Cameron's Report

Cameron Lucas

3/10/2023

Loading Data, Libraries, and Functions

Initial EDA

I start by replacing all "null"s with NAs. Next, I removed the variables that don't make sense to use, such as name – which is completely random, address – which should be unique, TAS_ID – which is all the same value, etc.

```
sub = adv
for (i in 1:ncol(adv)) {
  if (is.character(adv[, i]))
    sub[, i] = nullToNA(adv[, i])
}

subRelRmv = sub %>% select(
  -c(
    "Not_in_use",
    "TAS_ID",
    "UniqueID",
    "SubmitDate",
    "Source",
    "_rescued_data",
    "HomeAddressStreet1",
    "HomeAddressStreet2",
    "HomeAddressSystemID",
    "FirstName",
    "MiddleName",
    "LastName",
    "HomeCity",
    "HomePostCode",
    "n_tran_credit_card_purchase"
  )
)
```

Here I modify the types of the classes from character to their correct type, either numeric, factor, or character.

```
nNames = names(subRelRmv %>% select(starts_with("n")))
catNames = names(subRelRmv %>% select(starts_with("cat")))
valNames = names(subRelRmv %>% select(starts_with("val") &
                                        !ends_with("map")))
mapNames = names(subRelRmv %>% select(ends_with("map")))
```

```
amtNames = names(subRelRmv %>% select(starts_with("amt")))
indNames = names(subRelRmv %>% select(starts_with("ind")))
pctNames = names(subRelRmv %>% select(starts_with("p")))
decileNames = names(subRelRmv %>% select(ends_with("decile")))


subRelRmv = subRelRmv %>% mutate_at(c(nNames, valNames, amtNames, pctNames),
                                    as.numeric)
subRelRmv = subRelRmv %>% mutate_at(c(catNames, mapNames, indNames, decileNames,
                                      "HomeState"),
                                    as.factor)

# Not included in the original variables
subRelRmv$FY_DONOR_GIFT_FIRST = as.numeric(subRelRmv$FY_DONOR_GIFT_FIRST)
```

We each took approximately a third of the variables to look at individually. I took the final variables after 96.

```
# ------------------------------------------------------------------------
# Looking at subset of variables for distribution and ideas of use
myCut = subRelRmv[97:ncol(subRelRmv)]
```

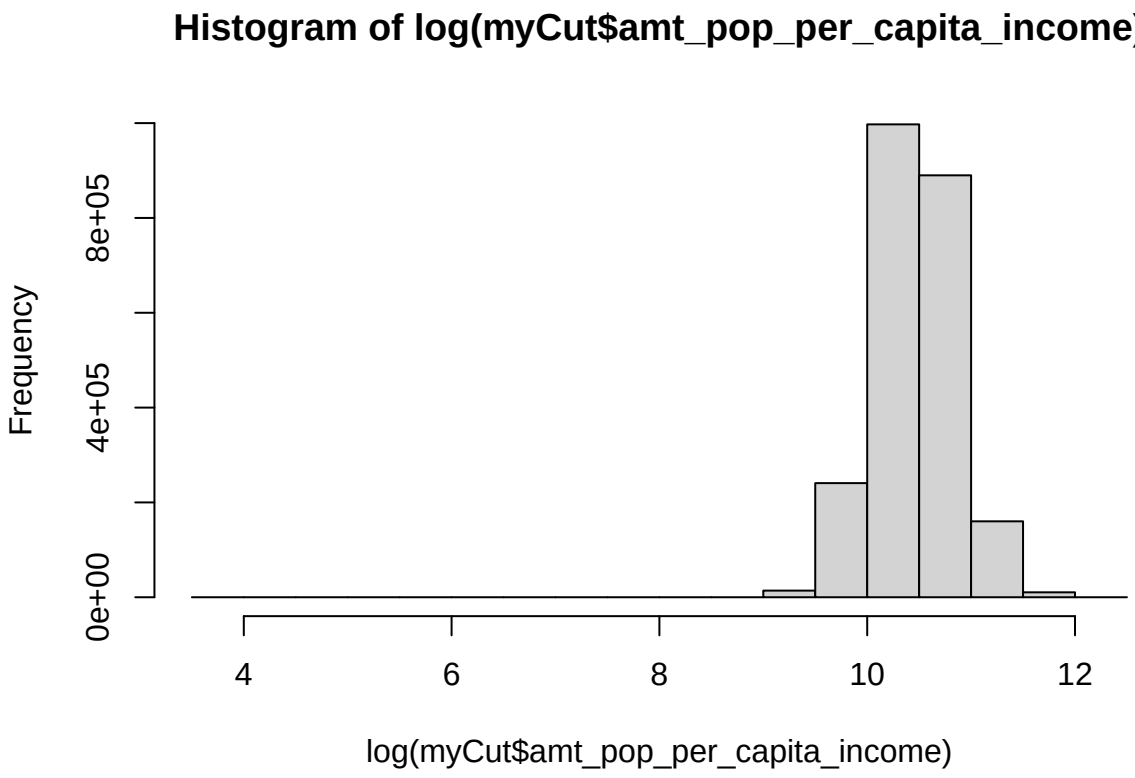
Checking Distributions

Next I looked through the distributions of every variable, noting variables that needed to be transformed, as well as checking for the percentage of observations that are NA. I will omit the code/output except for one example line for the page limit, because it is really the same line over and over ~50 times plus ~50 graphs.

```
sum(is.na(myCut$amt_pop_per_capita_income))/nrow(myCut) # 0.1% NAs!
```

```
## [1] 0.001280007
```

```
hist(log(myCut$amt_pop_per_capita_income)) # wants log transform
```




```
# Probably a good index of general income -- could be correlated with location
```

