DRIVERS OF DIGITAL ENGAGEMENT

PROJECT SUMMARY

For this project, I performed the initial step of a much bigger initiative to become more proactive in getting our clients (financial advisors) more digitally engaged. Digital engagement is important for two reasons: it helps us understand what all advisors, not only the ones reading our online contnent, are looking for in their preferred investment managers and it is an cost-efficient marketing avenue. For non-digitally-active advisors, we obviously have little to work with in terms of what content pillars have been of interest. Currently, only ~25% of advisors are reading our online content, which provides us a limited set of information. Once we better understand what drives digital engagement, we can apply user-user collaborative filtereing (identifying non-digitally engaged counterparts to the current digitally engaged advisors) to recommend specific content to specific advisors (e.g. Netflix recommendation engine). This is a component of an effort to deliver more robust information to our saleforce to further customize their approach to generating sales.

The first objective of this research was to better understand what drives digital engagement. The dependent variable here is a binary 0 or 1 determined by whether a given advisor visited our web materials in the 90 days leading up to June 2016. I collected 30 independent variables that I thought may contribute to a client's likliehood of utilizing our online sources. I will go over those in the *data deep dive* portion of the report. I then created a function that cleans the data by removing variables with low significance and high co-linearity. These will be expounded upon in the *methodology* section.

We made the following key assumptions:

1. The last 90 days of our advisors' web (we didn't include any email data) behavior is a good representation of their digital behavior in general.
2. The 160,000 advisors so represented in our data are a good proxy for the over 280,000 advisors in total.
3. Logistic regression is appropriate model to apply to advisor digital engagement.
4. Content pillars are exchangeable across all media (mail, email, website material, in-person topics, etc.), i.e. if we know one's preferred content pillar, we can meet their needs across a variety of touchpoints.

Given these assumptions, we hypothesized the following:

To in/validate using the model:

- We can attain a model with AUC greater than .75 using simple regression techniques.

To in/validate using WoE tables:

- Clients in the IBD space are significantly more likely to be digitally engaged than those in the FC space. In particular, Edward Jones clients will be more likely to be digitally engaged.
- Those with more portfolio diversity (i.e. asset types, account types, record-keeping types) are more likely to have higher engagement.
- The newer the advisor is to our products, the more likely they will be to be digitally engaged.

To in/validate using clustering:

- The level of digital engagement will show to be independent from content preference.

We will make these results actionable by directly targeting the non-digitally active advisors that fit the profile of digitally active advisors (i.e. the model outputs a 1).

RESULTS SUMMARY

We built a logistic model with 10 inputs that had an accuracy of .76 on our test data and an ROC AUC of .79.

The biggest drivers of digital engagement are:

1. size of client (transaction count (TXN CNT); assets largest share class in portfolio (SHARE CLASS 1 AUM); internal prospect denomination (RTL SLS TIER DESC))
2. portfolio diversity (proportion of the portfolio that is the largest account type (ACCT TYPE PSHARE); proportion of the portfolio that is the largest and second largest asset classes (ASSET CLASS 1/2 PSHARE))
3. client type (firm (TOP 7 FIRMS); largest asset class in porfolio (ASSET CLASS 1); largest share class in porfolio (SHARE CLASS 1); largest record-keeping type in porfolio (RK TYPE))

We can use the WoE tables to validate the first 2 of the 3 hypotheses in the WoE category (IBD/FC and portfolio diversity) and to invalidate the 3rd hypothesis in that category (we were wrong about newer advisors being more digitally active).

We successfully applied clustering to validate that content preferences and digital engagement are independent (i.e. if we look within the digitally engaged population, scale the data and segment the advisors by distance - we don't see much variance in content preference cluster to cluster). This suggests that user-user collaborative filtering will be effective in generating recommendations for non-digitally engaged advisors.

Inside look in WoE tables:

1. **Size of client:**

This is the summary WoE table for transaction count (TXN CNT). This is the biggest driver of digital engagement: those who transacted fewer than 2,000 times are significantly less likely to be digitally engaged. The same trend is seen in the SHARE CLASS 1 AUM table (not shown).

```
##      Cutpoint CntRec CntGood CntBad GoodRate BadRate     WoE     IV
## 1        <= 9  16451    1092  15359   0.0664  0.9336 -1.7315 0.1865
## 2       <= 28  17141    1635  15506   0.0954  0.9046 -1.3374 0.1303
## 3       <= 65  18260    2322  15938   0.1272  0.8728 -1.0141 0.0878
## 4      <= 126  17063    2687  14376   0.1575  0.8425 -0.7649 0.0501
## 5      <= 295  23606    5285  18321   0.2239  0.7761 -0.3310 0.0146
## 6      <= 555  16806    5114  11692   0.3043  0.6957  0.0853 0.0008
## 7     <= 2055  30010   13271  16739   0.4422  0.5578  0.6800 0.0945
## 8   <= 325491  25153   15729   9424   0.6253  0.3747  1.4244 0.3609
## 9     Missing      0       0      0       NA      NA      NA     NA
## 10      Total 164490   47135 117355   0.2866  0.7134  0.0000 0.9255
```

