Creative Gaming

true true true

Preliminaries

Determine notebook defaults:

Load packages:

Read in the data:

```
# use load("filename.Rdata") for .Rdata files
# rm(list=ls())
load("/Users/dain/Programs/R_Projects/MKTG_482_HW4/creative_gaming.Rdata")
```

Assignment answers

Questions: 1. How to convert csv -> Rdata? 2. How to set up Rdata into multiple importable variables (like this project)? 3. How come we're using the same data for nnet for training & testing? 4. How are my convert & predicted convert using the models so far off?

Part 1: Exploratory Analytics

First, to gain an understanding of whether the data is appropriate for predictive analytics, you decide to engage in some exploratory analytics. Please answer the following questions:

Hint: install the "skimr" package and use the "skim_without_charts()" command (don't forget to add library(skimr) in the header of your Rmd file). Don't worry about formatting when you knit to pdf.

1. What is the organic probability of converting to Zalon?

```
paste("OrganicProb(convert): ", percent(mean(cg_organic$converted), 0.01))
```

```
[1] "OrganicProb(convert):  5.75%"
```

2. For each feature, show basic summary statistics.

```
skim_without_charts(cg_organic)
```

Table 1: Data summary

Name	cg_organic
Number of rows	30000
Number of columns	20
Column type frequency:	
factor	6
numeric	14

Table 1: Data summary

Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
AcquiredSpaceship	0	1	FALSE	2	0: 21695, 1: 8305
AcquiredIonWeapon	0	1	FALSE	2	0: 29439, 1: 561
PurchasedCoinPackSmall	0	1	FALSE	2	0: 19857, 1: 10143
PurchasedCoinPackLarge	0	1	FALSE	2	0: 22061, 1: 7939
UserNoConsole	0	1	FALSE	2	0: 24546, 1: 5454
UserHasOldOS	0	1	FALSE	2	0: 27411, 1: 2589

Variable type: numeric

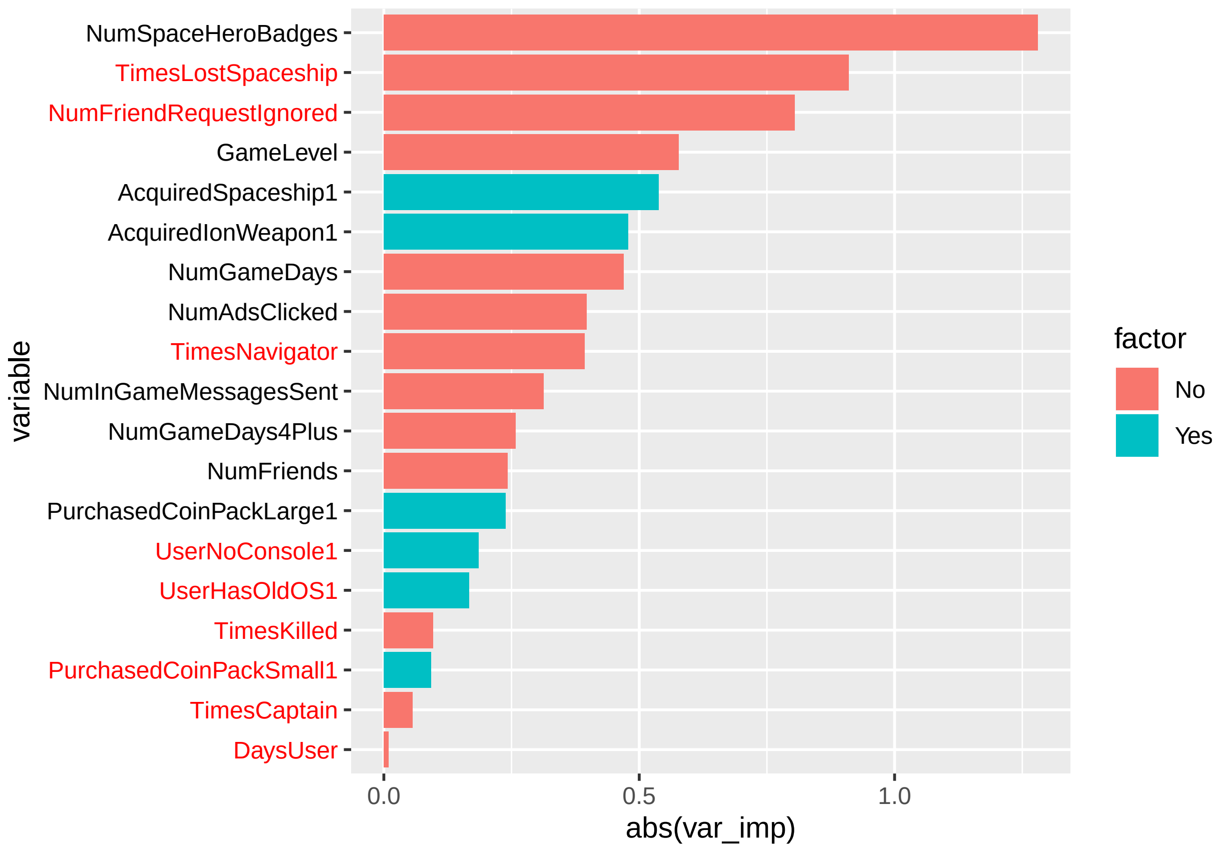
skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
converted	0	1	0.06	0.23	0	0	0	0	1
GameLevel	0	1	6.25	2.77	1	4	7	9	10
NumGameDays	0	1	12.24	7.10	1	6	13	18	28
NumGameDays4Plus	0	1	1.26	3.19	0	0	0	0	24
NumInGameMessagesSent	0	1	73.78	107.44	0	0	26	112	1227
NumSpaceHeroBadges	0	1	0.44	1.52	0	0	0	0	12
NumFriendRequestIgnored	0	1	29.59	33.99	0	0	16	53	121
NumFriends	0	1	47.73	94.33	0	0	5	43	486
TimesLostSpaceship	0	1	4.44	11.55	0	0	0	4	298
TimesKilled	0	1	0.29	3.42	0	0	0	0	178
TimesCaptain	0	1	1.58	8.77	0	0	0	0	429
TimesNavigator	0	1	1.40	7.95	0	0	0	0	545
NumAdsClicked	0	1	9.49	7.40	0	4	8	12	38
DaysUser	0	1	2626.37	661.43	244	2162	2557	3105	4139