I am going to remove the next set of variables because they are all duplicates with variables that either are also in the set or that my group members have.

```
myCut2 = myCut %>% select(
  -c(
    "cat_score_donor_persona_map",
    "val_score_direct_marketing_score_map",
    "val_score_telemarketing_score_map",
    "val_score_online_score_map",
    "val_score_sustainer_score_map",
    "val_score_giving_tuesday_score_map",
    "val_score_end_of_year_score_map",
    "val_score_p2p_event_score_map",
    "val_score_p2p_diy_score_map",
    "cat_score_p2p_persona",
    "cat_demo_gender_map",
    "cat_demo_marital_status_map",
    "cat_demo_person_type_map",
    "cat_demo_dwelling_size_map",
    "cat_financial_mortgage_remainder_amount_map",
    "cat_financial_estimated_income_range_map",
    "cat_demo_occupation_map",
    "cat_demo_education_map",
```

```
    "cat_calc_political_persona_map",
    "cat_calc_social_score_map",
    "cat_ta_total_identified_assets_map",
    "cat_ta_wealth_segments_map",
    "cat_demo_dual_income_map",
    "val_score_philanthropic_score_map"
  )
)
```

Checking Correlations

```
corr_simple(myCut2, 0.8)
```

```
##                                    Var1                                Var2
## 301      val_score_philanthropic_score        val_score_end_of_year_score
## 66  val_pop_family_income_state_decile  val_pop_family_income_cbsa_decile
## 110     val_pop_home_value_state_index      val_pop_home_value_cbsa_index
## 22           amt_pop_per_capita_income               val_pop_ispsa_index
## 85           amt_pop_per_capita_income     val_pop_home_value_state_index
## 44                 val_pop_ispsa_index val_pop_family_income_state_decile
## 109  val_pop_family_income_cbsa_decile      val_pop_home_value_cbsa_index
##          Freq
## 301 0.9455868
## 66  0.9040495
## 110 0.8812910
## 22  0.8682611
## 85  0.8527969
## 44  0.8488959
## 109 0.8014389
```

```
# Philanthropic score correlated highly with end of year score and p2p event
# score.
# Going to get rid the other two, with the understanding that they are different
# and if phil score ends up being good, we can look deeper into those

myCut2 = myCut2 %>% select(-c("val_score_end_of_year_score",
                              "val_score_p2p_event_score"))

# Per capita income, ispsa index, home value cbsa/state, and income cbsa/state
# decile are in a fully connected correlation graph. We should only keep one,
# and we're going to choose family income cbsa decile. We think this is best
# because it is the most direct measure of wealth, is comparative to where they
# live, and it is a real value as opposed to an estimated/predicted value.

myCut2 = myCut2 %>% select(-c("val_pop_family_income_state_decile",
"val_pop_ispsa_index", "val_pop_home_value_cbsa_index",
"val_pop_home_value_state_index", "amt_pop_per_capita_income"))

corr_simple(myCut2, 0.8)
```

```
## [1] Var1 Var2 Freq
## <0 rows> (or 0-length row.names)
```

```
corr_simple(myCut2, 0.7)
```

```
##                               Var1                             Var2       Freq
## 120     val_score_sustainer_score val_score_giving_tuesday_score  0.7682351
## 118 val_score_telemarketing_score val_score_giving_tuesday_score  0.7618559
## 87        cat_score_donor_persona         val_score_online_score -0.7611372
## 104 val_score_telemarketing_score      val_score_sustainer_score  0.7373731
```

```
# We now see no more correlation at the 0.8 level, but we see a couple at the
# 0.7 level. We think it is fair to also remove giving Tuesday score since it
# logically should be very similar to the other many scores and is only one day,
# but the correlation between donor persona and online score is actually
# interesting; we won't remove that because we might look into why that occurs.

myCut2 = myCut2 %>% select(-"val_score_giving_tuesday_score")
```