An interesting takeaway from the RTL SLS TIER DESC table is that those with the highly prospective (HP) designation have the lowest digital engagement rates. This validates the connection between digital engagement and prospective clients and should be where invest in digital targeting.

```
##                         Cutpoint CntGood CntBad GoodRate BadRate     IV
## 1                 = A - Priority   10346   9282   0.5271  0.4729 0.1433
## 2          = B - Medium Priority    8128  10389   0.4389  0.5611 0.0560
## 3                = C - Long Tail   23421  82186   0.2218  0.7782 0.0698
## 4 = HP - High Potential Prospects    3161  11576   0.2145  0.7855 0.0122
## 5                        = Other    2079   3922   0.3464  0.6536 0.0030
## 6                        Missing       0      0       NA      NA     NA
## 7                          Total   47135 117355   0.2866  0.7134 0.2843
```

2. **Portfolio diversity:**

This is the summary WoE table of the proportion of the portfolio that is the largest account type. Notice that those with only one account type representing over 55% percent of their entire portfolio have a bad rate of greater than 72%, meaning they're significantly less likely to be digitally engaged. This directly validates one of our hypotheses.

```
##      Cutpoint CntRec CntGood CntBad GoodRate BadRate     WoE     IV
## 1 <= 0.356117  23944   12025  11919   0.5022  0.4978  0.9210 0.1414
## 2  <= 0.41349  20147    7558  12589   0.3751  0.6249  0.4020 0.0213
## 3 <= 0.466927  21468    7098  14370   0.3306  0.6694  0.2069 0.0058
## 4 <= 0.556709  32573    8999  23574   0.2763  0.7237 -0.0508 0.0005
## 5 <= 0.650949  22186    5135  17051   0.2315  0.7685 -0.2879 0.0105
## 6 <= 0.869704  26652    4522  22130   0.1697  0.8303 -0.6758 0.0626
## 7        <= 1  17520    1798  15722   0.1026  0.8974 -1.2562 0.1204
## 8     Missing      0       0      0       NA      NA      NA     NA
## 9       Total 164490   47135 117355   0.2866  0.7134  0.0000 0.3625
```

3. **Client type:**

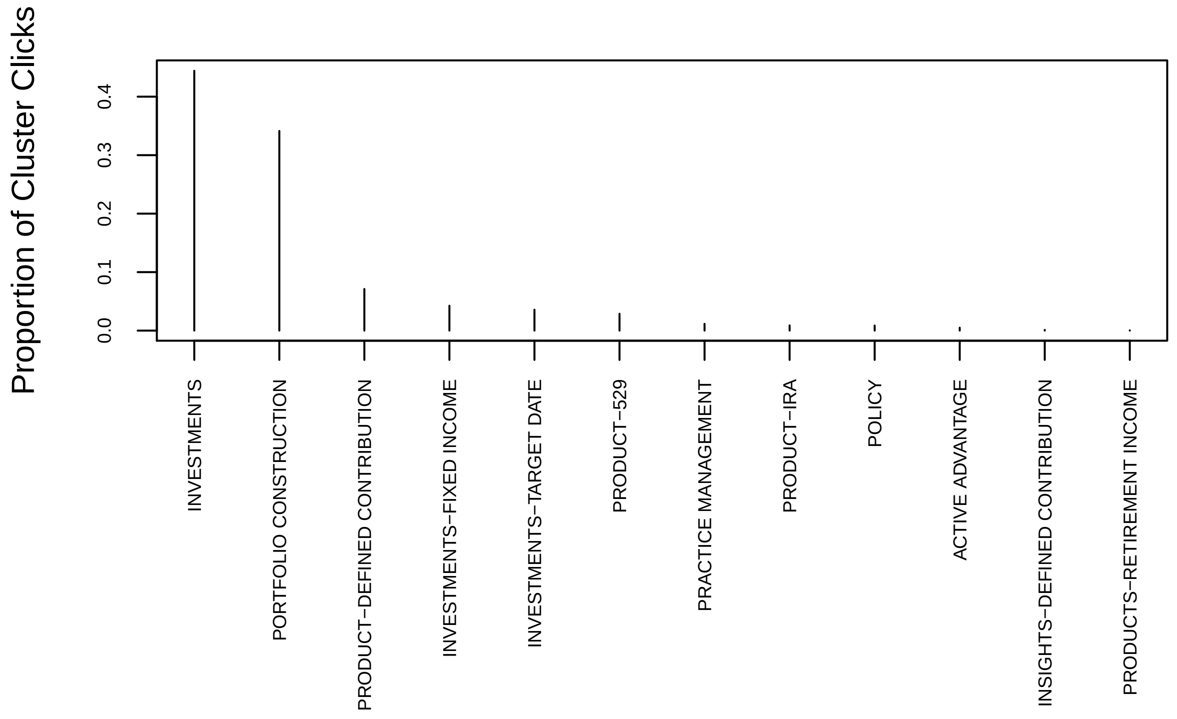
This is the summary WoE table of bad/good rates by either one of the top 7 firms in the industry or everyone else. The lowest good rates are clearly the wirehouses (Merrill Lynch, Morgan Stanley, UBS, Wells Fargo) and the highest good rate is Edward Jones. We hypothesized this would be the case given wirehouses' perceived approach to advising and Jones' strong relationship with our firm. Additional notable findings: the big users of our portfolio series had the highest asset class good rate of 39%; the big users of brokerage share classes (A,B,C) had the highest good rate of 32%; the biggest users of individual 403(b) record-keeping typs had the lowest good rates at <19%.

```
##            Cutpoint CntRec CntGood CntBad GoodRate BadRate     WoE     IV
## 1    = Edward Jones  13215    6150   7065   0.4654  0.5346  0.7735 0.0544
## 2             = LPL  10429    3685   6744   0.3533  0.6467  0.3078 0.0064
## 3   = Merrill Lynch   3025     122   2903   0.0403  0.9597 -2.2573 0.0500
## 4  = Morgan Stanley   9906     859   9047   0.0867  0.9133 -1.4422 0.0849
## 5           = Other 106073   32544  73529   0.3068  0.6932  0.0971 0.0062
## 6   = Raymond James   5788    1497   4291   0.2586  0.7414 -0.1409 0.0007
## 7             = UBS   4724     510   4214   0.1080  0.8920 -1.1996 0.0301
## 8     = Wells Fargo  11330    1768   9562   0.1560  0.8440 -0.7758 0.0341
## 9           Missing      0       0      0       NA      NA      NA     NA
## 10            Total 164490   47135 117355   0.2866  0.7134  0.0000 0.2668
```