Part 2: Predictive Model

1. Create a training and test sample based on the "cg_organic" dataframe. Please use the "sample_train_org" vector to select observations for both new dataframes. You can use this syntax:

```
cg_organic_train <- cg_organic[sample_train_org,]
cg_organic_test <- cg_organic[-sample_train_org,]

lrm <- glm(converted ~ ., family=binomial(logit), data=cg_organic_train)
# plot_model(lrm, show.values = TRUE, transform = NULL)
# plot_model(lrm, show.values = TRUE, transform = NULL, type = "eff")
varimp.logistic(lrm) %>% plotimp.logistic()
```



```
# A tibble: 19 x 6
   variable                factor  var_imp var_imp_lower var_imp_upper p_value
   <chr>                   <chr>     <dbl>         <dbl>         <dbl>   <dbl>
 1 NumSpaceHeroBadges      No       1.28          1.20          1.36     0
 2 TimesLostSpaceship      No      -0.910        -1.14         -0.676    0
 3 NumFriendRequestIgnored No      -0.805        -0.976        -0.633    0
 4 GameLevel               No       0.577         0.424         0.730    0
 5 AcquiredSpaceship1      Yes      0.538         0.397         0.679    0
 6 AcquiredIonWeapon1      Yes      0.478         0.0798        0.876    0.019
 7 NumGameDays             No       0.469         0.318         0.621    0
 8 NumAdsClicked           No       0.398         0.289         0.506    0
 9 TimesNavigator          No      -0.393        -0.606        -0.179    0
10 NumInGameMessagesSent   No       0.313         0.158         0.467    0
11 NumGameDays4Plus        No       0.258         0.149         0.366    0
12 NumFriends              No       0.242         0.129         0.355    0
13 PurchasedCoinPackLarge1 Yes      0.239         0.0952        0.383    0.001
14 UserNoConsole1          Yes     -0.185        -0.364        -0.00592  0.043
15 UserHasOldOS1           Yes     -0.166        -0.410         0.0769   0.18
16 TimesKilled             No      -0.0964       -0.297         0.105    0.347
17 PurchasedCoinPackSmall1 Yes     -0.0924       -0.230         0.0452   0.188
18 TimesCaptain            No      -0.0565       -0.183         0.0703   0.383
19 DaysUser                No      -0.00907      -0.141         0.123    0.893
```

What is the training/test split?

```r
tot_rows <- nrow(cg_organic_train) + nrow(cg_organic_test)
paste(
   "Training Data: ", percent(nrow(cg_organic_train) / tot_rows),
  "Test Data: ", percent(nrow(cg_organic_test) / tot_rows))
```

```
[1] "Training Data:  70% Test Data:  30%"
```

2. Train a logistic regression model using all features
    a. What are the 5 most important features?

```
varimp.logistic(lrm) %>%
  mutate(abs_var_imp=(abs(var_imp))) %>%
  arrange(desc(abs_var_imp)) %>%
  select(variable, factor, abs_var_imp, var_imp) %>%
  slice(0:5)
```

```
# A tibble: 5 x 4
  variable                factor abs_var_imp var_imp
  <chr>                   <chr>        <dbl>   <dbl>
1 NumSpaceHeroBadges      No           1.28    1.28 
2 TimesLostSpaceship      No           0.910  -0.910
3 NumFriendRequestIgnored No           0.805  -0.805
4 GameLevel               No           0.577   0.577
5 AcquiredSpaceship1      Yes          0.538   0.538
```

   b. For each feature of the top 5, summarize what you learn from marginal effect plots for those variables.

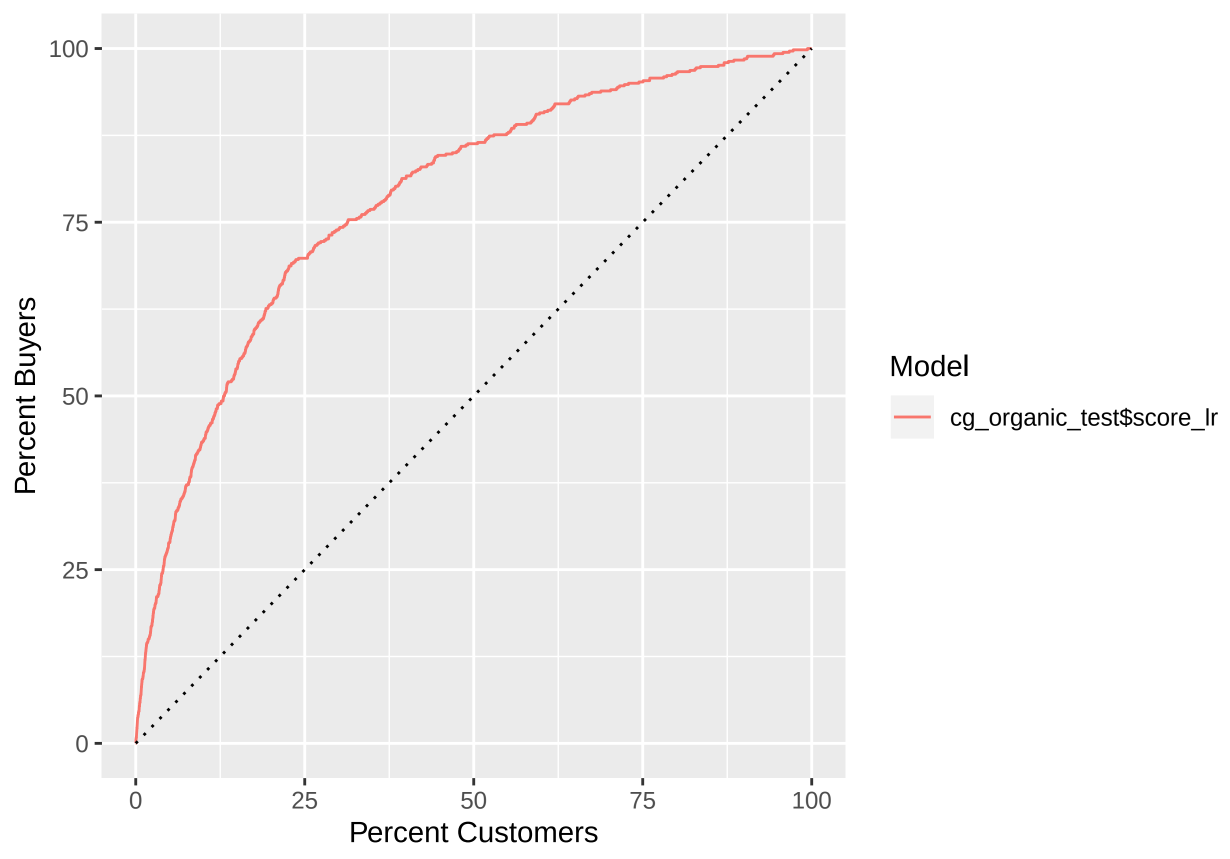
```
"
1. NumSpaceHeroBadges: [POS] - As number of space hero badges go up, probability of
   conversion goes up almost to 100% (S-shaped)
2. TimesLostSpaceship: [NEG] - As # of times lost spaceship goes up, probability of
   conversion goes down from 4% to almost 0%
3. NumFriendRequestIgnored: [NEG] - As number of friend requests ignored goes up,
   probability of conversion does down
4. GameLevel: [POS] - As Game Level goes up, so does probability - from 2% to 5%+
5. AcquiredSpaceship1: [POS] - Yes/No: If Acquired Spacehip, probability jumps from ~3% to ~5%
"
```

```
[1] "\n1. NumSpaceHeroBadges: [POS] - As number of space hero badges go up, probability of \n   convers
```