Combining Our Sections Back Together

Now I am going to combine each of our datasets to get one standard one to use. This will reflect my work, so I will exclude some variables that my partners started with but had removed by the time I received their variables.

```
# Putting things together
# ------------------------------------------------------------------------------
# Chris Data

chrisNames = c(
  "val_donor_education_charities",
  "val_donor_private_foundation",
  "amt_ta_income",
  "amt_ta_discretionaryspending",
  "amt_ta_discretionaryspending_philanthropy",
  "amt_ta_networth",
  "amt_ta_investments_savings",
  "amt_ta_investments_savings_bonds",
  "pct_pop_some_college" ,
  "pct_pop_associate_degree",
  "pct_pop_bachelor_degree",
  "pct_pop_graduate_degree",
  "pct_pop_in_preschool",
  "Ppct_pop_in_elementary",
  "pct_pop_in_high_school",
  "pct_pop_in_college",
  "pct_pop_in_grad_school",
  "pct_pop_in_public_school",
  "pct_pop_in_private_school",
  "amt_tran_total_dollars_purchase",
  "amt_tran_avg_dollar_purchase",
  "cat_demo_political_affiliation",
  "cat_calc_social_score",
  "amt_financial_assessed_home_value",
  "n_demo_length_of_residence",
```

```
  "amt_financial_estimated_monthly_mortgage"
)

# Not including networth, it will need to be dealt with separately
logFix = c(
  "amt_ta_income",
  "amt_ta_discretionaryspending",
  "amt_ta_discretionaryspending_philanthropy",
  "pct_pop_in_college",
  "pct_pop_in_grad_school",
  "pct_pop_in_private_school",
  "amt_tran_total_dollars_purchase",
  "amt_tran_avg_dollar_purchase",
  "amt_financial_assessed_home_value",
  "n_demo_length_of_residence",
  "amt_financial_estimated_monthly_mortgage"
)

chrisCut = subRelRmv %>% select(all_of(chrisNames))

# Doing transformations

# Have to take out net worth for a sec because there are negatives
chrisCut = chrisCut %>% mutate(across(all_of(logFix), function(x)
  log(x + 1)))

# This has several hundred large amounts in the negatives, so I'm going to use
# z-scores and then do a log transform
worthMean = mean(chrisCut$amt_ta_networth, na.rm = T)
worthSd = sd(chrisCut$amt_ta_networth, na.rm = T)
#hist(log((chrisCut$amt_ta_networth - worthMean)/worthSd+1))
chrisCut$amt_ta_networth = log((chrisCut$amt_ta_networth - worthMean) /
                                 worthSd + 1)

# Additional change that needs to be done
chrisCut$amt_ta_investments_savings_bonds =
  as.factor(chrisCut$amt_ta_investments_savings_bonds)

logFix[12] = "amt_ta_income"
for (col in logFix) {
  col_index <- which(colnames(chrisCut) == col)
  colnames(chrisCut)[col_index] <- paste(col, "log", sep = "_")
}
# Chris data is transformed
# ------------------------------------------------------------------------------


# ------------------------------------------------------------------------------
# Irfan Data


irNamesNotToInclude = c(
  "cat_demo_date_of_birth",
```

```
  "cat_demo_marital_status",
  "cat_demo_person_type",
  "cat_financial_mortgage_remainder_amount",
  "cat_demo_occupation",
  "ind_lifestyle_cont_animal",
  "ind_lifestyle_cont_child_welfare",
  "ind_lifestyle_cont_conspoli",
  "ind_lifestyle_cont_culture",
  "ind_lifestyle_cont_environment",
  "ind_lifestyle_cont_health",
  "ind_lifestyle_cont_libpol",
  "ind_lifestyle_cont_political",
  "ind_lifestyle_cont_religion",
  "ind_lifestyle_cont_social_services",
  "ind_lifestyle_cause_volunteer",
  "n_demo_length_of_residence",
  "amt_financial_estimated_monthly_mortgage",
  "amt_financial_assessed_home_value"
)

irCut = subRelRmv[1:50] %>% select(-all_of(irNamesNotToInclude))

# I looked at the distributions here -- commenting out for space
# for (i in 1:ncol(irCut)) {
#   if (class(irCut[,i]) == "factor") {
#     barplot(table(irCut[,i]), main=paste('Feature', i))
#   } else {
#     hist(irCut[,i], main=paste('Feature', i))
#   }
# }

# I think we should also remove: dwelling size

# Indicator variables with x% of data in a single category (x >= 95). I feel like
# it's hard to make the case for including them.

# 98% money market, real estate, libpoli
# 97% for iras
# 96% priv comp, conspoli, mail response, multi buyer
# 95% humanitarian

# Looking at plots, they're not correlated much at all with ltg, nor are they
# distributed in an interesting way. I think we take them out, with the
# understanding that if some type of "assets" or politics variable is important,
# we could potentially use these to differentiate more.

irCut = irCut %>% select(
  -c(
    "cat_demo_dwelling_size",
    "ind_investments_money_market",
    "ind_investments_real_estate",
    "ind_lifestyle_cause_libpoli",
    "ind_investments_iras",
```

```
    "ind_demo_private_company_ownership",
    "ind_lifestyle_cause_conspoli",
    "ind_purchase_mail_responsive_buyer",
    "ind_purchase_dm_multi_buyer",
    "ind_lifestyle_cont_humanitarian"
  )
)

# Irfan data done
# ------------------------------------------------------------------------------



# ------------------------------------------------------------------------------
# Add together

subTrim = cbind(myCut2, chrisCut, irCut)

# We can see there are only 4 correlated above 0.9, that's not that bad actually.
# We're comfortable keeping both degree variables.
corr_simple(subTrim, 0.8)
```

```
##                                   Var1
## 1037    amt_ta_discretionaryspending_log
## 1159                      amt_ta_networth
## 2013 amt_tran_total_dollars_purchase_log
## 1464             pct_pop_bachelor_degree
##                                                Var2      Freq
## 1037 amt_ta_discretionaryspending_philanthropy_log 0.9940208
## 1159                    amt_ta_investments_savings 0.9194206
## 2013             amt_tran_avg_dollar_purchase_log 0.9005213
## 1464                      pct_pop_graduate_degree 0.8373588
```

```
subTrim = subTrim %>% select(
  -c(
    "amt_ta_discretionaryspending_log",
    "amt_ta_investments_savings",
    "amt_tran_avg_dollar_purchase_log"
  )
)

# Combining Done
# ------------------------------------------------------------------------------
```