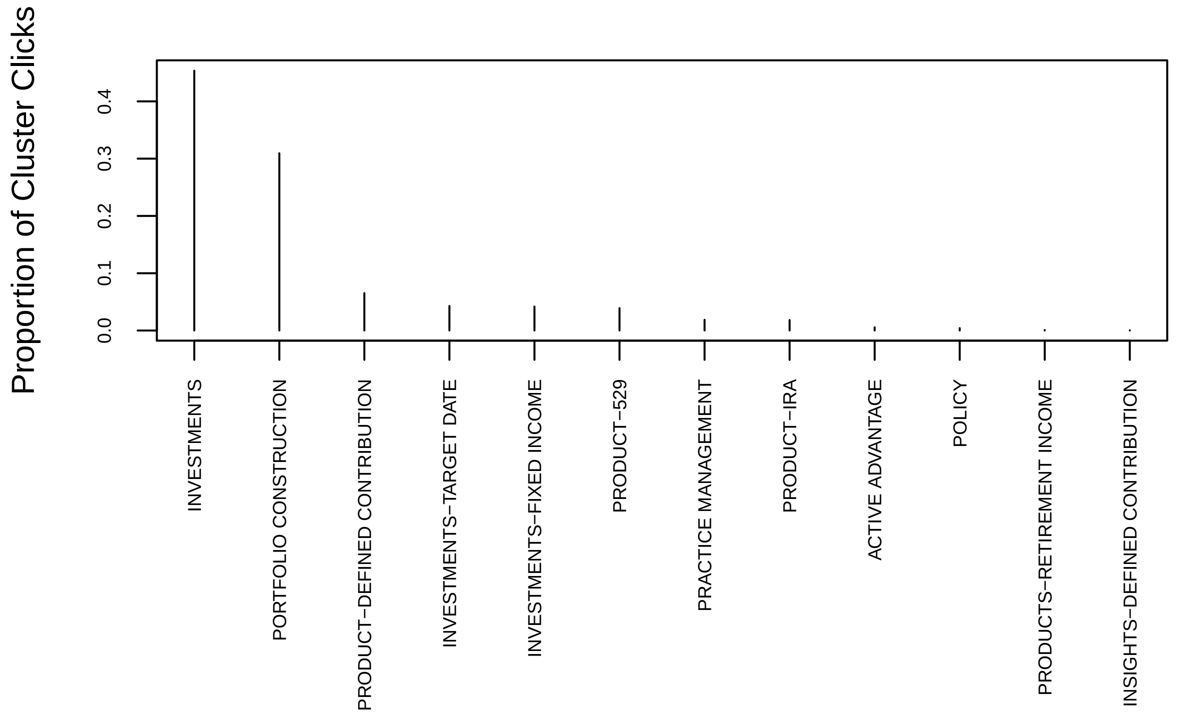
Inside look into clustering:

Here are content preferences of clusters 1 and 6. Notice that the differences are neglible, which means that the gamet of content behavior cannot be easily broken down into segments and that in order to accurately recommend particular content pillars to specific advisors, we have to use a more robust approach. All clusters look like this and it doesn't change much when we alter the k-means algorithm or the number of clusters.

Cluster 1:



Cluster 6:



DATA DEEP DIVE

There are two primary data sets we used for this project. The first is the raw file at the advisor and content pillar levels of granularity and the second is a summary version of this data set at the advisor level of granularity. The distinction here is rooted in the fact that for the initial stages of this project, we are investigating advisor characteristics that may correlate to more or less digital activity in one or multiple content pillars. Therefore, we are developing non-behavioral models and will only rely only on digital engagement and particular content pillars as binary dependent variables from which to build our models.

The Raw Data was used, however, for measuring content preferences by cluster and will be used when we do user-user collaborative filtering down the road.

Raw Data: advisor and content pillar level

This data set pulls in from 3 primary sources - our internal database and two 3rd party data sources, Discovery and Adobe. All are as of June 2016 and preceding 90 days when applicable.

