week5

Paul Anderson

9/18/2017

Rundown

Alright gang, looks like people are starting to get a better handle on the data. There really aren't any shortcuts here. It just takes time. OAKS is currently being a real pain right now, so I've downloaded your week 4 reports and gone through them and created notes. Here is a rundown on some of the things I think will help you reach your goals:

- Regression - some of you want to predict something that is continuous. Easy to do this with R, see example below.
- Clustering - other people seem to be thinking along the lines of clustering the data somehow.
- Dates - other people really want to use the dates
- More classification - I'll show another classification algorithm as well
- Budgeting - a lot of people are thinking budgeting apps still which is cool, but let's think about how we can break the mold of traditional budgeting apps that break things up into house, fast food, etc. How about breaking things up based on their patterns? Early in the month? Consistent? Similar to other users? Variable categories? Any shifts in these patterns over time could indicate...?
- Correlation
- Merging data

Loading the data

This is primarily from last week, but I'm including it here as well.

```
library(readxl)
month_end_balances <- as.data.frame(read_excel("/usr/local/Learn2Mine-Main/galaxy-dist/lesson_datasets/
    sheet = "Month end balances ", col_types = c("numeric",
        "numeric", "numeric", "numeric",
        "numeric", "numeric", "numeric",
        "numeric", "numeric", "numeric",
        "numeric", "numeric", "numeric",
        "numeric", "numeric", "numeric",
        "numeric", "numeric", "numeric",
        "numeric", "numeric", "numeric")))

month_end_balances$mortgage_flag = factor(month_end_balances$mortgage_flag )

daily_interactions_WF <- read_excel("/usr/local/Learn2Mine-Main/galaxy-dist/lesson_datasets/Fake+Data+a
    sheet = "Daily interactions with WF")

daily_interactions_WF$Des1 = factor(daily_interactions_WF$Des1)
levels(daily_interactions_WF$Des1)
```

```
##   [1] "Account Closed"
##   [2] "ACCOUNT HISTORY COPY REQUEST"
##   [3] "Account Hold Add"
##   [4] "Account Hold Remove"
```

```
##   [5] "Account Inquiry"
##   [6] "Account Maintenance"
##   [7] "Account Open"
##   [8] "Account Open - IRA Account"
##   [9] "ACH Prenote"
##  [10] "Add Banker Note"
##  [11] "Add Contact Event"
##  [12] "ADDRESS CHANGE"
##  [13] "Advance"
##  [14] "Advance Reversal"
##  [15] "Agent Call"
##  [16] "ATM/CHECK CARD MAINTENANCE"
##  [17] "ATM/DEBIT PIN CARD"
##  [18] "ATM Failure"
##  [19] "ATM Time Out"
##  [20] "AUTHENTICATION_TRACKER_ON_OFF"
##  [21] "Authorized Credit"
##  [22] "Authorized Debit"
##  [23] "Balance Inquiry"
##  [24] "Balance Transfer Initiated"
##  [25] "BALANCE TRANSFERS"
##  [26] "Bank-initiated debit"
##  [27] "Bank-initiated transfer"
##  [28] "Bank Product Purchase"
##  [29] "Bank Product Purchase Reversal"
##  [30] "Bill Payment Miscellaneous"
##  [31] "Bill Payment Reject"
##  [32] "Bill Payment Reversal"
##  [33] "Book Transfer Create"
##  [34] "Business Credit Only flow selection"
##  [35] "Business Deposit and Credit flow selected"
##  [36] "Business Deposit - IOLTA flow selection"
##  [37] "Business - Deposit - RETA flow  selected"
##  [38] "Business  Deposit Only - Special Relationship flow"
##  [39] "Cancel Contact Event"
##  [40] "Cancelled"
##  [41] "Cash Check on Credit Card/LOC"
##  [42] "Cash EE Bond"
##  [43] "Cash Non-WFB Check"
##  [44] "Cash WFB Check"
##  [45] "Cash WFB Check-OWNER"
##  [46] "Check"
##  [47] "Check Card Credit"
##  [48] "Check Card Purchase Preauthorization"
##  [49] "Check Card Purchase Transaction"
##  [50] "CHECK ORDER"
##  [51] "CIVSALES_CARDS (PI)"
##  [52] "CIVSALES_CIP_UPDATE"
##  [53] "CIVSALES_CIP_VALIDATION"
##  [54] "CIVSALES_CREDIT_OPTION_GUIDE"
##  [55] "CIVSALES_CUST_NEEDS_ASSESSMENT"
##  [56] "CIVSALES_CUSTOMER_OFFERS"
##  [57] "CIVSALES_CUSTOMER_SESSION"
##  [58] "CIVSALES_CUST_PROFILE_EDITS"
```

```
##  [59] "CIVSALES_NEW_CUSTOMER_PROFILE"
##  [60] "CIVSALES_NEW_PMA"
##  [61] "CIVSALES_ONLINE_BANKING_BILL_PAY"
##  [62] "CIVSALES_PARTNER_REFERRAL_CREATED"
##  [63] "CIVSALES_PMA_ACCOUNT_CONVERSION"
##  [64] "CIVSALES_PMA_ADD/REMOVE_OWNERS"
##  [65] "CIVSALES_REPORT_REASON_FOR_CALL"
##  [66] "CIVSALES_RISK_SCREENING"
##  [67] "CIVSALES_SAVE_AS_YOU_GO_MAINTENANCE"
##  [68] "CIVSALES_TAB_CLICKER"
##  [69] "CIV_SPECIAL_RATES"
##  [70] "CIVSSALES_PMA_ACCOUNT_LINKAGES"
##  [71] "CLAIMS_PROCESSING"
##  [72] "CLIENT_SYSTEM_INFO"
##  [73] "Close"
##  [74] "CNA Added"
##  [75] "CNA Updated"
##  [76] "College Information Maintenance"
##  [77] "Common Customer Event History tool Account Level"
##  [78] "Common Customer Event History tool Customer Level"
##  [79] "Competitor Accounts Inquiry"
##  [80] "Complete Contact Event"
##  [81] "Consumer  Credit Only flow selected"
##  [82] "Consumer Deposit and Credit flow selected"
##  [83] "Consumer Deposit Only flow selected"
##  [84] "Consumer Establish/Maintain Remittance Agreement"
##  [85] "Consumer  New ATM/Check Card Only flow selected"
##  [86] "Consumer Recommendations Create MSR"
##  [87] "CONTACT_EVENT_CREATED"
##  [88] "Correction"
##  [89] "CORRESPONDENCE"
##  [90] "CreateMSR"
##  [91] "Credit Adjustment"
##  [92] "Credit Application"
##  [93] "CREDIT_CARD_FEE_REIMBURSEMENT"
##  [94] "CREDIT CARD REWARDS"
##  [95] "Credit Offer at ATM"
##  [96] "Credit Options guide (COG) tool accessed"
##  [97] "Credit Reversal"
##  [98] "CUAC Maintenance"
##  [99] "Customer Address Change"
## [100] "UpdateMSR"
## [101] "Verify Non-WFB Funds"
## [102] "Verify WFB Funds"
## [103] "Wire Transfer Create"
```