Clustering

K-Medoids With Gower Distance and PAM

We are only able to do this with a subset of the data because of hardware ability. Based on our results though, we don't feel like this is something worth pursuing.

```r
# ------------------------------------------------------------------------------

tinyTest = subTrim[1:10000,]
gowerDist = daisy(tinyTest, metric='gower')

gower_mat <- as.matrix(gowerDist)

# Output most similar pair -- Commented for space

# tinyTest[
#   which(gower_mat == min(gower_mat[gower_mat != min(gower_mat)]),
#         arr.ind = TRUE)[1, ], ]

# Output most dissimilar pair -- Commented for space

# tinyTest[
#   which(gower_mat == max(gower_mat[gower_mat != min(gower_mat)]),
#         arr.ind = TRUE)[1, ], ]

# I'm commenting out the following code where I test multiple cluster numbers to see which performs the

# # Calculate silhouette width for many k using PAM
#
# sil_width <- c(NA)
#
# for(i in 2:10){
#
#   pam_fit <- pam(gowerDist,
#                  diss = TRUE,
#                  k = i)
#
#   sil_width[i] <- pam_fit$silinfo$avg.width
#
# }
#
# # Plot silhouette width (higher is better)
#
# plot(1:10, sil_width,
#      xlab = "Number of clusters",
#      ylab = "Silhouette Width")
# lines(1:10, sil_width)

# 2 clusters is the clear winner

pam_fit <- pam(gowerDist, diss = TRUE, k = 2)

# Could be interesting, but removed for space. Similar but more in depth look is coming
# pam_results <- tinyTest %>%
#   mutate(cluster = pam_fit$clustering) %>%
#   group_by(cluster) %>%
#   do(the_summary = summary(.))
#
# pam_results$the_summary
```

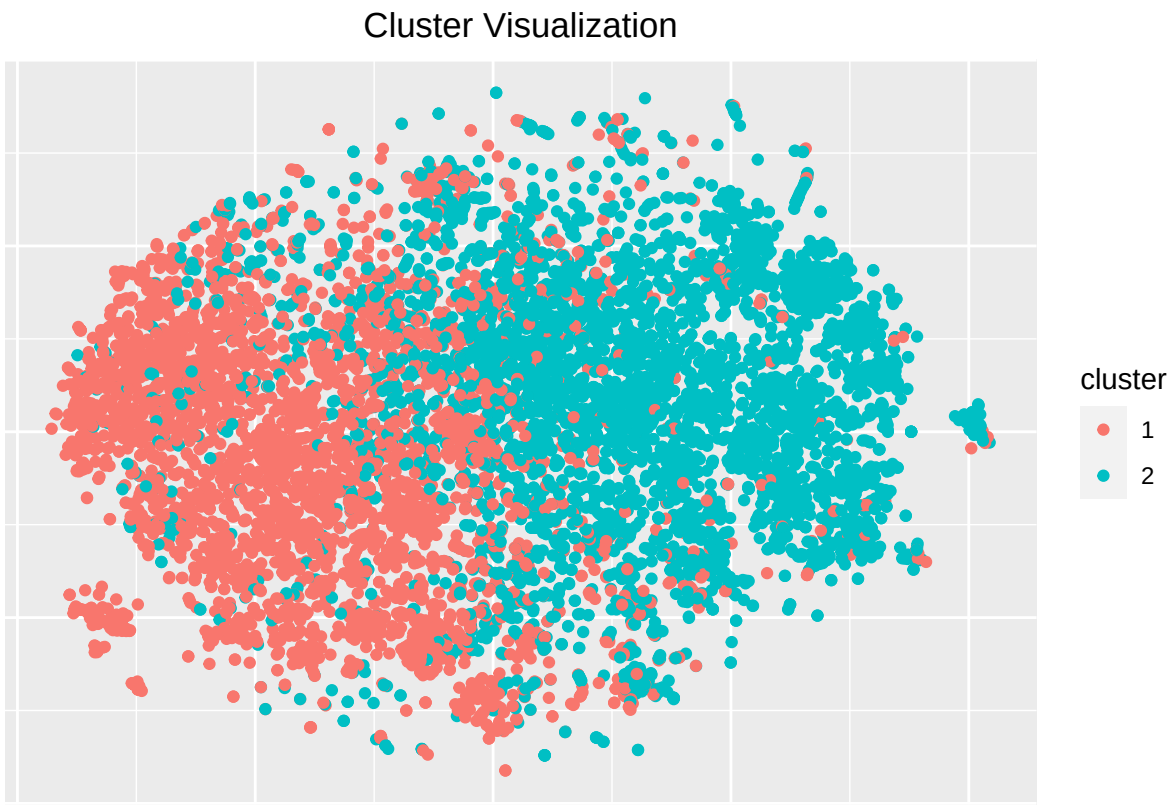
```
# We can see that there are large differences between the two clusters
# tinyTest[pam_fit$medoids, ] --- Commented out for space
```

