Module 5 Project: Model Evaluation and Deployment

Ali Ladha

Notes on compiling this document:

- Change the information in the yaml header above: title and author.
- Make sure the output argument is set correctly. It should read: output: html_document or output: word_document.
- Once you are finished writing the code necessary to answer the questions in the quiz, clear your environment by clicking on the broom icon in the environment pane (upper right quadrant).
- Run each code chunk individually (click the green arrow icon in the upper right of the chunk). Start at the top of this document and proceed sequentially to the bottom. Fix any code errors you find.
- Once your code is error-free, click “knit” in the menu above. Your document should compile to HTML, if the output is set to “html_document” (or to word if the output is set to “word_document”).

In the code chunk above (entitled “setup”) echo is set to TRUE. This means that the code in your chunks will be displayed, along with the results, in your compiled document.

Load and Transform Data

Below is code to clean and prepare the data set for modeling. Before running that code, follow these preparatory steps:

1. Download the RMarkdown template and the data sets for the assignment from Canvas.
2. Copy or move these files from your downloads folder to a folder dedicated to this class–say, MKTG-6487.
3. You need to define this folder as your “working directory.” To do so, navigate to that folder using the files tab in the lower right quadrant in RStudio. (You should see your files you moved into this folder in the previous step.) Click the “More” button in the menu under the Files tab and select “Set As Working Directory.”

Once the files are in the right location on your computer then run this code to clean and format the data:

```
# You must run this code to format the data set properly!

advise_invest <- read_csv("adviseinvest.csv")  |>           # Download data
  select(-product) |>                                       # Remove the product column
  filter(income > 0,                                         # Filter out mistaken data
         num_accts < 5) |>
  mutate(answered = factor(ifelse(answered==0, "no","yes"),   # Turn answered into yes/n
o factor
                           levels  = c("no", "yes")),
         female = factor(female),                            # Make categorical variabl
es into factors
         job = factor(job),
         rent = factor(rent),
         own_res = factor(own_res),
         new_car = factor(new_car),
         mobile = factor(mobile),
         chk_acct = factor(chk_acct),
         sav_acct = factor(sav_acct))
```

```
## Rows: 29502 Columns: 14
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## dbl (14): answered, income, female, age, job, num_dependents, rent, own_res,...
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

And here is code to load the data set of prospective customers from your working directory. Note that in order to use this data set for prediction, the variables need to be formatted exactly the same as in the data used to fit the model. It does not include a target variable because the event of answering or not answering has not happened yet for scheduled customers.

```
prospective <- read_csv("customer_data.csv") |>
  mutate(female = factor(female),
         job = factor(job),
         rent = factor(rent),
         own_res = factor(own_res),
         new_car = factor(new_car),
         mobile = factor(mobile),
         chk_acct = factor(chk_acct),
         sav_acct = factor(sav_acct))
```

```
## Rows: 1000 Columns: 13
## ── Column specification ────────────────────────────────────────────────────
## Delimiter: ","
## chr  (1): customer_id
## dbl (12): income, female, age, job, num_dependents, rent, own_res, new_car, ...
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Questions

Read the instructions for this phase of the project at Canvas.