Example: Regression

```
library(randomForest)
```

```
## randomForest 4.6-10
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

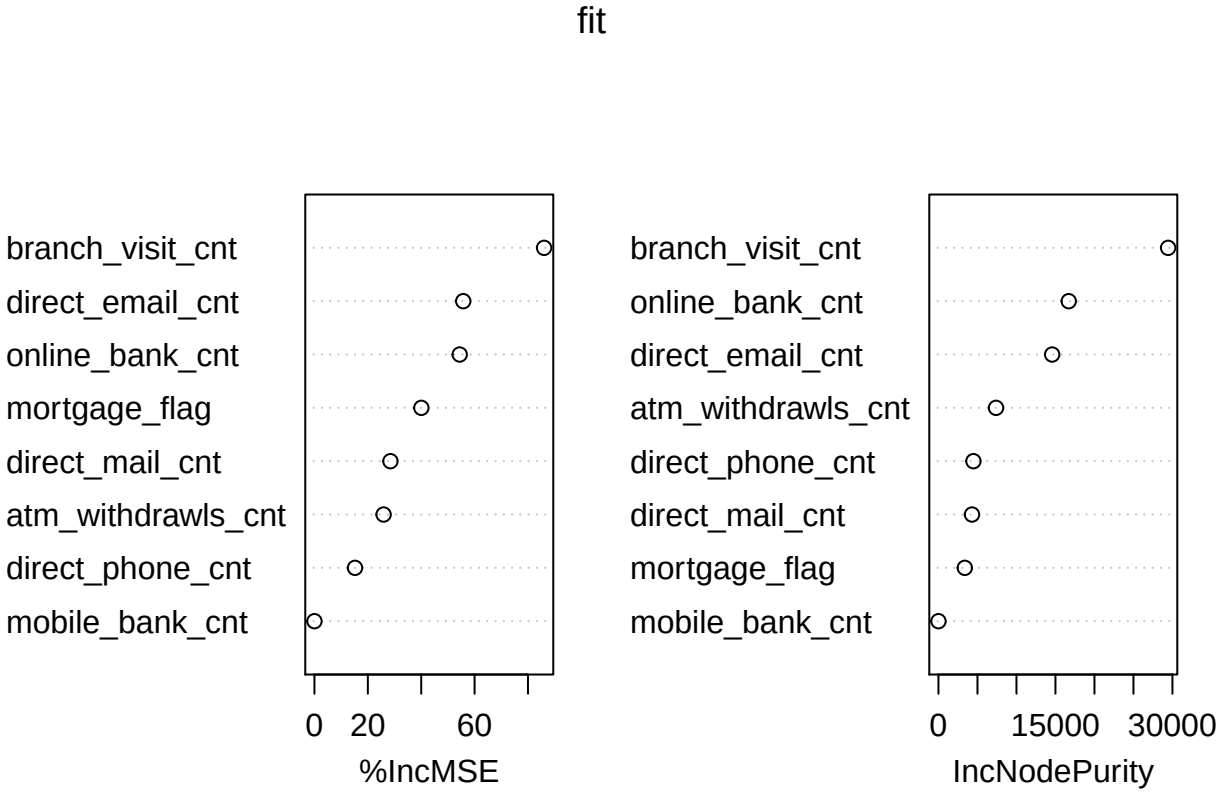
```
fit <- randomForest(age ~ branch_visit_cnt + online_bank_cnt + direct_phone_cnt + atm_withdrawls_cnt + d
                      data=month_end_balances,
                      importance=TRUE,
                      ntree=2000)
print(fit)
```