   c. Plot the gains curve in the test sample and report the AUC of the model.

```
cg_organic_test <- cg_organic_test %>% mutate(score_lr = predict(lrm, newdata=cg_organic_test, type="re

gplot <- gainsplot(cg_organic_test$score_lr, label.var = cg_organic_test$converted)
```



```r
paste("AUC: ", gplot$auc)
```

```
[1] "AUC:  0.803"
```

Part 3: The Ad-Experiment

You have finished building a logistic regression model to predict what kind of Space Pirates users were most likely to purchase the Zalon campaign. After debriefing Mi Haruki on the performance of the predictive model, she lays out the next steps:

"We will test the effectiveness of the in-app ad and the predictive model by exposing a random 150,000 customers to a 2-week ad campaign and measuring their conversion to Zalon over the next 2 months. At the same time, we will randomly pick another 30,000 customers and observe their organic upgrade behavior over the same period, having not served them an in-app ad. I want you to compare three groups based on this data.

Group 1: The randomly picked 30,000 Space Pirates users who did not receive in-app ads during the experimental period. Group 2: A randomly picked 30,000 Space Pirates users among the 150,000 who were served in-app ads for Zalon. Group 3: A model-selected 30,000 Space Pirates users among the 120,000 (after taking out Group 2) who were served in-app ads for Zalon.

Please report back to me how well the ads are working in terms of conversion rates and profits and by how much the model improves these metrics, all based on targeting 30,000 customers. To calculate profits, please use revenues of $14.99 from selling Zalon. The cost of serving ads to a consumer for 2 weeks is $1.50 in lost coin purchases."

1. Calculate the response rate and profit of group 1.The dataframe is called "cg_organic_control"

```r
revenue_per <- 14.99
cost_per <- 1.50
```

```
profit_func <- function(df, revenue_per, cost_cost){
   resp_rate   <- mean(df$converted)
   num_purch   <- nrow(df)  * resp_rate
   tot_revenue <- num_purch * revenue_per
   tot_cost    <- num_purch * cost_per
   data.frame("NumTargets"=nrow(df), "NumPurchasers"=num_purch, "ResponseRate"= percent(resp_rate, 0.01
}

profit_func(cg_organic_control, revenue_per, cost_per)
```

```
  NumTargets NumPurchasers ResponseRate    Revenue   Cost     Profit
1      30000          1706        5.69% $25,572.94 $2,559 $23,013.94
```

2. Calculate the response rate and profit of group 2. To randomly select 30,000 customers, please use the “sample_random_30000” vector to select observations from the 150,000 row dataframe “cg_ad_treatment” which contains the customers who were exposed to the ad campaign. You can use this syntax to create the sample:

```
cg_ad_random <- cg_ad_treatment[sample_random_30000,]

profit_func(cg_ad_random, revenue_per, cost_per)
```

```
  NumTargets NumPurchasers ResponseRate    Revenue      Cost     Profit
1      30000          3913       13.04% $58,655.87 $5,869.50 $52,786.37
```

3. Calculate the response rate and profit of group 3. To do this please:

(a) Use the logistic regression model you trained in Part 2 to score the 120,000 customers in “cg_ad_treatment” who remained after sampling 30,000 in “cg_ad_test”. You can use this syntax to create the sample for scoring:

```
cg_ad_scoring <- cg_ad_treatment[-sample_random_30000,]

cg_ad_scoring <- cg_ad_scoring %>% mutate(score_lr = predict(lrm, newdata=cg_ad_scoring, type="response

# TODO: DID I MUCK SOMETHING UP?!? `profit_func` isn't using $score_lr, it's using $converted
cg_ad_scoring %>% summarise(resp_rate=mean(converted), pred_resp_rate=mean(score_lr))
```

```
  resp_rate pred_resp_rate
1     0.131         0.0804
```

```
profit_func(cg_ad_scoring, revenue_per, cost_per)
```

```
  NumTargets NumPurchasers ResponseRate  Revenue      Cost   Profit
1     120000         15735       13.11% $235,868 $23,602.50 $212,265
```