0. Dependent variables

- Digital engagement flag (the primary dependent variable of interest)
- Content pillar flags: 1 if engaged with each pillar, 0 if not
    - Active advantage flag
    - Investments flag
    - Investments - fixed income flag
    - Investments - target date flag
    - Insights flag
    - Insights - defined contribution flag
    - Product - defined contribution flag
    - Product - IRA flag
    - Portfolio construction flag
    - Product - 529 flag
    - Products - retirement income flag
    - Policy flag
    - Practice management flag

1. Identification data

- Internal advisor unique ID
- First Name
- Last Name
- CRD (Central Registration Depository): ID assigned to each advisor by Financial Industry Regulatory Authority (FINRA)

2. Demographic and professional data

- Gender (Male, Female)
- Licenses (6,7,63,65,66): each as a ‘yes’/‘no’
- Number of years advisor has been a finanical advisor
- Number of years since first CG transaction
- Parent firm channel (Independent Broker Dealer (IBD), Financial Conglomerate (FC), Other)
- Parent firm name (big 7: Merrill Lynch, Morgan Stanley, Wells Fargo, UBS, Edward Jones, Raymond James, LPL, Other)

Comments: I only included the licenses that were the most common. IBDs and FCs have inherent differences in the strategies and personalities of their advisors. Our hypotheis going in was that IBD clients would be more digitally engaged and FC less digitally engaged because FCs have a reputation of focussing much more on fund analytics (fund ratings, returns, risk metrics, etc.) than branding or the underlying investment manager narrative. The next variable, parent firm name, was included to see if any of the biggest 7 firms stood out as particularly digitally engaged or not.

4. Portfolio and transaction data

- Internal prospecting score (A,B,C,High Potential(HP),Other)
- Biggest asset class in portfolio (Portfolio Series, Income, Growth, Balances, Tax-exempt, Other)
- Dollar amount of this asset class
- Proportion of portfolio this asset class represents
- Second biggest asset class in portfolio (Portfolio Series, Income, Growth, Balances, Tax-exempt, Other)
- Dollar amount of this asset class
- Proportion of portfolio this asset class represents
- Biggest share class in portfolio (ABC Brokerage, F1F2 Advisory, R1R2R3 Small RP Plan, R4R5R6 Mid Large RP Plan, Other)
- Dollar amount of this share class class
- Proportion of portfolio this share class represents
- Biggest recordkeeping type in portfolio (IRA, Individual 403(b), Employer Sponsored IRAs, PlanPremier TPA, Adviser Sold Investment Only, RecordKeeper Connect, RecordKeeper Direct - Other, Other)
- Dollar amount of this recordkeeping type
- Proportion of portfolio this recordkeeping type represents
- Biggest account type in portfolio (SIMPLE, 403(b), 529, Traditional IRA, Rollover IRA, Roth IRA, Other 401(k), Other Individual Not Qualified, Other)
- Dollar amount of this account type
- Proportion of portfolio this account type represents
- Total transactions (Sale, Redemption, Exchange)
- Total sales
- Total redemptions
- Total advisor assets under management (AUM)

5. Digital engagement

- Content pillar
- Number of clicks (proxy for degree of engagement)

Model Data: Advisor level

To reshape the above data to advisor level as opposed to content pillar level, we ignore the 'Digital Engagement' metrics: content pillar and number of clicks. We also included fewer advisor identification metrics. It is this cleaner data set that sets the foundation for the logisitic model.

Data Wrangling in SQL

I performed almost all of the data engineering within the SQL server using Microsoft Visual Studio on the raw data file above.

There are various limitations to this data including: * Asset class coverage, share class coverage, account type coverage and recordkeeping type coverage have been moved from rows to columns, which limits how advisors look from a product perspective. For example, if an advisor has a CG portfolio composed of 5 asset classes, we will only get insight into the top two. The alternative approach could have been to create a column for every asset class, share class, account type and then populate the advisors assets under management (AUM) for each. * Advisor IDs in our system include historical advisors, as well as clients that are dually registered, which makes it challenging to know with confidence whether a row on the advisor attributes table truly represents a distinct advisor. There are some methods we can use to increase that confidence, but not 100%. * The content pillars may prove to be too coarse in nature to extract meaningful recommendations. We may need to make them more granular down the road. * Mix of categorical and continuous numerical variables which poses a challenge for segmentation, clustering and distance calculations in general. I think this will make our results especially sensitive to the segmenting and clustering algorithms. * In many of the categorical variables, I broke them down to coarser categories, including an 'Other' category - those in the 'Other' category are less likely to fall into meaningful buckets and may cause the model to overfit the training data.

I did a series of data cleaning techniques including: custom-querying from more than 10 different internal tables, spreading asset classes, share classes and account types from rows to columns to preserve advisor as most granular level in advisor attributes table, unioning two data sets with the same column names but content pillars represented, filtering down to only advisors that 1. have an internal prospecting score and 2. are in the XC or FC retail channel and joining the two data sets above at different levels of granularity to create a new table at the most granular level.