```
##
## Call:
##  randomForest(formula = age ~ branch_visit_cnt + online_bank_cnt +      direct_phone_cnt + atm_withdr
##                Type of random forest: regression
##                      Number of trees: 2000
## No. of variables tried at each split: 2
##
##           Mean of squared residuals: 202.5389
##                     % Var explained: 45.28
```

So with a simple change, random forest can also be used for regression. Now what is reported is percent variable explained which we want to be high (near 100%) and mean of squared residuals (error term) which we want to be small. Looks like predicting an age from the data I have is a little tricky as expected.

Now everyone loves a graph, so a cool thing about random forest is you can see how important a variable is to prediction:

```
varImpPlot(fit)
```



But this shows us that branch visit count is a good indicator, which supports the trend of younger people using banking differently. Can you think of other ways to find trends for wells fargo that might be unexpected?

Clustering

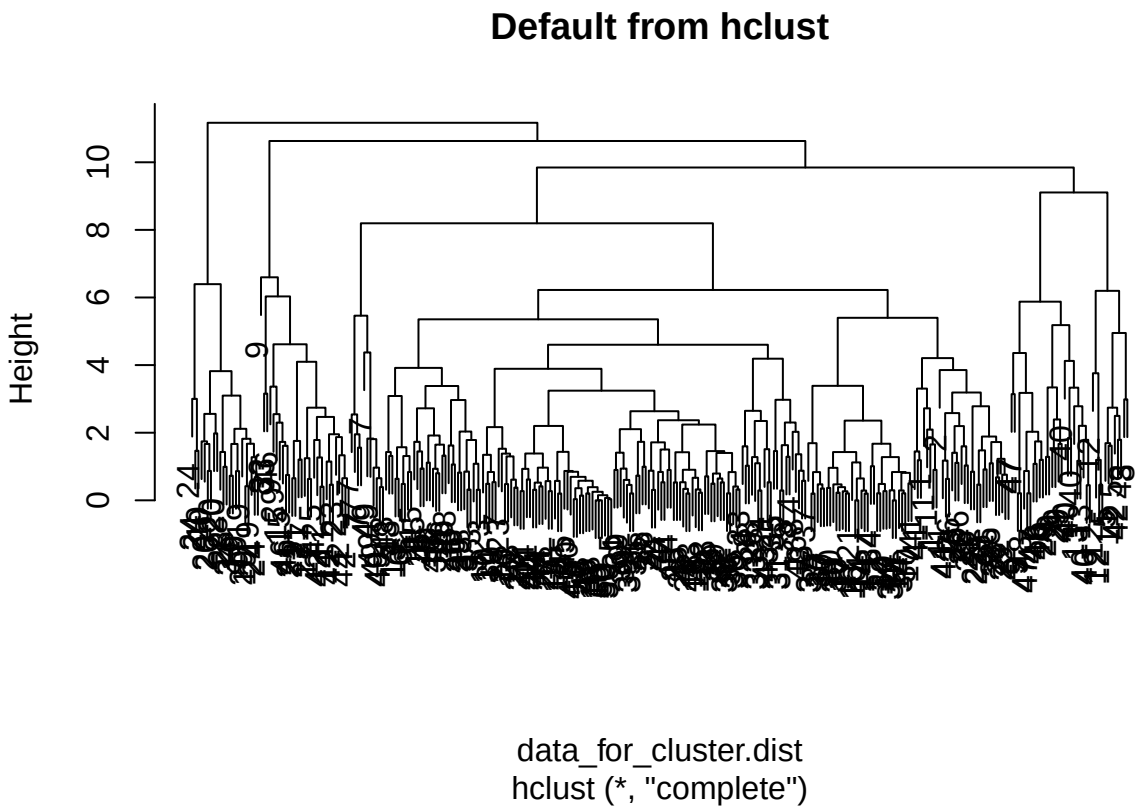
There is a lot of different ways to cluster. A quick google of clustering and R gives tons of results (e.g., https://www.stat.berkeley.edu/~s133/Cluster2a.html). I'm going to show you some hierarchical clustering in R which can create a cool graph.

Before we can do any of that we need to make select the data that we want and we need to clean it up. Specifically, we need to pick only numeric data and we need to scale it so it is in the same range.

```
data_for_cluster = as.matrix(month_end_balances[,c('age','branch_visit_cnt','online_bank_cnt','direct_pl

data_for_cluster = apply(data_for_cluster,2,as.numeric)
means = apply(data_for_cluster,2,mean)
sds = apply(data_for_cluster,2,sd)
data_for_cluster = scale(data_for_cluster,center=means,scale=sds)
```

Now we can compute the distances between the rows and then cluster and visualize.

```
data_for_cluster.dist = dist(data_for_cluster)
data_for_cluster.hclust = hclust(data_for_cluster.dist)
plot(data_for_cluster.hclust,labels=month_end_balances$masked_id,main='Default from hclust')
```



It is a little hard to tell what exact labels there are for each group, but you can see that there are clear groups of people in the dataset. It is confusing because there are many people in our dataset. One could dig into what these clusters of people are and try to find meaning in the clustering.

Let's say you want to break this into 4 groups:

```
groups.4 = cutree(data_for_cluster.hclust,4)
table(groups.4)
```

```
## groups.4
##   1   2   3   4
## 211  38  29  22
```

There are 3 smallish groups and one large group. If you want to figure out who is in one of the smaller groups:

