, we predict new probabilities under a hypothetical scenario that everyone's *Monetary* variable went up by one unit

$$V_{bi}^{new} = \beta_0 + \beta_1 Recency_i + \beta_2 Frequency_i + \beta_3(Monetary_i + 1)$$

```r
# calculate new logit probabilities (Monetary+1)
RFMdata_new <- RFMdata
RFMdata_new$Monetary <- RFMdata_new$Monetary + 1
RFMdata$New.Probability <- predict(model, RFMdata_new, type="response")
```

We compare mean new probability across individuals to the mean of old probabilities, and also calculate the lift metric.

$$p_{old} = \frac{1}{N}\sum_{i=1}^{N} P(Purchase_i) = \frac{1}{N}\sum_{i=1}^{N}\frac{\exp(V_{bi})}{\exp(V_{bi})+1}$$

$$= \frac{1}{N}\sum_{i=1}^{N}\frac{\exp(\beta_0 + \beta_1 Recency_i + \beta_2 Frequency_i + \beta_3 Monetary_i)}{\exp(\beta_0 + \beta_1 Recency_i + \beta_2 Frequency_i + \beta_3 Monetary_i) + 1}$$

$$p_{new} = \frac{1}{N}\sum_{i=1}^{N} P(Purchase_i^{new}) = \frac{1}{N}\sum_{i=1}^{N}\frac{\exp(V_{bi}^{new})}{\exp(V_{bi}^{new})+1}$$

$$= \frac{1}{N}\sum_{i=1}^{N}\frac{\exp(\beta_0 + \beta_1 Recency_i + \beta_2 Frequency_i + \beta_3(Monetary_i + 1))}{\exp(\beta_0 + \beta_1 Recency_i + \beta_2 Frequency_i + \beta_3(Monetary_i + 1)) + 1}$$

$$Lift = \frac{p_{new} - p_{old}}{p_{old}}$$

```r
# mean predicted base probability
mean(RFMdata$Base.Probability)
```

```
## [1] 0.45
```

```r
# mean new predicted probability
mean(RFMdata$New.Probability)
```

```
## [1] 0.4578851
```

```r
# lift
(mean(RFMdata$New.Probability) - mean(RFMdata$Base.Probability))/mean(RFMdata$Base.Probability)
```

```
## [1] 0.01752255
```

```r
# remove predicted purchase variable
RFMdata$Predicted.Purchase <- NULL
# data
kable(head(RFMdata,5),row.names = TRUE)
```

	Recency	Frequency	Monetary	Purchase	Base.Probability	New.Probability
1	120	7	41.66	0	0.0030728	0.0036319
2	90	9	46.71	0	0.0008332	0.0009852
3	120	6	103.99	1	0.9833225	0.9858611
4	270	17	37.13	1	0.9999999	0.9999999
5	60	5	88.92	0	0.0032378	0.0038267

4 Recap

- Logistic regression is a powerful method and a particularly good fit for many marketing problems with binary outcomes. We will cover the choice model later for modelling product choice among sets of alternatives.
- *Logistic regression* relates a binary outcome such as purchase to predictors that may include continuous and factor variable by modelling the variable's association with the probability of the outcome.
  - Although we performed logistic regression here with categorical predictors (factor variables) due to the structure of the amusement park sales data, we could also use continuous predictors in *glm()*. Just add those to the right-hand side of the model formula as we did with *lm()*
- A logistic regression model, also known as a *logit model*, is a member of the *generalized linear model* family and is fit using *glm(, family = binomial)*.
- Coefficient in a logit model can be interpreted in terms of *odds ratios*, the degree to which they are associated with the increased or decreased likelihood of an outcome. This is done simply by exponentiating the coefficients with *exp()*.
- A statistically significant result does not always mean that the model is appropriate. It is important to explore data thoroughly and construct models on the basis of careful consideration.
  - We saw that the estimated effect of promotion was positive when we estimated one model yet negative when we estimated another. This shows that it is crucial to explore data thoroughly before modelling or interpreting a model. For most marketing data, no model is ever definitive. However, through careful data exploration and consideration of multiple models, we may increase our confidence in our models and the inferences drawn from them.