MarketingHeadLDA work -- R Notebook

BUS256 Marketing Analytics – Prof Carver

This is an [R Markdown](http://rmarkdown.rstudio.com) Notebook. When you execute code within the notebook, the results appear beneath the code.

Try executing this chunk by clicking the *Run* button within the chunk or by placing your cursor inside it and pressing *Ctrl+Shift+Enter*.

1. First order of business is the usual setting of working directory and calling desired packages

setwd("C:/Users/Rob/Box Sync/My R Work/BUS256")  
library(dplyr)  
library(MASS)  
library(ggplot2)

1. Next, we read in the training and test data provided with the case study. Note that in the "R Studio Notebook" environment, we need to specify the entire file path.

train <- read.csv("C:/Users/Rob/Box Sync/My R Work/BUS256/Data/MHC\_training.csv", header=TRUE)  
test <- read.csv("C:/Users/Rob/Box Sync/My R Work/BUS256/Data/MHC\_testing.csv", header=TRUE)  
dim(train)  
## [1] 2965 11

dim(test)  
## [1] 1000 8

str(train)  
## 'data.frame': 2965 obs. of 11 variables:  
## $ Opportunity.No. : Factor w/ 2965 levels "Training1","Training10  
## $ Reporting.Status : Factor w/ 2 levels "Lost","Won": 1 1 1 1 2 1   
## $ Product : Factor w/ 7 levels "ContactSys","Finsys",..:   
## $ Industry : Factor w/ 19 levels "Agriculture",..: 4 7 5 9 ## $ Region : Factor w/ 9 levels "Africa","Americas",..: 6 ## $ Segment : Factor w/ 66 levels "ContactSys,Banks,UK",..: ## $ Relative.Strength.in.the.segment: int 57 51 79 55 32 73 56 50 31 52 ...  
## $ Profit.of.customer....Mn : Factor w/ 1398 levels " (0.000)"," (0.001)",. ## $ Sales.Value....Mn : num 6.5 9.9 7 8.9 5.7 7.9 9.6 4.6 9.1 8 ...  
## $ Profit.. : num 64 56 59 34 43 42 35 56 51 46 ...  
## $ Joint.Bid...WSES.Portion : int 59 58 48 41 63 56 58 44 68 61 ...

1. We notice that the column names are descriptive, but very long. Knowing that we'll want to refer to columns by name when we build the model, it's a good idea to rename the columns with shorter names.

The rename function in package dplyr will do the job:

train <- rename(train, Result = Reporting.Status,   
 Strength= Relative.Strength.in.the.segment,  
 CustProf = Profit.of.customer....Mn,   
 Salesval= Sales.Value....Mn,  
 Profit = Profit..,  
 Joint = Joint.Bid...WSES.Portion)  
test <- rename(test, Result = Reporting.Status,   
 Strength= Relative.Strength.in.the.segment,  
 CustProf = Profit.of.customer....Mn,   
 Profit = Profit..,  
 Salesval= Sales.Value....Mn,  
 Joint = Joint.Bid...WSES.Portion)

1. Now we can run a Linear Discriminant analysis using the Training set. The lda function (package MASS) will estimate the model.

With a large data set, the algorithm takes a while to run. In the following code chunk, we estimate a model, generate predictions of Results for each Opportunity, and then compare the predicted to actuals for the first 10 transactions.

ldamodel <- lda(Result ~ Strength + CustProf + Salesval +   
 Profit + Joint, data=train )  
# use model to predict classifications  
predlda <- predict(ldamodel, train)  
predlda$posterior <- round(predlda$posterior,4)  
cols <- c("Opportunity.No.", "Result")  
compare <- train[cols]  
compare <- cbind(compare, predlda$class, predlda$posterior)  
head(compare, n=10)

## Opportunity.No. Result predlda$class Lost Won  
## 1 Training1 Lost Lost 1.0000 0.0000  
## 2 Training2 Lost Lost 1.0000 0.0000  
## 3 Training3 Lost Lost 1.0000 0.0000  
## 4 Training4 Lost Lost 0.9995 0.0005  
## 5 Training5 Won Won 0.0001 0.9999  
## 6 Training6 Lost Lost 1.0000 0.0000  
## 7 Training7 Won Won 0.0078 0.9922  
## 8 Training8 Lost Lost 0.9928 0.0072  
## 9 Training9 Won Won 0.0002 0.9998  
## 10 Training10 Won Won 0.0018 0.9982

Finally, we can summarize the model performance with a *Confusion Matrix*, which is just a crosstabulation showing how often the model correctly predicted outcomes and how often it was incorrect. After the matrix, we compute the error, or misclassification, rate.

# table command makes a crosstab of (rows, columns)  
tablda <- table(predlda$class,train$Result) # confusion matrix  
tablda  
## Lost Won  
## Lost 1494 40  
## Won 21 1410

# calculate misclassification rate  
1-sum(diag(tablda))/sum(tablda)

## [1] 0.02057336

Finally, we repeat the entire process with the test sample to see if the same variables are effective with the test data.

ldamodel2 <- lda(Result ~ Strength + CustProf + Salesval +   
 Profit + Joint, data=test)  
testlda <- predict(ldamodel2, test)  
  
tablda2 <- table(testlda$class, test$Result) # confusion matrix  
tablda2  
## Lost Won  
## Lost 557 3  
## Won 5 435

1-sum(diag(tablda2))/sum(tablda2)

## [1] 0.008