```
month_end_balances$masked_id[groups.4 == 2] # the 2 is the group number
```

```
##  [1] 12 12 12 12  5  5  5 47 47 47 47 47 47 11 42 42 40 40 40 40 40 40 43
## [24]  8  9  9  9  9 13 13 13 29 29 29 29 48 48 48
```

So this pointed out to me that we actual have multiple entries for the same user, so we can group our data by that which is an important thing to show everyone, so I'll give it it's own heading.

Grouping/Aggregating Data

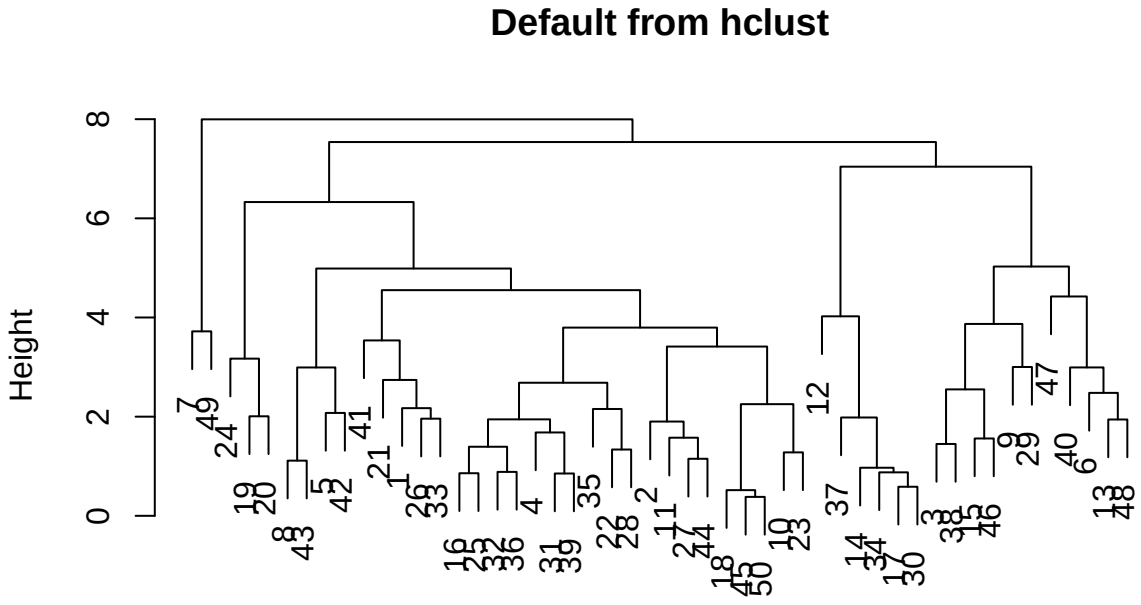
In order to do this we need to add the column with the labels back to the data:

```
data_for_cluster_with_ids = as.data.frame(month_end_balances[,c('age','branch_visit_cnt','online_bank_c

data_for_cluster_agg = aggregate(. ~ masked_id, data_for_cluster_with_ids, mean)

head(data_for_cluster_agg)
```

```
##   masked_id  age branch_visit_cnt online_bank_cnt direct_phone_cnt
## 1         1 59.0         9.666667       39.000000        0.1666667
## 2         2 38.0         6.166667        6.333333        0.0000000
## 3         3 57.0         1.333333        1.500000        0.3333333
## 4         4 45.0         2.833333        8.166667        0.3333333
## 5         5 35.5         2.333333       40.500000        1.6666667
## 6         6 65.0        10.666667       22.833333        0.0000000
##   atm_withdrawls_cnt direct_mail_cnt mortgage_flag direct_email_cnt
## 1          5.1666667       0.8333333      1.000000       12.0000000
## 2          1.5000000       0.0000000      1.000000        9.6666667
## 3          0.6666667       1.6666667      2.000000        0.1666667
## 4          4.8333333       0.5000000      1.000000        4.3333333
## 5          0.5000000       0.6666667      1.000000        5.6666667
## 6          0.1666667       1.1666667      1.833333        0.5000000
```

Much better. Now we have 1 entry per person and we can do our clustering again.

```
orig_data_for_cluster_agg = data_for_cluster_agg
data_for_cluster_agg = apply(data_for_cluster_agg[,-1],2,as.numeric) # Remove the masked_id while we ar
means = apply(data_for_cluster_agg,2,mean)
sds = apply(data_for_cluster_agg,2,sd)
data_for_cluster_agg = scale(data_for_cluster_agg,center=means,scale=sds)

data_for_cluster_agg.dist = dist(data_for_cluster_agg) # Take out the masked_id from the cluster
data_for_cluster_agg.hclust = hclust(data_for_cluster_agg.dist)
plot(data_for_cluster_agg.hclust,labels=orig_data_for_cluster_agg$masked_id,main='Default from hclust')
```



Much better! Now let's cut this into 5 groups:

```
groups.5 = cutree(data_for_cluster_agg.hclust,5)
table(groups.5)
```

```
## groups.5
##  1  2  3  4  5
## 28 11  2  6  3
```

If you want to figure out who is in group 2:

```
orig_data_for_cluster_agg$masked_id[groups.5 == 2] # the 2 is the group number
```

```
##  [1]  3  6  9 13 15 29 38 40 46 47 48
```

And those are the people who are in group 5. You could plot the values of those people, etc...

Clustering is definitely a great way to explore the data, but for now I'm going to move onto one of the othe topics.

Dates

Let's take a look at one of the columns:

```
head(month_end_balances$asof_yyyymm)
```

```
## [1] 201612 201611 201610 201609 201608 201607
```

I want to use another date packages, so to install this or any package you want use:

```
# install.packages("lubridate") # uncomment to install package
```

Those are currently treated as just numbers. One of our problems is we lack a day of the month for this column. We can add this in:

```
asof_yyyymmdd = paste(as.character(month_end_balances$asof_yyyymm),"01",sep="")
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

```
month_end_balances$asof_yyyymm_asdate = ymd(asof_yyyymmdd)
head(month_end_balances$asof_yyyymm_asdate)
```

```
## [1] "2016-12-01" "2016-11-01" "2016-10-01" "2016-09-01" "2016-08-01"
## [6] "2016-07-01"
```

Then you can ask things like:

```
month(month_end_balances$asof_yyyymm_asdate)
```

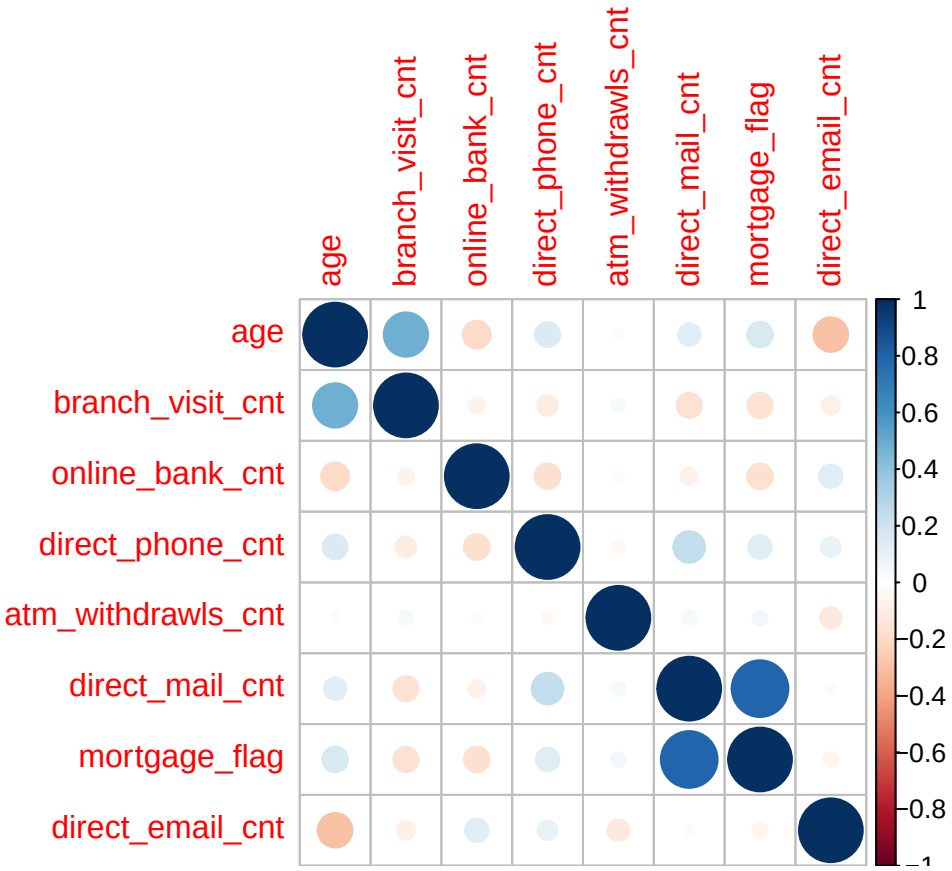
```
##   [1] 12 11 10  9  8  7 12 11 10  9  8  7 12 11 10  9  8  7 12 11 10  9  8
##  [24]  7 12 11 10  9  8  7 12 11 10  9  8  7 12 11 10  9  8  7 12 11 10  9
##  [47]  8  7 12 11 10  9  8  7 12 11 10  9  8  7 12 11 10  9  8  7 12 11 10
##  [70]  9  8  7 12 11 10  9  8  7 12 11 10  9  8  7 12 11 10  9  8  7 12 11
##  [93] 10  9  8  7 12 11 10  9  8  7 12 11 10  9  8  7 12 11 10  9  8  7 12
## [116] 11 10  9  8  7 12 11 10  9  8  7 12 11 10  9  8  7 12 11 10  9  8  7
## [139] 12 11 10  9  8  7 12 11 10  9  8  7 12 11 10  9  8  7 12 11 10  9  8
## [162]  7 12 11 10  9  8  7 12 11 10  9  8  7 12 11 10  9  8  7 12 11 10  9
## [185]  8  7 12 11 10  9  8  7 12 11 10  9  8  7 12 11 10  9  8  7 12 11 10
## [208]  9  8  7 12 11 10  9  8  7 12 11 10  9  8  7 12 11 10  9  8  7 12 11
## [231] 10  9  8  7 12 11 10  9  8  7 12 11 10  9  8  7 12 11 10  9  8  7 12
## [254] 11 10  9  8  7 12 11 10  9  8  7 12 11 10  9  8  7 12 11 10  9  8  7
## [277] 12 11 10  9  8  7 12 11 10  9  8  7 12 11 10  9  8  7 12 11 10  9  8
## [300]  7
```