Visualization With T-SNE

```
tsne_obj <- Rtsne(gowerDist, is_distance = TRUE)

tsne_data <- tsne_obj$Y %>%
  data.frame() %>%
  setNames(c("X", "Y")) %>%
  mutate(cluster = factor(pam_fit$clustering))

ggplot(aes(x = X, y = Y), data = tsne_data) +
  geom_point(aes(color = cluster)) + theme(axis.ticks.x=element_blank(),
                                           axis.text.x=element_blank(),
                                           axis.ticks.y=element_blank(),
                                           axis.text.y=element_blank(),
                                       plot.title=element_text(hjust=0.5)) +
  ylab("") + xlab("") + ggtitle("Cluster Visualization")
```



Cluster Analysis

Now I will manually look through the clusters and compare numeric medians/categorical modes to each other, to the min/max, and to the overall median/mode to find differences.

```
clusters = tinyTest %>%
  mutate(cluster = pam_fit$clustering)

cluster1 = (clusters %>% filter(cluster==1) %>% select_if(is.numeric) %>%
              apply(2, median, na.rm=T))[-37]
cluster2 = (clusters %>% filter(cluster==2) %>% select_if(is.numeric) %>%
              apply(2, median, na.rm=T))[-37]
med = (clusters %>% select_if(is.numeric) %>% apply(2, median, na.rm=T))[-37]

maxMin = t(data.frame(Max=apply(clusters %>% filter(cluster==1) %>%
                                  select_if(is.numeric),
      2, max, na.rm=T), Min=apply(clusters %>% filter(cluster==1) %>%
      select_if(is.numeric), 2, min, na.rm=T)))[,-37]

forChart = as.data.frame(rbind(maxMin, Cluster1=cluster1,
                               Cluster2=cluster2, Median = med))

prcDif = abs(forChart[3,] - forChart[4,]) / ((forChart[3,] + forChart[4,])/2) * 100

# Commenting out because it opens weird apps when I knit
# View(rbind(percent_difference = prcDif, forChart[5,],cluster1 = forChart[3,],
#            cluster2 = forChart[4,],
#            max = t(data.frame(maxMin[1,])), min = t(data.frame(maxMin[2,]))))

# Categorical

getmode <- function(v) {
  uniqv <- unique(v)
  if (is.na(uniqv[1])) {
    uniqv = uniqv[-1]
  }
  uniqv[which.max(tabulate(match(v, uniqv)))]
}

cluster1Cat = (clusters %>% filter(cluster==1) %>% select_if(is.factor) %>%
                 apply(2, getmode))
cluster2Cat = (clusters %>% filter(cluster==2) %>% select_if(is.factor) %>%
                 apply(2, getmode))
mode = (clusters %>% select_if(is.factor) %>% apply(2, getmode))


forChartCat = as.data.frame(rbind(Cluster1=cluster1Cat,
                               Cluster2=cluster2Cat, Mode = mode))

# Commenting out because it opens weird apps when I knit
# View(rbind(cluster1 = forChartCat[1,],
#            cluster2 = forChartCat[2,],
#            mode=forChartCat[3,]))
```

When we look at these clusters, there are some positives and negatives. The positives are that we achieved some fairly clear splits with fairly even clusters (4500 in 1 and 5500 in 2). It's obvious that cluster 1 tends to be less educated, less wealthy, and less likely to donate, while cluster 2 is the opposite. The negatives are that this doesn't segment our population much — we already expected that more educated/wealthy people would donate more.

However, our metrics give us 2 clusters as the ideal number. This is likely because the data just isn't distributed well to make nice clusters, so more clusters just makes more overlap/noise within clusters. We should zoom out.

Chris has done interesting work with random forests and we've reached a new goal: leverage Blackbaud's clusters to try to figure out what variables they used, rather than creating our own clusters from scratch.

I wanted to work on this as well, and I started very similarly to Chris, by trying our main, cleaned dataset in a random forest with the p2p personas as a response variable.