Data Modifying in R

I pulled data directly from our Microsoft cloud from the R server as xdf files. The "xtract_data" function was written internally to do that using the "RevoScaleR" package and others:

```r
xtract_data("select * from [MSS_U_NADBI_S].[dbo].[RYHL_CAPSTONE_DATA]", "lasr",
    "/users/teams/nad/ryhl/Capstone/Content_Rec_Data_Raw.xdf")
xtract_data("select * from [MSS_U_NADBI_S].[dbo].[RYHL_CAPSTONE_DATA_CLEAN]",
    "lasr", "/users/teams/nad/ryhl/Capstone/Content_Rec_Data.xdf")
```

I then read in the data and adjusted data types as necessary:

```r
data <- rxReadXdf("/users/teams/nad/ryhl/Capstone/Content_Rec_Data.xdf", stringsAsFactors = T)
data$AGE_BEEN_REP <- as.numeric(data$AGE_BEEN_REP)
data$YRS_BUS_W_CG <- as.numeric(data$YRS_BUS_W_CG)
data$FINCL_ITRMY_PRSON_UID <- as.factor(data$FINCL_ITRMY_PRSON_UID)
data$DIGITAL_ENGAGEMENT_FLAG <- as.numeric(as.character(data$DIGITAL_ENGAGEMENT_FLAG))
data$TXN_CNT <- as.numeric(data$TXN_CNT)
```

METHODOLOGY

The Logistic Model Function

I created a function that takes in a data set with the first two columns as the advisor ID and a binary flag (either "digital engagement flag" or a content pillar flag) specified by the second input and the rest of columns as all independent variables. The bin count is an input into the IV table calculation from the "Information" package (we're using 10); the seed sets the randomizer (we're using 1 the entire time); the IV_min sets the minimum IV and independent variable must have to remain in the model (we're using .02); the split ratio sets the proportion of rows used to build the training set (we're using 0.7); p value max is the maximum p value an independent variable must have to remain in the model (we're using .001), response min is the cut off point for a TRUE or FALSE recommendation (i.e. if response min is 0.65, we will only recommend the binary outcome if the model outputs a 0.65 or higher – we alter this depending on the dependent variable).

Note that all commentary and charts will be with respect to the primary dependent variable of interest: digital engagement.

```r
# logistic_model(data_clean, 'DIGITAL_ENGAGEMENT_FLAG', 10, 1, 0.02 , 0.7,
# 0.001, 0.65)
data_clean <- data[, -c(3:15)]
binary_flag <- "DIGITAL_ENGAGEMENT_FLAG"
bin_cnt <- 10
seed <- 1
IV_min <- 0.02
split_ratio <- 0.7
p_value_max <- 0.001
response_min <- 0.65
```

STEP 1: Eliminate independent variables with lower IVs than 0.02. Then, to address outliers, replace all values of independent variables with their respective weight of evidences. I also created the functions "replace with WOE smb" and "replace with WOE smb cat" which use the "smbinning" package WoE tables to replace the numeric and factoral variables. See those structures in the *appendix*.

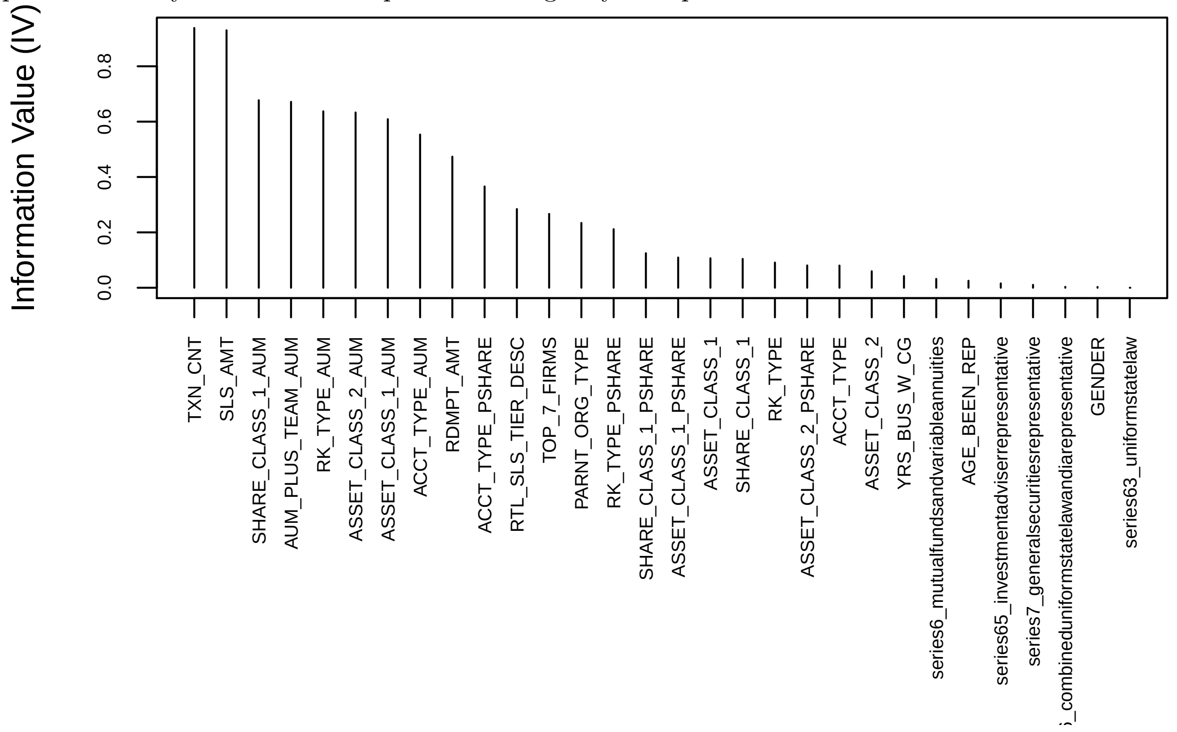
```r
set.seed(seed)

# builds a table of IVs for all vars -- you can customize bin amt: note that
# this IV calculation uses the 'Information' package, not the 'smbinning'
# package. The reason is efficiency.
IV <- create_infotables(data = data_clean[, -1], y = binary_flag, bins = bin_cnt)

# filters down to only vars with IV higher than input amount (for us, it
# will be 0.02)
model_ind_vars <- IV$Summary$Variable[IV$Summary$IV > IV_min]
data_clean <- data_clean[, names(data_clean) %in% c("FINCL_ITRMY_PRSON_UID",
    binary_flag, model_ind_vars)]

# replace all vars with WoEs: this is a common method of dealing with
# outliers.
data_clean <- replace_with_WOE_smb(data_clean, binary_flag)  #numeric indep vars
data_clean <- replace_with_WOE_smb_cat(data_clean, binary_flag)  #categorical indep vars
```