(b) Select the 30,000 customers with best scores and use only these 30,000 (who correspond to group 3) to compute conversion rates and profits of group 3 in the next question.

```
quartileCG <- cg_ad_scoring %>%
   mutate(pred_buyer_quartile = ntile(-score_lr, 4)) %>%
   group_by(pred_buyer_quartile) %>%
   summarise(num_cust=n(), num_buyers=sum(converted), resp_rate=sum(converted)/n(), pred_resp_rate=mean

# TODO: This has to be wrong... Resp Rates are way higher than Predicted Response Rate"
quartileCG
```

```
# A tibble: 4 x 5
```

```
  pred_buyer_quartile num_cust num_buyers resp_rate pred_resp_rate
                <int>    <int>      <int>     <dbl>          <dbl>
1                   1    30000       6452    0.215          0.233
2                   2    30000       4686    0.156          0.0477
3                   3    30000       2771    0.0924         0.0271
4                   4    30000       1826    0.0609         0.0137
```

```
profitByQuartileOrig <- quartileCG[1,] %>%
   mutate(revenue=num_buyers * revenue_per, cost=num_buyers * cost_per, profit=revenue-cost) %>%
   select(pred_buyer_quartile, num_cust, num_buyers, resp_rate, revenue, cost, profit)
profitByQuartileOrig
```

```
# A tibble: 1 x 7
  pred_buyer_quartile num_cust num_buyers resp_rate revenue  cost profit
                <int>    <int>      <int>     <dbl>   <dbl> <dbl>  <dbl>
1                   1    30000       6452     0.215  96715.  9678 87037.
```

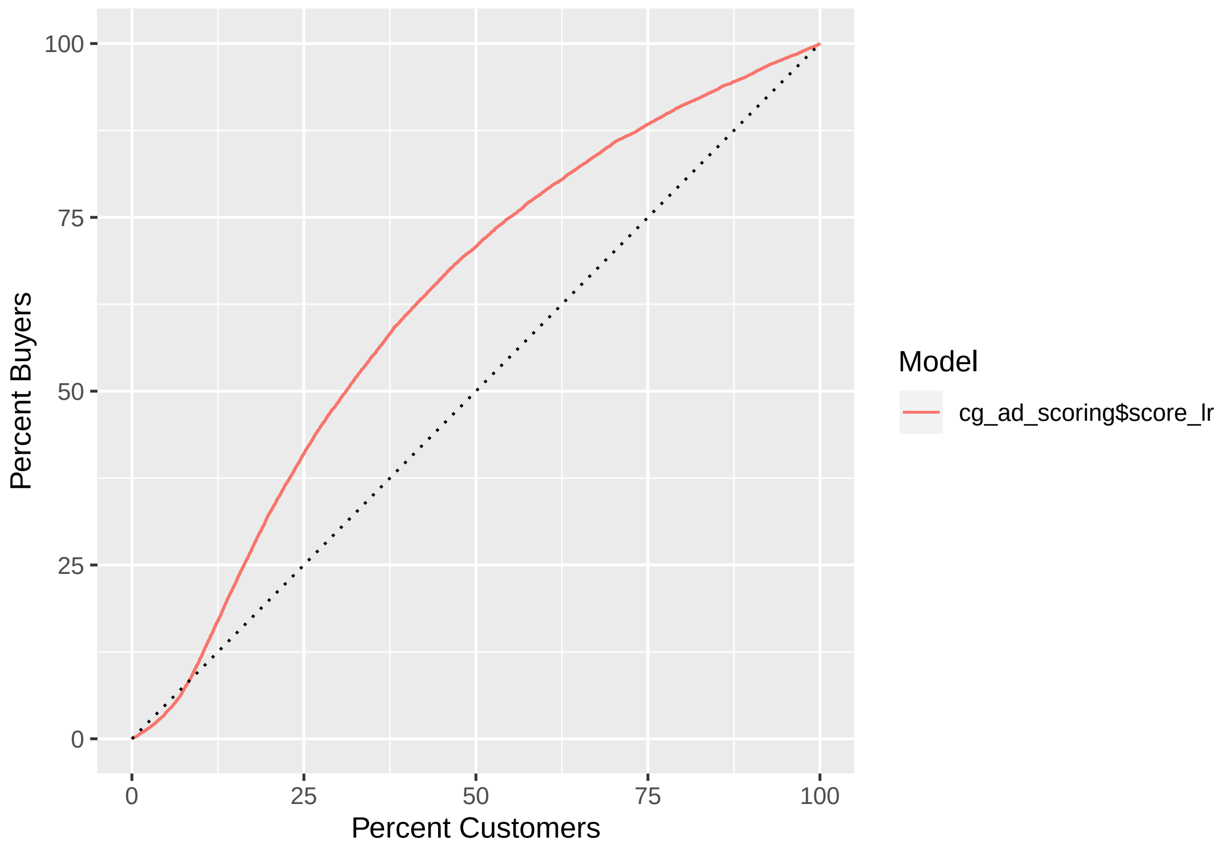
4. Answer Mi Haruki's question: "Please report back to me how well the ads are working in terms of conversion rates and profits and by how much the model improves these metrics, all based on targeting 30,000 customers."

```
"
The ads have increased conversion rates from ~5.7% to ~13.1%. The resulting profits have
increased 2.3x over the control group. [TBD on how the model affects this]
"
```

```
[1] "\nThe ads have increased conversion rates from ~5.7% to ~13.1%. The resulting profits have \nincrea
```

5. Plot the gains curve in the cg_ad_scoring sample you scored in Part 3 Q3 (which used the model trained in Part 2 Q2). Also report the AUC of the model. Compare the gains curve and AUC to the ones you calculated in Part 2 Q2c. Why are they different?

```
gplot <- gainsplot(cg_ad_scoring$score_lr, label.var = cg_ad_scoring$converted)
```



```r
paste("AUC: ", gplot$auc)
```

```
[1] "AUC:  0.644"
```

6. What is the purpose of group 1 given that we already had data on organic conversions?

```r
"TBD - IDK"
```

```
[1] "TBD - IDK"
```

Part 4: Better Data, Better Predictions

Mi Hiruki called for a meeting to plan the next steps. She explained: "Before we roll out the campaign globally, we want to see whether we can use the experimental data to retrain the model. The idea is to model trial in response to the in-app ad, not just organic conversion, as we did initially. We know that the in-app ad, on average, increases Zalon conversions. However, if the in-app ad works for people who would not have purchased the Zalon campaign organically, updating the model based on the in-app ad data should improve predictive performance. Let's retrain the model based on the randomly chosen Space Pirates users we messaged in the experiment and see how well the updated model compares to the original model in a test sample."

1. Retrain the logistic regression model from Part 2 (which was trained on "cg_organic_train") on the sample of customers who were exposed to the ad campaign ("cg_ad_random"). Instead of creating separate training/test datasets, please use the full 30,000 sample "cg_ad_random" you created in Part 3, Q2 to train the model.