Random Forest

Forest 1 — Fully Cleaned Dataset

First I have to clean the data a little more to make random forest work:

```
levels(subTrim$HomeState)[levels(subTrim$HomeState) %in% c("AA", "AE", "AP")] =
  "Armed Forces"
levels(subTrim$HomeState)[levels(subTrim$HomeState) %in%
                            c("PR", "GU", "MP", "VI")] = "Territory"
levels(subTrim$cat_score_p2p_persona_map)[levels(subTrim$cat_score_p2p_persona_map)
                                          == "Average Joes"] = "9 Average Joes"

subTrim$HomeState = droplevels(subTrim$HomeState, exclude = "AB")


# Also going to remove the other persona var because there's no point
subTrimRF = subTrim %>% select(-"cat_score_donor_persona")

# To use random forest, we also can only use complete observations. We still have
# over half a million observations though.
subTrimRF = na.omit(subTrimRF)
```

Modeling

```
################################################################################
#RF MODELING

rf <- randomForest(cat_score_p2p_persona_map ~ .,
                   data=subTrimRF, importance=T,
                   ntree=30, maxnodes=50)

rf$err.rate[30,]
```

```
##                   OOB          1 Go Getters 2 Caring Contributors
##             0.5748244             0.6828401             0.3319883
## 3 Casual Contributors          4 Do Gooders       5 Generous Joes
##             0.6539625             0.9791023             0.6277522
##      6 Over Achievers   7 Cause Enthusiasts      8 Thrill Seekers
##             0.4340194             0.9696176             0.2455008
##        9 Average Joes
##             0.9825052
```

```
# We can see that predictions are pretty good for some categories (2 and 8),
# decent for some (1, 5, 6, and 7, as well as overall), while three categories
# had horrible error rates (4, 7, and 9)
```

It seems unlikely that Blackbaud would have some top secret data that completely predicts these very poor categories — more likely, some of the variables that we removed during data cleaning were important to distinguish these categories. We want to retry using random forests, but this time include all of the reasonable variables: under 50% NA. Other variables were removed either for multicolinearity or by intuition, but random forests shouldn't be susceptible to multicolinearity issues, and all we care about now is prediction accuracy.

Forest 2 — Expanded Data Set

```
badNames = names((sort(colSums(is.na(subRelRmv))/nrow(subRelRmv) >
                         0.5))[(sort(colSums(is.na(subRelRmv))/nrow(subRelRmv) > 0.5))])
subTrimRF2 = (subRelRmv %>% select(-all_of(badNames)))

# Too many levels
subTrimRF2 = subTrimRF2 %>% select(-"cat_calc_mosaic")

# Fix date of birth
subTrimRF2$cat_demo_date_of_birth = as.numeric(substr(
  as.numeric(as.character(subTrimRF2$cat_demo_date_of_birth)), 1, 4))
# Get rid of maps, they're duplicates
subTrimRF2 = subTrimRF2 %>% select(
  -c(
    "cat_score_donor_persona_map",
    "val_score_direct_marketing_score_map",
    "val_score_telemarketing_score_map",
    "val_score_online_score_map",
    "val_score_sustainer_score_map",
    "val_score_giving_tuesday_score_map",
    "val_score_end_of_year_score_map",
    "val_score_p2p_event_score_map",
    "val_score_p2p_diy_score_map",
    "cat_demo_gender_map",
    "cat_demo_marital_status_map",
    "cat_demo_person_type_map",
    "cat_demo_dwelling_size_map",
    "cat_financial_mortgage_remainder_amount_map",
    "cat_financial_estimated_income_range_map",
    "cat_demo_occupation_map",
    "cat_demo_education_map",
```

```
    "cat_calc_political_persona_map",
    "cat_ta_total_identified_assets_map",
    "cat_ta_wealth_segments_map",
    "val_score_philanthropic_score_map"
  )
)

levels(subTrimRF2$HomeState)[levels(subTrimRF2$HomeState) %in% c("AA", "AE", "AP")] =
  "Armed Forces"
levels(subTrimRF2$HomeState)[levels(subTrimRF2$HomeState) %in%
                               c("PR", "GU", "MP", "VI")] = "Territory"
levels(subTrimRF2$cat_score_p2p_persona_map)[levels(subTrimRF2$cat_score_p2p_persona_map)
                                          == "Average Joes"] = "9 Average Joes"

subTrimRF2$HomeState = droplevels(subTrimRF2$HomeState, exclude = "AB")


# Also going to remove the other persona var because there's no point and the
# non-map p2p persona because it will just predict everything perfectly
subTrimRF2 = subTrimRF2 %>% select(-"cat_score_donor_persona")
subTrimRF2 = subTrimRF2 %>% select(-"cat_score_p2p_persona")

subTrimRF2 = na.omit(subTrimRF2) # Still over 400,000 observations!

# Some further cleaning is done but not included, because it is a repeat of the log transformations fro
```

```
################################################################################
#RF MODELING 2

rf2 <- randomForest(cat_score_p2p_persona_map ~ .,
                    data=subTrimRF2, importance=T,
                    ntree=50)

rf2$err.rate[50,]
```

```
##                   OOB          1 Go Getters 2 Caring Contributors
##             0.3375637             0.3015864             0.2153877
## 3 Casual Contributors          4 Do Gooders       5 Generous Joes
##             0.3425329             0.4052257             0.3820683
##      6 Over Achievers   7 Cause Enthusiasts      8 Thrill Seekers
##             0.3311334             0.4752826             0.2311188
##        9 Average Joes
##             0.5407473
```

I would like to note that despite using a seed, the forest I get when knitting is different from the one when running for analysis. However, the end error is very similar, so I think it won't affect much.