Correlation

As I mentioned on Facebook, correlation is really easy and provides a cool visual. I'm going to use our clean data we used for clustering as correlation only makes sense for numeric data.

```
# install.packages('corrplot') # uncomment to install package
```

```
library(corrplot)
M <- cor(data_for_cluster_agg)
corrplot(M, method="circle")
```



Bigger circles are more correlation. This is a great tool for you to use to explore the data and get a feel for thing.

Merging data

OK. We need to get to how to combine these different sheets. The good news is everything seems to have a masked_id so that is how we will do it. The problem as we've seen above is that there are multiple entries for each user in a sheet. This means we'll need to aggregate before we merge, but that's easy :) I'll copy duplicate code down here, so you have it all in one place.

```
library(readxl)
month_end_balances <- as.data.frame(read_excel("/usr/local/Learn2Mine-Main/galaxy-dist/lesson_datasets/
    sheet = "Month end balances ", col_types = c("numeric",
        "numeric", "numeric", "numeric",
        "numeric", "numeric", "numeric",
        "numeric", "numeric", "numeric",
        "numeric", "numeric", "numeric",
        "numeric", "numeric", "numeric",
        "numeric", "numeric", "numeric",
        "numeric", "numeric", "numeric",
        "numeric", "numeric", "numeric")))

month_end_balances$mortgage_flag = factor(month_end_balances$mortgage_flag )

daily_interactions_WF <- as.data.frame(read_excel("/usr/local/Learn2Mine-Main/galaxy-dist/lesson_datase
    sheet = "Daily interactions with WF"))
```

Just take a look at what we have:

```
head(daily_interactions_WF$masked_id)
```

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

```
head(month_end_balances$masked_id)
```

```
## [1] 12 12 12 12 12 12
```