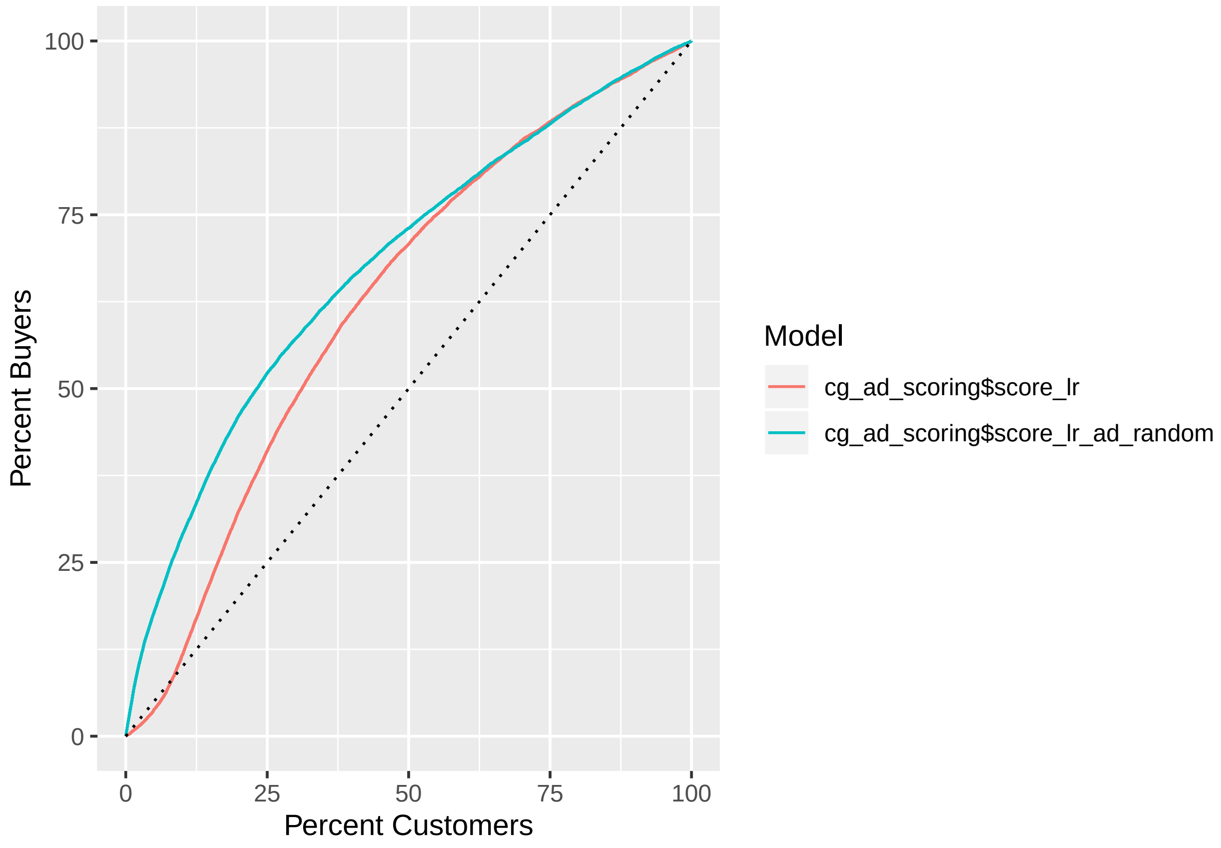
```r
cg_ad_random2 <- cg_ad_treatment[sample_random_30000,]
lr_ad_random <- glm(converted ~ ., family=binomial(logit), data=cg_ad_random2)
```

2. Compare the performance of the original "organic" model from Part 2 and the "ad" model on the

"cg_ad_scoring" sample you created in Part 3, Q3. Use gains curves and AUC to make the comparison. What do you find?

```r
cg_ad_scoring <- cg_ad_scoring %>%
   mutate(
      score_lr_ad_random = predict(lr_ad_random, newdata=cg_ad_scoring, type="response"),
   )

gainsplot(cg_ad_scoring$score_lr, cg_ad_scoring$score_lr_ad_random, label.var = cg_ad_scoring$converted
```



```
# A tibble: 2 x 2
  model                              auc
  <chr>                            <dbl>
1 cg_ad_scoring$score_lr           0.644
2 cg_ad_scoring$score_lr_ad_random 0.703
```

3. Calculate the profit improvement of using a model trained on ad treatment instead of organic data to target the best 30,000 customers in the "cg_ad_scoring" sample.

```r
quartileNew <- cg_ad_scoring %>%
   mutate(pred_buyer_quartile = ntile(-score_lr_ad_random, 4)) %>%
   group_by(pred_buyer_quartile) %>%
   summarise(num_cust=n(), num_buyers=sum(converted), resp_rate=sum(converted)/n(), pred_resp_rate=mean

# TODO: This has to be wrong... Resp Rates are way higher than Predicted Response Rate
quartileNew
```

```
# A tibble: 4 x 5
  pred_buyer_quartile num_cust num_buyers resp_rate pred_resp_rate
```

```
                    <int>    <int>     <int>    <dbl>         <dbl>
1                       1    30000      8217    0.274         0.139
2                       2    30000      3281    0.109         0.0986
3                       3    30000      2360    0.0787        0.0560
4                       4    30000      1877    0.0626        0.0274
```