The IV summary is shown below. The various degrees of correlation with the dependent variable suggest that not only are only a portion of the IVs important but that there is likely ample co-linearity. The first step is removing any independent variables with an IV less than 0.02.



STEP 2: Create training and testing subsets.

Note here that the ratio of digitally engaged to not digitally engaged is far from 1:1. To address this, I reduced our training set to have a comparable number of rows with the binary flag equal to 1 as 0.

```r
# split entire data set into train/ test subsets

# split <- sample.split(data_clean[,2], SplitRatio = 0.7)
train_t <- dplyr::sample_frac(data_clean, split_ratio)
test_t <- dplyr::setdiff(data_clean, train_t)

# reduce train to have comparable number of dig engs and not dig engs
n_bin_flag <- nrow(train_t[train_t[[binary_flag]] == 1, ])
train_not_eng <- dplyr::sample_n(train_t[train_t[[binary_flag]] == 0, ], n_bin_flag,
    replace = FALSE)
train_t <- rbind(train_t[train_t[[binary_flag]] == 1, ], train_not_eng)
test <- dplyr::setdiff(data_clean, train_t)
data_clean <- train_t
```

STEP 3: Build model and use it to identify the independent variables with p-values lower than 0.001.

```r
dep_var <- as.formula(paste0(binary_flag, " ~ ", paste(colnames(data_clean[,
    -which(names(data_clean) %in% c("FINCL_ITRMY_PRSON_UID", binary_flag))]),
    sep = "+", collapse = "+")))
model1 <- glm(dep_var, as.data.frame(data_clean), family = "binomial")

# filter down by p-value
ind_vars <- rownames(summary(model1)$coef)[summary(model1)$coef[, 4] < p_value_max]
```

```r
ind_vars_low_p_val <- ind_vars[2:length(ind_vars)]

# rebuild model
data_clean <- data_clean[, names(data_clean) %in% c("FINCL_ITRMY_PRSON_UID",
    binary_flag, ind_vars_low_p_val)]
```

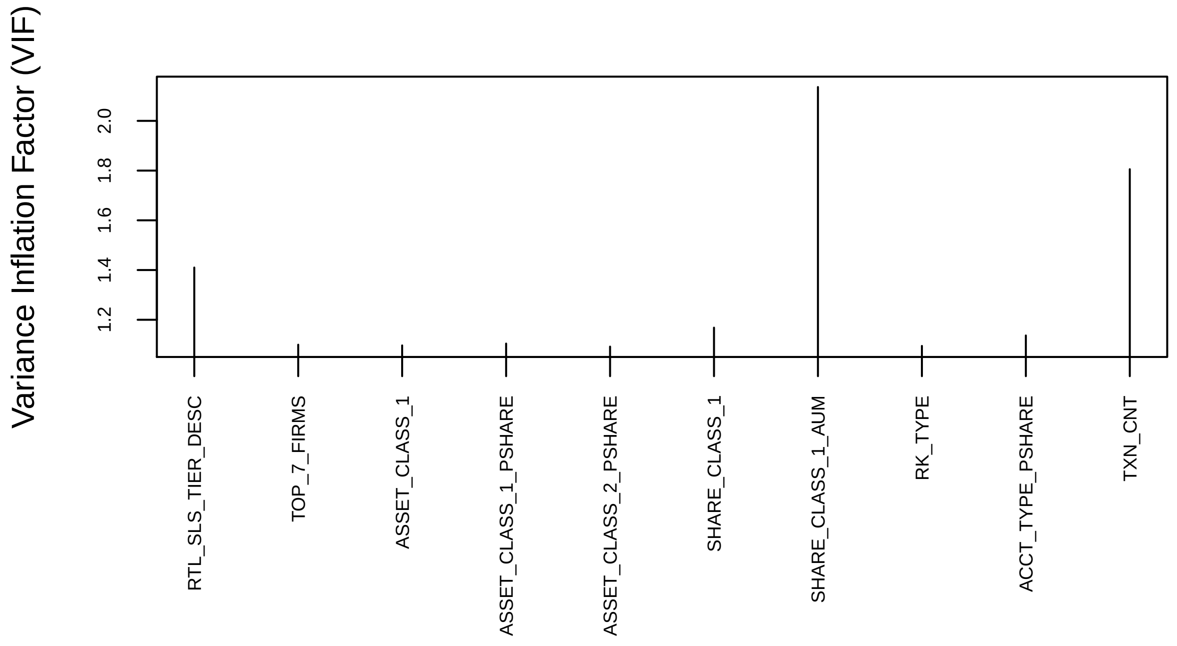
STEP 4: Re-build model with reduced number of independent variables and reduce further by ordering variables by IV, then iteratively excluding variables with inferior IVs (i.e. lower signficance) and high co-linearity.

```r
# ordered by IV
ind_vars_pre_multi_co_ord <- setdiff(IV$Summary$Variable, setdiff(IV$Summary$Variable,
    colnames(data_clean[, -c(1:2)])))

# bad will represent the less significant, highly correlated variables
bad <- NULL
i <- 1
while (i <= length(ind_vars_pre_multi_co_ord)) {
    j <- i + 1
    while (j <= length(ind_vars_pre_multi_co_ord)) {
        if (cor(data_clean[[which(names(data_clean) == ind_vars_pre_multi_co_ord[i])]],
            data_clean[[which(names(data_clean) == ind_vars_pre_multi_co_ord[j])]]) >=
            0.8) {
            bad <- c(bad, ind_vars_pre_multi_co_ord[j])
        }
        j <- j + 1
    }
    i <- i + 1
}

ind_vars <- setdiff(ind_vars_pre_multi_co_ord, bad)[1:10]
data_clean <- data_clean[, names(data_clean) %in% c("FINCL_ITRMY_PRSON_UID",
    binary_flag, ind_vars)]
```