We can see that the overall error is much improved (~24%), and we see huge improvements in the error rate of the three worst personas, 4, 7, and 9.

Here you can see the top 5 most important variables for each persona, as well as the 5 best for overall accuracy/impurity.

```
z2 = importance(rf2)
data.frame(Go_Getters_1 = names(head(sort(abs(z2[,1]), decreasing=T), 5)),
           Caring_Contributors_2 = names(head(sort(abs(z2[,2]), decreasing=T), 5)),
           Casual_Contributors_3 = names(head(sort(abs(z2[,3]), decreasing=T), 5)),
           Do_Gooders_4 = names(head(sort(abs(z2[,4]), decreasing=T), 5)),
           Generous_Joes_5 = names(head(sort(abs(z2[,5]), decreasing=T), 5)),
           Over_Achievers_6 = names(head(sort(abs(z2[,6]), decreasing=T), 5)),
           Cause_Enthusiasts_7 = names(head(sort(abs(z2[,7]), decreasing=T), 5)),
           Thrill_Seekers_8 = names(head(sort(abs(z2[,8]), decreasing=T), 5)),
           Average_Joes_9 = names(head(sort(abs(z2[,9]), decreasing=T), 5)),
           Overall_Accuracy = names(head(sort(abs(z2[,10]), decreasing=T), 5)),
           Overall_Gini = names(head(sort(abs(z2[,11]), decreasing=T), 5)))
```

```
##                              Go_Getters_1
## 1 cat_financial_mortgage_remainder_amount
## 2          n_demo_length_of_residence_log
## 3                                HomeState
## 4        val_score_direct_marketing_score
## 5                 val_score_p2p_diy_score
##                    Caring_Contributors_2
## 1 cat_financial_mortgage_remainder_amount
## 2                  pct_pop_in_college_log
## 3           pct_pop_in_private_school_log
## 4                pct_pop_in_public_school
## 5          n_demo_length_of_residence_log
##                           Casual_Contributors_3
## 1      cat_financial_mortgage_remainder_amount
## 2               n_demo_length_of_residence_log
## 3                                     HomeState
## 4                    pct_pop_in_college_log
## 5 amt_financial_estimated_available_equity_log
##                                    Do_Gooders_4
## 1        cat_financial_mortgage_remainder_amount
## 2                          Ppct_pop_in_elementary
## 3                          pct_pop_in_college_log
## 4 amt_ta_discretionaryspending_entertainment_log
## 5                       pct_pop_in_public_school
##                          Generous_Joes_5
## 1 cat_financial_mortgage_remainder_amount
## 2                  pct_pop_in_high_school
## 3                                HomeState
## 4                   pct_pop_in_preschool
## 5                  pct_pop_in_college_log
##                                      Over_Achievers_6
## 1              cat_financial_mortgage_remainder_amount
## 2                        n_demo_length_of_residence_log
## 3 amt_ta_discretionaryspending_internationaltravel_log
## 4                              pct_pop_in_high_school
## 5                           val_demo_number_of_adults
##                           Cause_Enthusiasts_7
## 1      cat_financial_mortgage_remainder_amount
## 2                                    HomeState
## 3                   cat_calc_political_persona
```

```
## 4              n_demo_length_of_residence_log
## 5 amt_financial_estimated_available_equity_log
##                                    Thrill_Seekers_8
## 1 amt_ta_discretionaryspending_internationaltravel_log
## 2                            val_score_p2p_diy_score
## 3        amt_financial_estimated_available_equity_log
## 4             cat_financial_mortgage_remainder_amount
## 5                           pct_pop_in_high_school
##                        Average_Joes_9                        Overall_Accuracy
## 1             val_score_p2p_diy_score cat_financial_mortgage_remainder_amount
## 2                           HomeState               val_demo_number_of_adults
## 3      n_demo_length_of_residence_log                pct_pop_in_public_school
## 4  amt_tran_total_dollars_purchase_log                    pct_pop_in_preschool
## 5 cat_financial_estimated_income_range          n_demo_length_of_residence_log
##                     Overall_Gini
## 1 val_score_giving_tuesday_score
## 2  val_score_telemarketing_score
## 3       val_score_p2p_diy_score
## 4     val_score_p2p_event_score
## 5        val_score_online_score
```