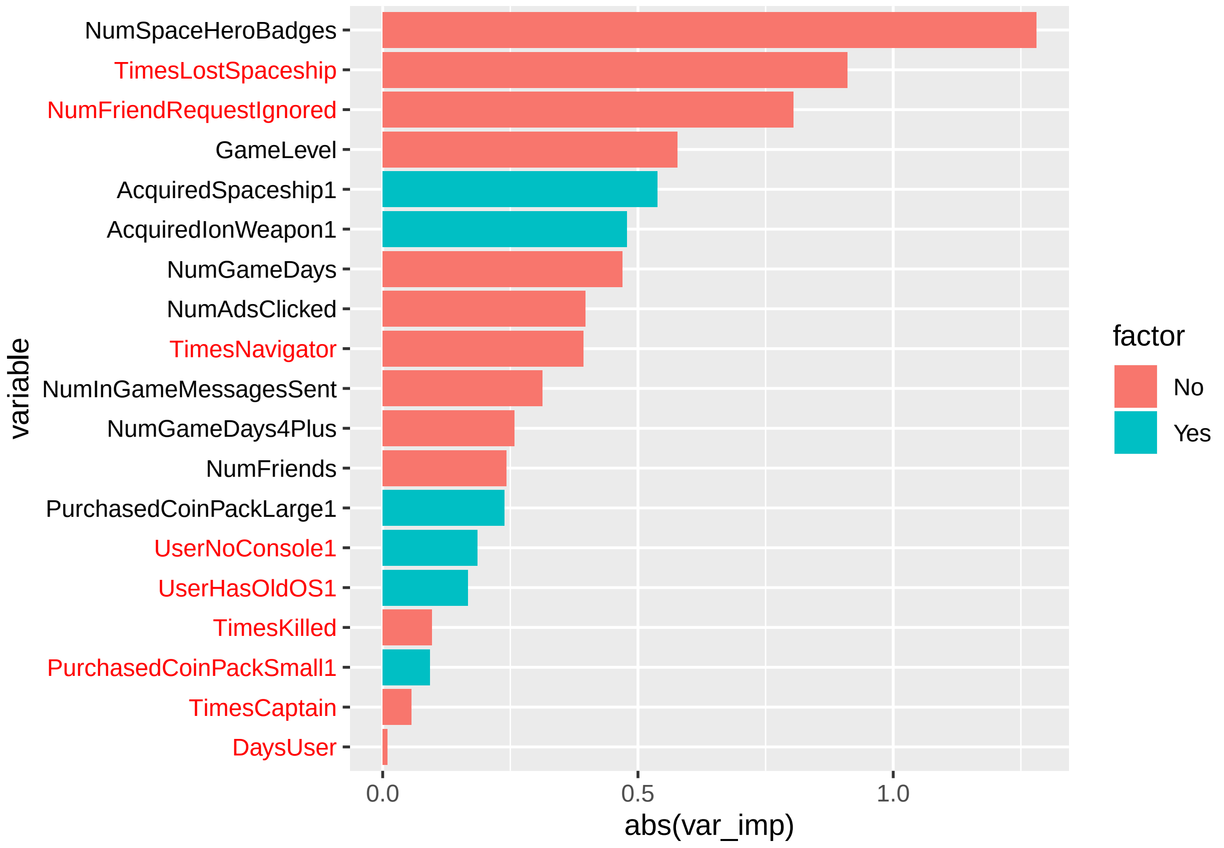
```
profitByQuartileNew <- quartileNew[1,] %>%
   mutate(revenue=num_buyers * revenue_per, cost=num_buyers * cost_per, profit=revenue-cost) %>%
   select(pred_buyer_quartile, num_cust, num_buyers, resp_rate, revenue, cost, profit)

paste("Orig Profit: ", dollar(profitByQuartileOrig$profit),
      "New Profit: ", dollar(profitByQuartileNew$profit),
      "Improvement: ", dollar(profitByQuartileNew$profit -profitByQuartileOrig$profit ))
```

```
[1] "Orig Profit:  $87,037.48 New Profit:  $110,847 Improvement:  $23,809.85"
```