Notice how now all variance inflation factors (VIFs) are less than or around 2. This suggests that co-linearity has been greatly reduced.



STEP 5: Re-build final model with the leftover independent variables (these should be significant and should have minimized co-linearity) and print out confusion matrix with respect to test set from earlier as well as area under the curve (AUC) of the receiver operating characteristic (ROC) using the “ROCR” package. ROC curves chart true positive rate vs. false positive rate to get a gauge of how much better the model is from a random guess.

```
dep_var <- as.formula(paste0(binary_flag, " ~ ", paste(colnames(data_clean[,
    -which(names(data_clean) %in% c("FINCL_ITRMY_PRSON_UID", binary_flag))]),
    sep = "+", collapse = "+")))
model3 <- glm(dep_var, as.data.frame(data_clean), family = "binomial")

# Ensure test subset has same columns as training subset
test_t <- test_t[, colnames(data_clean)]

predictTest = predict(model3, type = "response", newdata = test_t)
print(table(test_t[, binary_flag], predictTest > response_min))

### calculate AUC

ROCRpred = prediction(predictTest, test_t[, binary_flag])
print(as.numeric(performance(ROCRpred, "auc")@y.values))
```

K-means Clustering

We then applied k-means clustering to the output data set to investigate whether advisors' digital engagement behavior creates signficicant content preference differences across clusters.

STEP 1: Read in data, filter down to only digitally engaged advisors, build training set on 70% of advisors and testing set on the leftover 30% then scale the training data.

```r
set.seed(1)
split_ratio = 0.7

data_new <- read.csv("/users/teams/nad/ryhl/Capstone/DIGITAL_ENGAGEMENT_FLAG/model_data_clean.csv")
row.names(data_new) <- NULL
data_new <- filter(data_new, DIGITAL_ENGAGEMENT_FLAG == 1)  #use only digitally engaged population
data_new <- data_new[, -c(1, 3)]  #remove row number column and digital engagement flag from data
data_new[, -1] <- scale(data_new[, -1])
```

STEP 2: Clean data, apply 'Lloyd' k-mean algorithm to data and add on the cluster as a column

```r
row.names(data_new) <- NULL

# run k-means on chosen cluster amount
data_new_k_means <- kmeans(data_new[, -1], 6, iter.max = 1000, algorithm = "Lloyd")

data_new_cluster <- data_new_k_means$cluster
data_new <- cbind(data_new, unname(train_t_cluster))
data_new <- rename(data_new, c(`unname(train_t_cluster)` = "CLUSTER"))
data_new$CLUSTER <- as.factor(data_new$CLUSTER)
```

STEP 3: Join results with raw data file explained above to gauge content preferences by cluster.

```r
data_w_con_pillars <- rxReadXdf("/users/teams/nad/ryhl/Capstone/Content_Rec_Data_Raw.xdf",
    stringsAsFactors = T)
temp <- left_join(data_new, data_w_con_pillars, by = "FINCL_ITRMY_PRSON_UID")
new_temp <- temp[colnames(temp) %in% c("FINCL_ITRMY_PRSON_UID", paste0(colnames(data_new[,
    -1]), ".x"), "CLUSTER", "CONTENT_PILLAR_CATEGORY", "CLICK_CNT")]

#######

pie_chart <- new_temp %>% dplyr::select(CLUSTER, CONTENT_PILLAR_CATEGORY, CLICK_CNT) %>%
    dplyr::group_by(CLUSTER, CONTENT_PILLAR_CATEGORY) %>% dplyr::summarise(CLICKS = sum(CLICK_CNT)) %>%
    dplyr::arrange(CLUSTER, CONTENT_PILLAR_CATEGORY)

write.csv(pie_chart, "/users/teams/nad/ryhl/Capstone/pie_chart_k_means.csv")
```

WHAT'S NEXT?

Formulate experiment aimed at getting more advisors to become digitally active.

There are around 170,000 usable advisors (data is complete enough for modeling across all variables). We can use the model to identify the advisors likely to be digitally engaged who are not currently and directly target them for email campaigns. We can create a control group as well to confirm that the model outperforms the control group.

Build recommendation engine.

