Predictive Modeling for Direct Marketing

Bank Case

September 2019

Contents

Importance of direct marketing decision support through predictive customer response modeling is recognized

Background

- A bank is marketing Certificates of Deposit (CD) to a pool of 1,000 potential customers. The bank estimates that 13% of them are likely to buy a CD if they are contacted.

- The cost of each contact is $10. The NPV to the bank of a customer buying a CD is $50. Clearly, it is not profitable to contact all the potential customers.

- It has an unbalanced data set* (bank.csv) of some customers and whether they bought the CD or not. The data set has 16 features and a class variable for a total of 17 columns (e.g., age, job, marital, education, default, housing, balance, loan, campaign). It has 4,521 observations.

Request for Analytics Assistance

- To maximize expected profit, the bank is expected to employ a predictive model to predict whether or not a potential customer is a likely buyer.

- The data set will be used here to build the predictive model. After unbalanced data set is adjusted and decision tree is calibrated using complexity parameter and optimal cutoff, the expected profit is maximized at $2261.017.

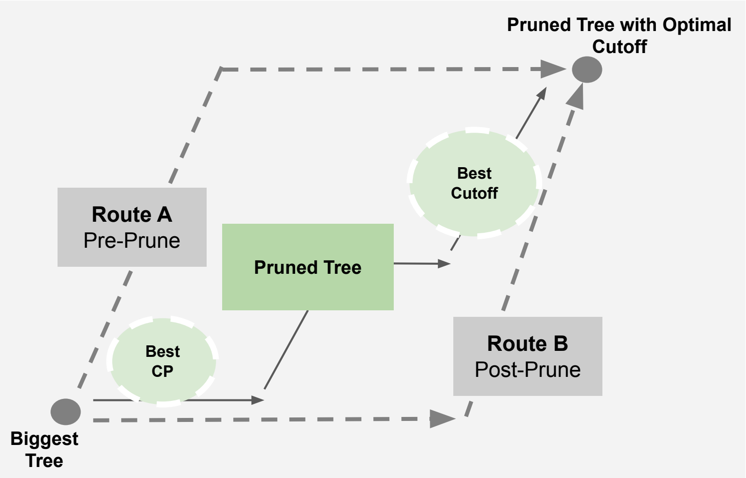
- Given the goal of driving future profit by better predicting which of past customers are best prospects to purchase CDs, the implementation of analytics team should focus on customer data organization, model performance perfection, and results monitoring.

*See dataset description in Appendix.

Contents

Decision tree technique will be applied for target selection, reaching objective of direct marketing to maximize expected profit

A Decision Tree Pruned Using Complexity Parameter (CP) and Optimal Cutoff



- Route A (Pre-Prune): fit = rpart(churn ~ ., data=b.train, control=rpart.control(xval=10, cp=bestcp))
- Route B (Post-Prune): fit = prune.rpart(fit, cp=bestcp)
- Two routes produce similar results, thus in code post-pruned way is used as example.
- In the next slide, how to get optimal tree by finding best cp and best cutoff in a balanced dataset will be discussed.

Maximum Profit by Building A Predictive Model with Good Quality

Predicted \ Actual	Positive	Negative
Positive	130 * (1 - FNR) * $40	- 870 * FPR * $10
Negative	$0	$0

Expected Profit of Using Predictive Model to Contact Customers Predicted Positive:

130 * (1 - FNR) * 40 - 870 * FPR * 10

= 5200 - 5200 * FNR(beta) - 8700 * FPR(alpha)

Analytics Design & Process - Workstream Highlights*

1 Build a complex tree

```
library(rpart)
# xval = 10 : cross-sample with 10 folds to determine error rate at each node
# minsplit = 10  : min number of observations to attempt split
# cp = 0  : minimum improvement in complexity parameter for splitting
fit = rpart(y ~ ., # formula
        data=b.train, # dataframe used
        control=rpart.control(xval=10, minsplit=10, cp = 0.0))
nrow(fit$frame) # the biggest tree have 193 nodes
```

2 Post prune with best complexity parameter

```
bestcp =
fit$cptable[which.min(fit$cptable[,"xerror"]),
"CP"]
# lowest error occurs at cp = 0.01186944

# post-pruning
fit.post = prune.rpart(fit, cp=bestcp)
nrow(fit.post$frame)
# the pruned tree have 17 nodes

# compute the confusion matrices in test
set
cm = confusionMatrix(table(predict(fit.post,
b.test, type="class"),  b.test$y),
positive='yes')

profit = 5200 - 5200 *
(1-cm$byClass["Sensitivity"][[1]]) - 8700 *
(1-cm$byClass["Specificity"][[1]])
# profit = $1600.222
```

3 Create a balanced dataset

```
# splitting the training
dataset
b.train.yes = subset(b.train,
y == 'yes')
b.train.no = subset(b.train, y
== 'no')

# take a subsample from
b.train.no with the same
number of observations as
b.train.yes.
set.seed(234)
no =
sample(1:nrow(b.train.no),nr
ow(b.train.yes))
b.train.no = b.train.no[no,]

# Combine b.train.yes and
subsample of b.train.no to
make a new balanced
training data frame
b.bal =  rbind(b.train.yes,
b.train.no)
```

4 Retrain model using balanced dataset

```
# bal.fit has 95 nodes

# lowest error occurs at
bal.bestcp = 0.007418398

# post-prund tree has
bal.fit.post 19 nodes

# profit = $2186.02
```

5 Find Optimal Threshold for Cutoff

```
library(ROCR)
# start by predicting prob instead of class
y.pred = as.data.frame(predict(bal.fit.post, b.bal, type="prob"))

# first step is to compute the score object
y.pred.score = rediction(y.pred[,2], b.bal$y)

# now we will determine the optimal cutoff
# rather than use  ROCR curve directly, the cost built into ROC is used
y.cost = performance(y.pred.score, measure="cost", cost.fn=5200, cost.fp=8700)
cutoff.best = y.cost@x.values[[1]][which.min(y.cost@y.values[[1]])] # 0.7142857

# make predictions using this cutoff rate for the test set
y.pred.test = predict(bal.fit.post, b.test, type="prob")
y.pred.test.cutoff = ifelse(y.pred.test[,2] > cutoff.best,'yes','no')

# now find the profit for the test data set using the optimal cutoff
cm = confusionMatrix(table(pred=y.pred.test.cutoff,actual = b.test$y), positive='yes')
profit = 5200 - 5200 * (1-cm$byClass["Sensitivity"][[1]]) - 8700 * (1-cm$byClass["Specificity"][[1]])
# profit =  $2261.017
```

*See complete code on https://github.com/Shelly-Yang/Predictive-Modeling-for-Direct-Marketing.

Contents

Dataset Description

Feature	Description
age	Age of customer in years (integer)
job	Job category, one of 12 possible values, (string)
marital	Marital status, one of 3 possible values, (string)
education	Level of education, one of 4 possible values, (string)
default	Whether the customer has previously defaulted on a loan (yes or no)
balance	Average daily balance (numeric)
housing	Whether the customer has a mortgage with the bank (yes or no)
loan	Whether the customer has a revolving loan credit with the bank (yes or no)
day	Date of last contact
month	Month of last contact
contact	Method of last contact, one of three values (string)
duration	Duration of last contact, in minutes, (integer)
campaign	Campaign identifier
pdays	Days to prior contact
previous	Number of previous contacts
poutcome	Outcome of previous
y	Whether the customer purchased a CD or not (yes, or no), the class variable