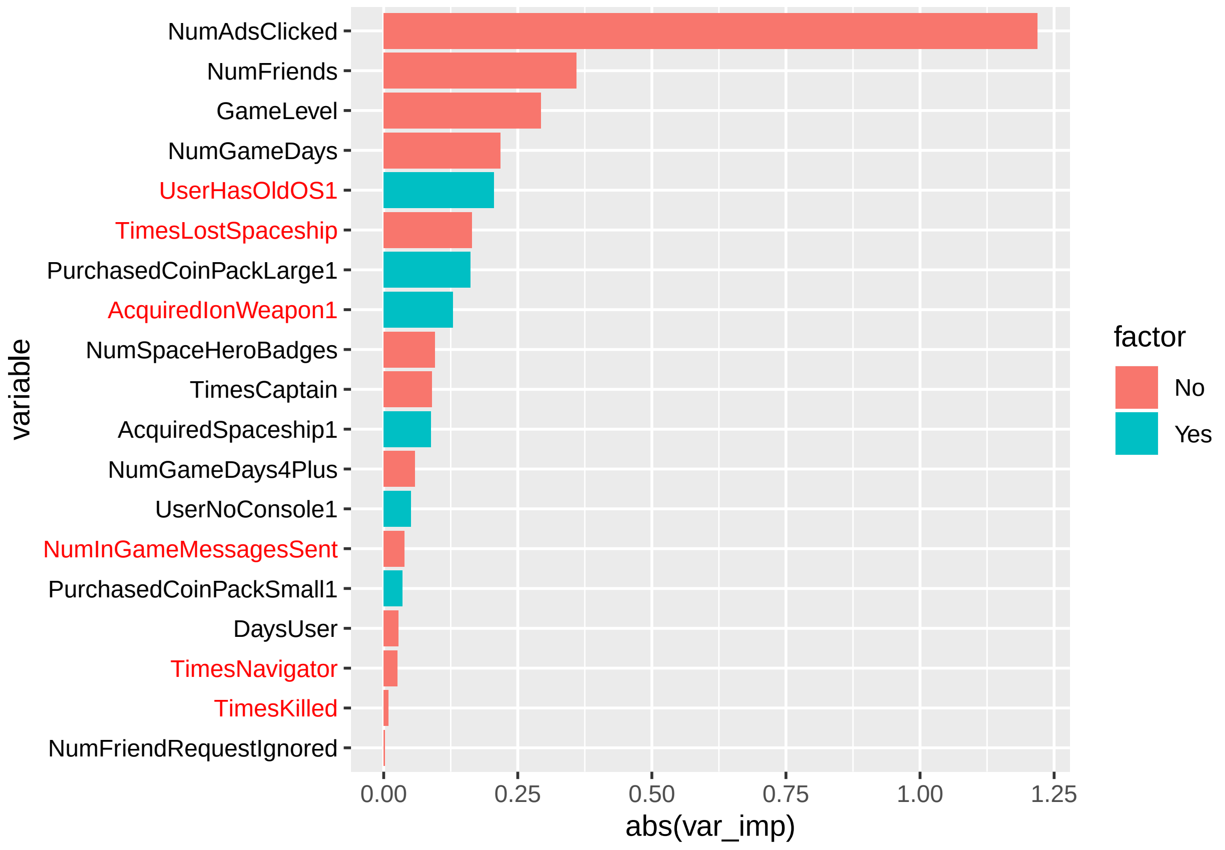
4. Compare the variable importance plot of the "organic" model and the "ad" model to explain why the performance of the models differ.

```
varimp.logistic(lrm) %>% plotimp.logistic()
```



```
# A tibble: 19 x 6
   variable                factor  var_imp var_imp_lower var_imp_upper p_value
   <chr>                   <chr>     <dbl>         <dbl>         <dbl>   <dbl>
 1 NumSpaceHeroBadges      No        1.28          1.20          1.36        0
 2 TimesLostSpaceship      No       -0.910        -1.14         -0.676       0
 3 NumFriendRequestIgnored No       -0.805        -0.976        -0.633       0
 4 GameLevel               No        0.577         0.424         0.730       0
 5 AcquiredSpaceship1      Yes       0.538         0.397         0.679       0
 6 AcquiredIonWeapon1       Yes     0.478         0.0798        0.876      0.019
 7 NumGameDays              No      0.469         0.318         0.621      0
 8 NumAdsClicked            No      0.398         0.289         0.506      0
 9 TimesNavigator           No     -0.393        -0.606        -0.179      0
10 NumInGameMessagesSent    No      0.313         0.158         0.467      0
11 NumGameDays4Plus         No      0.258         0.149         0.366      0
12 NumFriends               No      0.242         0.129         0.355      0
13 PurchasedCoinPackLarge1 Yes      0.239         0.0952        0.383      0.001
14 UserNoConsole1           Yes    -0.185        -0.364        -0.00592    0.043
15 UserHasOldOS1            Yes    -0.166        -0.410         0.0769     0.18
16 TimesKilled              No     -0.0964       -0.297         0.105      0.347
17 PurchasedCoinPackSmall1 Yes     -0.0924       -0.230         0.0452     0.188
18 TimesCaptain             No     -0.0565       -0.183         0.0703     0.383
19 DaysUser                 No     -0.00907      -0.141         0.123      0.893
```

```
varimp.logistic(lr_ad_random) %>% plotimp.logistic()
```



```
# A tibble: 19 x 6
   variable                factor  var_imp var_imp_lower var_imp_upper p_value
   <chr>                   <chr>     <dbl>         <dbl>         <dbl>   <dbl>
 1 NumAdsClicked            No      1.22          1.16          1.28      0
 2 NumFriends               No      0.359         0.295         0.423     0
 3 GameLevel                No      0.293         0.212         0.375     0
 4 NumGameDays              No      0.217         0.135         0.300     0
 5 UserHasOldOS1            Yes    -0.206        -0.339        -0.0723    0.003
 6 TimesLostSpaceship       No     -0.164        -0.267        -0.0616    0.002
 7 PurchasedCoinPackLarge1 Yes      0.162         0.0822        0.242     0
 8 AcquiredIonWeapon1       Yes    -0.129        -0.400         0.143     0.353
 9 NumSpaceHeroBadges      No      0.0960        0.0263         0.166    0.007
10 TimesCaptain            No      0.0896        0.0307         0.148    0.003
11 AcquiredSpaceship1      Yes     0.0878        0.00751        0.168    0.032
12 NumGameDays4Plus        No      0.0587       -0.0124         0.130    0.106
13 UserNoConsole1          Yes     0.0504       -0.0455         0.146    0.303
14 NumInGameMessagesSent   No     -0.0385       -0.128          0.0508   0.398
15 PurchasedCoinPackSmall1 Yes     0.0352       -0.0395         0.110    0.356
16 DaysUser                No      0.0274       -0.0453         0.100    0.46
17 TimesNavigator          No     -0.0260       -0.0906         0.0385   0.429
18 TimesKilled             No     -0.00912      -0.0832         0.0650   0.809
19 NumFriendRequestIgnored No      0.00235      -0.0862         0.0909   0.958
```

```
"
It appears the models differ because the original buyers of the expansion pack included
power users. The introduction of the ad changes the model to include new users regardless
of in-app telemtry data.
"
```

```
[1] "\nIt appears the models differ because the original buyers of the expansion pack included\npower u
```

Part 5: Better Models, Better Predictions

Mi Haruki had enjoyed by how much the model improved when it was retrained on the ad treatment instead of organic data. However, she knew that the analytics team was not done yet: “I know we have been trying to keep the modeling simple. Perhaps the logistic regression is what we go with. However, I want to explore using some machine learning models to see whether they improve our predictions.”

Hints: - To use a neural network we use the nnet package (please install it first and then load it in your R notebook). Please see the in-class handout “Using Neural Networks in Customer Analytics,” (Class8a_NN_demo.pdf) for how to run a neural network.

1. Train a neural network on the sample of customers who were exposed to the ad campaign. (If you time you can also try a random forest!).

```r
set.seed(1234)
cg_ad_random3 <- cg_ad_treatment[sample_random_30000,]
nn_cg <- nnet(converted ~ ., data=cg_ad_scoring, size=5, decay=0.1, maxit=1000)
```