1. Add another layer to gauge the caliber of digital engagement: 0 if not engaged, 1 if total clicks between 1 and 5, 2 if total clicks greater than 5.
2. Apply hybrid filtering (nearest neighbor for non-digitally engaged, nearest neighbor with similar content preferences for minimally engaged) to recommend specific content to specific advisors.

Validate model using experimentation and backtesting.

1. Apply model to digitally engaged advisors in different 90-day periods.
2. Study the advisors who made a transition from non-digitally-engaged to digitally-engaged in the observed data. Back test if the materials recommended by the model overlap and/or coincide with those that were engaged with by the advisors.

APPENDIX

Commentary on variables that didn't prove to big drivers of digital engagement:

We hypothesized that newer the advisor is to our products, the more likely s/he would use digital media. We included two proxies for this: number of years the advisor has been a licensed financial advisor and the number of years the advisor did business with us. Despite ubiquitous techonology-engagement trends of younger generations leveraging digital means more regularly, advisor digitally engagement only goes up as they get older. I believe our logical flaw on this hypothesis was that investment digital engagement is a proxy for general digital engagement. The newer reps may have much higher level reading to do than on particular investment manager strategies.

```
##   Cutpoint CntRec CntGood CntBad GoodRate BadRate     WoE     IV
## 1     <= 1  16761    3403  13358   0.2030  0.7970 -0.4553 0.0190
## 2     <= 5  20260    6013  14247   0.2968  0.7032  0.0496 0.0003
## 3    <= 14  71419   18658  52761   0.2612  0.7388 -0.1273 0.0068
## 4    <= 17  19220    6047  13173   0.3146  0.6854  0.1336 0.0021
## 5    <= 56  36830   13014  23816   0.3534  0.6466  0.3079 0.0225
## 6  Missing      0       0      0       NA      NA      NA     NA
## 7    Total 164490   47135 117355   0.2866  0.7134  0.0000 0.0507
```

Additionally, gender and the specific licenses we selected did not at all correlate to digital engagement.

Functions I created:

These are functions I built to replace numeric and categorical variables within the dataset with their weight of evidences (WoEs) respectively. The 'xtract_data' function was built by another analyst to access our Microsoft database (called LASR) through the R server.

```
replace_with_WOE_smb <- function(data_clean, binary_flag) {
    numeric_ind_vars <- split(names(data_clean), sapply(data_clean, function(x) paste(class(x),
        collapse = " ")))$numeric
    for (var in numeric_ind_vars[-1]) {
        woe <- smbinning(data_clean, binary_flag, var, p = 0.05)
        for (bin_n in 1:length(woe$bands) - 1) {
            data_clean[data_clean[[var]] >= woe$bands[bin_n] & data_clean[[var]] <=
                woe$bands[bin_n + 1], var] <- woe$ivtable$WoE[bin_n]
        }
        data_clean[[var]] <- as.numeric(data_clean[[var]])
    }
    data_clean
}

replace_with_WOE_smb_cat <- function(data_clean, binary_flag) {
    categ_ind_vars <- split(names(data_clean), sapply(data_clean, function(x) paste(class(x),
        collapse = " ")))$factor
    for (var in categ_ind_vars[-1]) {
        woe <- smbinning.factor(data_clean, binary_flag, var)
        data_clean[[var]] <- unfactor(data_clean[[var]])
        for (n in 1:length(woe$cuts)) {
            data_clean[data_clean[[var]] == woe$cuts[n], var] <- woe$ivtable$WoE[n]
        }
        data_clean[[var]] <- as.numeric(data_clean[[var]])
    }
    data_clean
}

xtract_data <- function(sql_file, db_nm, output_file_nm) {
    # Extracts data from DB based on sql_file or query string Args: sql_file:
    # file name with path or query string db_nm : self-defined DB names,
    # 'mssbi5', 'mssbi6', 'prd1', 'prd2' output_file_name: outfile name with
    # path Returns: the queried data as in list

    connection_string <- switch(db_nm, mssbi5 = paste0("DSN=afsdwp5;uid=", Sys.getenv("oracle_uid"),
        ";pwd=", Sys.getenv("oracle_pwd")), mssbi6 = paste0("DSN=afsdwp6;uid=",
        Sys.getenv("oracle_uid"), ";pwd=", Sys.getenv("oracle_pwd")), prd1 = paste0("DSN=afbi_prd1;uid=
        Sys.getenv("oracle_uid"), ";pwd=", Sys.getenv("oracle_pwd")), prd2 = paste0("DSN=fbi_prd2;uid="
        Sys.getenv("oracle_uid"), ";pwd=", Sys.getenv("oracle_pwd")), lasr = paste0("DSN=lasr_uat;uid="
        Sys.getenv("lasr_uid"), ";pwd=", Sys.getenv("lasr_pwd")), stop("invalid DB name"))
    len <- nchar(sql_file)
    if (substr(sql_file, len - 3, len) == ".sql") {
        # is .sql file
        query_string <- readChar(sql_file, file.info(sql_file)$size)
        # cat(query_string)
    } else {
        query_string <- sql_file
    }
    datasource <- RxOdbcData(sqlQuery = query_string, connectionString = connection_string,
        rowBuffering = F)
    xdf_path <- file.path(output_file_nm)
    xdf <- rxImport(datasource, xdf_path, overwrite = TRUE)
    res <- rxReadXdf(xdf)
    res
}
```