Q1.

```
tree_model <- rpart(answered ~., data = advise_invest)
tree_model
```

```
## n= 29499
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##     1) root 29499 13375 yes (0.4534052 0.5465948)
##       2) chk_acct=0,1,2 19199  8000 no (0.5833116 0.4166884)
##         4) income>=79840 1728   192 no (0.8888889 0.1111111) *
##         5) income< 79840 17471  7808 no (0.5530880 0.4469120)
##          10) mobile=0 16383  6848 no (0.5820057 0.4179943)
##            20) sav_acct=0,1 13311  4928 no (0.6297799 0.3702201)
##              40) income>=5900 13055  4672 no (0.6421295 0.3578705)
##                80) age< 25.5 3071   640 no (0.7915988 0.2084012)
##                 160) chk_acct=0,1 2879   448 no (0.8443904 0.1556096) *
##                 161) chk_acct=2 192     0 yes (0.0000000 1.0000000) *
##                81) age>=25.5 9984  4032 no (0.5961538 0.4038462)
##                 162) income< 10310 1280   256 no (0.8000000 0.2000000) *
##                 163) income>=10310 8704  3776 no (0.5661765 0.4338235)
##                   326) income>=11000 8448  3520 no (0.5833333 0.4166667)
##                     652) age>=27.5 7424  2880 no (0.6120690 0.3879310)
##                      1304) income< 13875 1152   192 no (0.8333333 0.1666667) *
##                      1305) income>=13875 6272  2688 no (0.5714286 0.4285714)
##                        2610) income>=17960 5568  2176 no (0.6091954 0.3908046)
##                          5220) female=1 768   128 no (0.8333333 0.1666667) *
##                          5221) female=0 4800  2048 no (0.5733333 0.4266667)
##                           10442) job=2 2944  1024 no (0.6521739 0.3478261)
##                             20884) income>=26300 1984   448 no (0.7741935 0.2258065)
*
##                             20885) income< 26300 960   384 yes (0.4000000 0.6000000)
##                               41770) new_car=1 384    64 no (0.8333333 0.1666667) *
##                               41771) new_car=0 576    64 yes (0.1111111 0.8888889) *
##                           10443) job=0,1,3 1856   832 yes (0.4482759 0.5517241)
##                             20886) income< 21525 384    64 no (0.8333333 0.1666667) *
##                             20887) income>=21525 1472   512 yes (0.3478261 0.6521739)
*
##                        2611) income< 17960 704   192 yes (0.2727273 0.7272727) *
##                     653) age< 27.5 1024   384 yes (0.3750000 0.6250000) *
##                   327) income< 11000 256     0 yes (0.0000000 1.0000000) *
##              41) income< 5900 256     0 yes (0.0000000 1.0000000) *
##            21) sav_acct=2,3,4 3072  1152 yes (0.3750000 0.6250000)
##              42) age< 34 1856   896 yes (0.4827586 0.5172414)
##                84) num_accts>=1.5 1216   448 no (0.6315789 0.3684211) *
##                85) num_accts< 1.5 640   128 yes (0.2000000 0.8000000) *
##              43) age>=34 1216   256 yes (0.2105263 0.7894737) *
##          11) mobile=1 1088   128 yes (0.1176471 0.8823529) *
##       3) chk_acct=3 10300  2176 yes (0.2112621 0.7887379)
##         6) income>=38910 2240  1088 no (0.5142857 0.4857143)
##          12) num_accts< 1.5 448     0 no (1.0000000 0.0000000) *
##          13) num_accts>=1.5 1792   704 yes (0.3928571 0.6071429)
##            26) new_car=1 448    64 no (0.8571429 0.1428571) *
```

```
##            27) new_car=0 1344   320 yes (0.2380952 0.7619048) *
##         7) income< 38910 8060  1024 yes (0.1270471 0.8729529) *
```

```
#this is the tree_model

table(predicted = predict(object = tree_model, type = "class"), observed = advise_invest
$answered)
```

```
##          observed
## predicted    no   yes
##       no  10367  2304
##       yes  3008 13820
```

```
#this is the confusion matrix

#in this case 13820 is the TRUE Positive since In the AdviseInvest scenario, TPs are cus
tomers that you have predicted will answer the phone and do answer, thus providing an op
portunity for your sales reps to make a sale.)
```

Q2

```
(75 * 13820) - (25 * 3008)
```

```
## [1] 961300
```

Q3

```
contact_list_advise <- prospective %>% mutate(prob_yes = predict(tree_model, newdata = p
rospective, type = "prob")[,1]) %>% filter(prob_yes >=.3) %>% select(customer_id, prob_y
es) %>% arrange(desc(prob_yes))

contact_list_advise
```

```
## # A tibble: 502 × 2
##    customer_id prob_yes
##    <chr>          <dbl>
##  1 U6236              1
##  2 H8469              1
##  3 V6007              1
##  4 R98                1
##  5 P6174              1
##  6 N4460              1
##  7 T294               1
##  8 P2126              1
##  9 Q3081              1
## 10 O5046              1
## # ℹ 492 more rows
```

Q4

AdviseInvest faces a situation in which a significant percentage of its customers fail to answer a phone call from a representative after completing the online process. This causes loss in revenue as those customers are unable to purchase a plan while also causing unnecessary expenses as some representatives are available with no tasks to complete. A classification tree was completed previously which discussed the importance of obtaining a customers mobile number in order to increase the answer rate for phone calls. Implementing this recommendation is likely increase profit. A cost benefit matrix was completed as well showing the potential of the company if it increases its answer rate. With the supervised trained model, we also created a contact list for customers who are likely to answer the phone. It is essential to prioritize contacting those customers first and comparing it to the previous answer rate based on historical data.