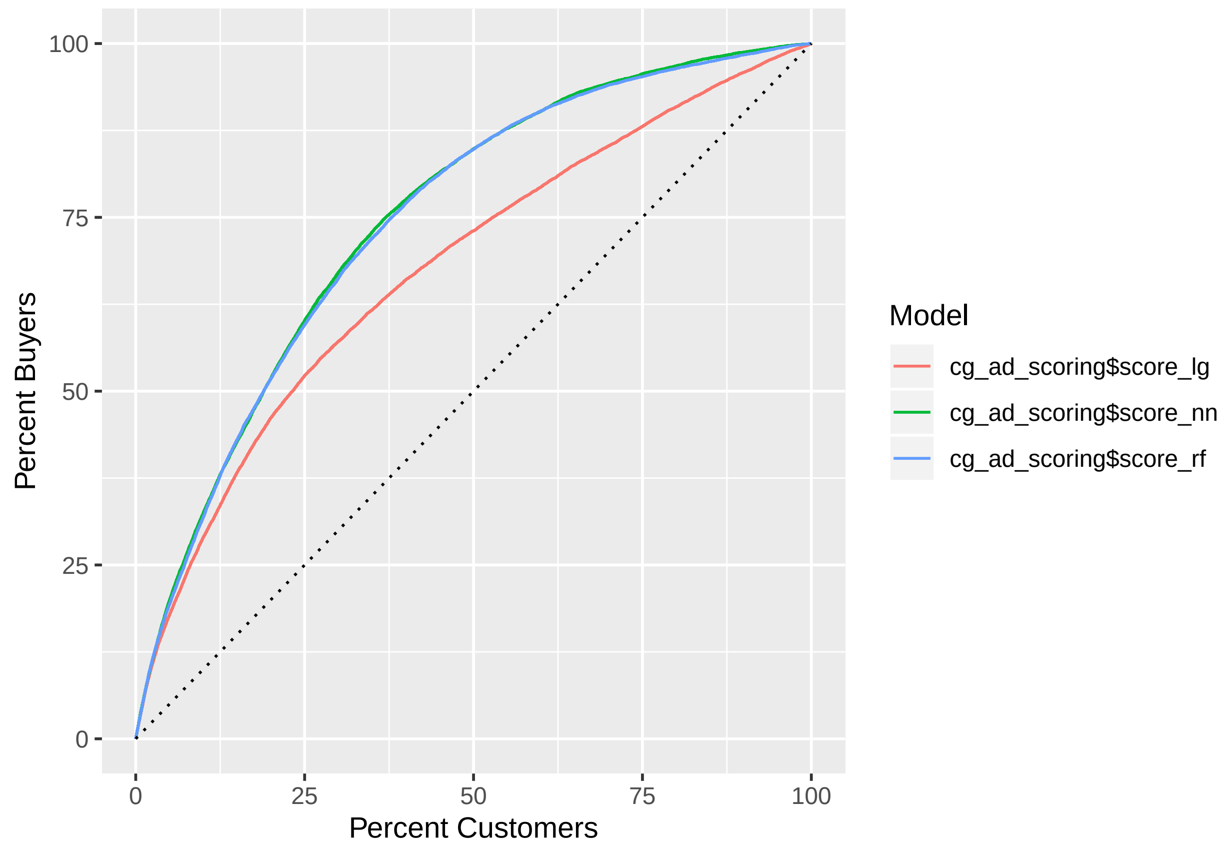
```
# weights:  116
initial  value 44917.841570
iter  10 value 16819.090724
iter  20 value 13699.478148
iter  30 value 13670.579037
iter  40 value 13615.863951
iter  50 value 13572.643694
iter  60 value 13045.075096
iter  70 value 12869.185748
iter  80 value 12609.937397
iter  90 value 12560.165485
iter 100 value 12513.516735
iter 110 value 12418.388942
iter 120 value 12365.404429
iter 130 value 12345.536051
iter 140 value 12321.174055
iter 150 value 12309.934606
iter 160 value 12278.759550
iter 170 value 12250.075142
iter 180 value 12231.084038
iter 190 value 12207.787317
iter 200 value 12182.507279
iter 210 value 12161.997534
iter 220 value 12129.987705
iter 230 value 12067.946062
iter 240 value 11995.045111
iter 250 value 11912.998517
iter 260 value 11838.625616
iter 270 value 11802.605388
iter 280 value 11784.569673
iter 290 value 11779.853463
iter 300 value 11777.706996
iter 310 value 11776.077938
iter 320 value 11765.658459
iter 330 value 11752.820670
iter 340 value 11749.537452
iter 350 value 11737.072339
iter 360 value 11716.225943
iter 370 value 11710.686531
iter 380 value 11708.653237
iter 390 value 11706.236180
iter 400 value 11705.466570
iter 410 value 11704.488946
iter 410 value 11704.488930
final  value 11704.488930
converged
```

```r
rf_cg <- ranger(converted ~ ., data=cg_ad_random3, probability=TRUE, mtry=3, min.node.size=1)
lr_cg <- glm(converted ~ ., family=binomial(logit), data=cg_ad_random3)

cg_ad_scoring <- cg_ad_scoring %>%
   mutate(score_nn = predict(nn_cg, type="raw"),
          score_rf = predict(rf_cg, data=cg_ad_scoring, type="response")[[1]][,2],
          score_lg = predict(lr_cg, newdata=cg_ad_scoring, type="response"))
```

2. Compare the performance of the neural network "ad" model and the logistic "ad" model from Part 4 on the "cg_ad_scoring" sample. Use gains curves and AUC to make the comparison. What do you find?

```r
gp <- gainsplot(
   cg_ad_scoring$score_nn,
   cg_ad_scoring$score_rf,
   cg_ad_scoring$score_lg,
   label.var=cg_ad_scoring$converted)
```



```
gp$auc
```

```
[1] 0.786 0.782 0.703
```

3. Calculate the profit improvement of using a neural network instead of a logistic regression (both trained on ad treatment data) to target the best 30,000 customers in the "cg_ad_scoring" sample.

```
quartileNN <- cg_ad_scoring %>%
   mutate(pred_buyer_quartile_nn = ntile(-score_nn, 4)) %>%
   group_by(pred_buyer_quartile_nn) %>%
   summarise(num_cust=n(), num_buyers=sum(converted), resp_rate=sum(converted)/n(), pred_resp_rate=mean

# TODO: This has to be wrong... Resp Rates are way higher than Predicted Response Rate
quartileNN
```

```
# A tibble: 4 x 5
  pred_buyer_quartile_nn num_cust num_buyers resp_rate pred_resp_rate
                   <int>    <int>      <int>     <dbl>          <dbl>
1                      1    30000       9462    0.315          0.0840
2                      2    30000       3889    0.130          0.0668
3                      3    30000       1698    0.0566         0.130
4                      4    30000        686    0.0229         0.0404
```

```
profitByQuartileNN <- quartileNN[1,] %>%
   mutate(revenue=num_buyers * revenue_per, cost=num_buyers * cost_per, profit=revenue-cost) %>%
   select(pred_buyer_quartile_nn, num_cust, num_buyers, resp_rate, revenue, cost, profit)

paste("Orig Profit: ", dollar(profitByQuartileNew$profit),
      "New Profit: ",  dollar(profitByQuartileNN$profit),
```

```r
      "Improvement: ", dollar(profitByQuartileNN$profit -profitByQuartileNew$profit ))
```

```
[1] "Orig Profit:  $110,847 New Profit:  $127,642 Improvement:  $16,795.05"
```