Analysis

Common Top Variables

All of the conclusions were drawn either from looking at graphs with the variable of interest as a response variable and the personas as the dependent variable (for numerical vars) or by filtering a single persona and looking at that against the variable of interest, and repeating for each persona (for categorical variables). As you may imagine, this resulted in many, many graphs. I will include one example of each type in the code for future use, but remove the rest for space.

```
################################################################################
# Looking at vars

# Variables to look at:

# For education in general, it follows as such: Personas 1-3 live in particularly
# educated areas; 4, 5, and 8 live in particularly uneducated areas; and 6, 7,
# and 9 are somewhat in the middle.

# For all score variables in general, 3 tends to underperform while 2 and 4
# overperform.

# For all money/wealth variables there is a general downwards trend, with the
# notable exception being that persona 2 is lower than 3.

# private school - Higher rates for category 1.

# number of adults in household - This has a median of 3 adults for every single
# persona except 1, which has a median of 4 adults in the household.

# length of residence - 7 and 9 have a shorter median length of residence, while
# 4 and 5 have noticeably longer residences.
```

```
# telemarketing - Particularly low for 7 and 9. Also, while it isn't very low
# for persona 1, it is uncharacteristically low. This is a weak spot for
# targeting them.

# homestate - Hard to find much with or even look at homestate. Our guess
# though is that it it just helps overall accuracy a lot because it has so
# many splits to offer at lower depths. But there probably isn't a clear
# pattern to find.

# age - Persona 5 has a median age about 3 years older than the overall median,
# while personas 7 and 9 are both 4 years younger than the overall median age.

# p2p diy events and online donation score - persona 4 scores particularly bad
# in these, despite overperforming in all of the others.



# Example of code used:
# boxplot(subTrimRF2$amt_financial_assessed_home_value~
#           subTrimRF2$cat_score_p2p_persona_map, outline=F,notch=T)
#
# barplot(table(subTrimRF2 %>% filter(cat_calc_political_persona == "03") %>%
# select(cat_score_p2p_persona_map))/table(subTrimRF2$cat_score_p2p_persona_map))
```

While most personas did not have politics as a top 5 important variable, multiple had it in their top 10s, and we thought it would be an interesting variable to analyze.

```
# Political tendancies by p2p category

# (x% higher) for y politics means that if the proportion of people in the dataset as a whole in politi

# Cat 1:
#   Are: On-the-fence-liberals (mean 10% higher), Mild republicans (8% higher)
#     super democrats (4% higher),
#   Not: Ultraconservative (13% lower), conservative democrats (6% lower)

# Cat 2:
#   Are: mild republicans  (9% higher)
#   Not: Ultraconservative (5% lower)

# Cat 3:
#   Are: Mild republicans (11% higher)
#   Not: Ultraconservatives (8% lower), conservative democrats (4% lower)

# Cat 4:
#   Are: Ultraconservatives (6% higher)

# Cat 5:
#   Are: Ultraconservatives (11% higher)
#   Not: Mild Republicans (8% lower)

# Cat 6: Completely even split
```

```
# Cat 7:
#   Not: Ultraconservatives (6% lower)

# Cat 8:
#   Are: Conservative democrats (6% higher) and ultraconservatives (10% higher)
#   Not: Mild Republicans (14% lower)

# Cat 9:
#   Are: Left out democrats (5% higher) and conservative democrats (5% higher)
#   Not: Mild republicans (10% lower)




# Example of code used:
# for (ind in c("1 Go Getters", "2 Caring Contributors", "3 Casual Contributors", "4 Do Gooders", 5 Gen
# "9 Average Joes")) {
#   print(paste("Table", ind))
#   print((table(subTrimRF2 %>% filter(cat_score_p2p_persona_map == ind) %>%
#         select(cat_calc_political_persona))/sum(table(subTrimRF2 %>%
#                                                 filter(cat_score_p2p_persona_map == ind) %>%
#                                                 select(cat_calc_political_persona))) -
#     table(subTrimRF2$cat_calc_political_persona) / nrow(subTrimRF2)) * 100)
# }
```

Using the full random forest, we see similar general trends as before. The most important variables overall are wealth, education, and the donation scores. However, we were able to much better divide the personas into four larger categories using these:

Personas 1, 2, and 3 — Highly educated, wealthy, and moderate in politics

Personas 4, 5, and 8 — Lower education and wealth, and conservative politics

Personas 7 and 9 — Average wealth, young, shorter length of residence, and respond poorly to donation messages, especially via telemarketing

Persona 6 — Average on all accounts

Moreover, we found ways to distinguish between the personas in each grouping if need be:

People with persona 1 attend private school in higher rates and have more adults in the household. They unfortunately respond poorly to telemarketing though, so they are best reached through other means.

Persona 2s oddly have less wealth than their persona 3 counterparts.

Persona 4s tend to have a longer length of residence. They have very high donation scores for their wealth, however not in p2p diy events or online donations. These people are worth targeting despite their wealth, but you really have to go out to get them – they won't come to you on their own.

People with persona 5 tend to be older and have longer lengths of residence.

Once again, persona 6s are your real average Joes.

People with persona 7 live in slightly higher educated areas, and while they don't fall under a single political ideology, they are not "superconservatives"

Persona 8s are all around unlikely to donate. Combined with their grouping of low wealth, these people should not be prioritized for donations.

People with persona 9 tend to live in slightly lower educated areas and are politically democrat.