Duplicates for both, so we need aggregation.

```
summary(daily_interactions_WF)
```

```
##    masked_id         Date               Des1               Des2
##  Min.   : 1.00   Length:6669        Length:6669        Length:6669
##  1st Qu.:14.00   Class :character   Class :character   Class :character
##  Median :26.00   Mode  :character   Mode  :character   Mode  :character
##  Mean   :25.83
##  3rd Qu.:38.00
##  Max.   :50.00
##      Des3
##  Length:6669
##  Class :character
##  Mode  :character
##
##
##
```

```
summary(month_end_balances)
```

```
##    masked_id     asof_yyyymm          age        tenure_altered
##  Min.   : 1.0   Min.   :201607   Min.   :13.00   Min.   : 0.677
##  1st Qu.:13.0   1st Qu.:201608   1st Qu.:35.00   1st Qu.: 8.475
##  Median :25.5   Median :201610   Median :50.50   Median :17.156
##  Mean   :25.5   Mean   :201610   Mean   :50.46   Mean   :18.204
##  3rd Qu.:38.0   3rd Qu.:201611   3rd Qu.:65.00   3rd Qu.:25.099
##  Max.   :50.0   Max.   :201612   Max.   :90.00   Max.   :47.492
##  checking_acct_ct savings_acct_ct mortgage_flag   heloc_flag
##  Min.   :0.000    Min.   :0.000   0:235         Min.   :0.00
##  1st Qu.:1.000    1st Qu.:1.000   1: 65         1st Qu.:0.00
##  Median :1.000    Median :1.000                 Median :0.00
##  Mean   :1.687    Mean   :1.957                 Mean   :0.08
##  3rd Qu.:2.000    3rd Qu.:2.000                 3rd Qu.:0.00
##  Max.   :6.000    Max.   :7.000                 Max.   :1.00
##  personal_loan_flag    cc_flag       prot_acct_flag check_bal_altered
##  Min.   :0.00       Min.   :0.0000   Min.   :0      Min.   :    -0.93
##  1st Qu.:0.00       1st Qu.:0.0000   1st Qu.:0      1st Qu.:  2679.67
##  Median :0.00       Median :1.0000   Median :0      Median :  7100.13
##  Mean   :0.08       Mean   :0.5067   Mean   :0      Mean   : 19777.90
##  3rd Qu.:0.00       3rd Qu.:1.0000   3rd Qu.:0      3rd Qu.: 19236.06
##  Max.   :1.00       Max.   :1.0000   Max.   :0      Max.   :289952.76
##  sav_bal_altered   mortgage_bal_altered heloc_bal_altered
##  Min.   :      0   Min.   :     0       Min.   :    -2.15
##  1st Qu.:  21641   1st Qu.:     0       1st Qu.:     0.00
##  Median :  55539   Median :     0       Median :     0.00
##  Mean   : 133185   Mean   : 38325       Mean   :  4027.67
```

```
##  3rd Qu.: 123940   3rd Qu.:     0       3rd Qu.:     0.00
##  Max.   :1456972   Max.   :588001       Max.   :198524.90
##  personal_loan_bal_altered atm_withdrawls_cnt atm_deposits_cnt
##  Min.   :    0             Min.   : 0.000     Min.   :0.00000
##  1st Qu.:    0             1st Qu.: 0.000     1st Qu.:0.00000
##  Median :    0             Median : 0.000     Median :0.00000
##  Mean   : 1421             Mean   : 2.223     Mean   :0.05333
##  3rd Qu.:    0             3rd Qu.: 2.000     3rd Qu.:0.00000
##  Max.   :29283             Max.   :30.000     Max.   :6.00000
##  branch_visit_cnt phone_banker_cnt mobile_bank_cnt online_bank_cnt
##  Min.   : 0.000   Min.   :0.0000   Min.   :0       Min.   :  0.00
##  1st Qu.: 1.000   1st Qu.:0.0000   1st Qu.:0       1st Qu.:  2.75
##  Median : 4.000   Median :0.0000   Median :0       Median :  8.50
##  Mean   : 5.723   Mean   :0.2367   Mean   :0       Mean   : 20.34
##  3rd Qu.:10.000   3rd Qu.:0.0000   3rd Qu.:0       3rd Qu.: 27.00
##  Max.   :22.000   Max.   :5.0000   Max.   :0       Max.   :230.00
##  direct_mail_cnt  direct_email_cnt direct_phone_cnt
##  Min.   :0.0000   Min.   : 0.000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.: 0.000   1st Qu.:0.0000
##  Median :0.0000   Median : 1.000   Median :0.0000
##  Mean   :0.5767   Mean   : 3.813   Mean   :0.4067
##  3rd Qu.:1.0000   3rd Qu.: 6.000   3rd Qu.:0.0000
##  Max.   :6.0000   Max.   :31.000   Max.   :6.0000
```

Now we can ask a more specific question. What if we wanted to add the most common Des1 to the other numeric data from month_end_balances. First aggregation:

```
month_end_balances_4merge = as.data.frame(month_end_balances[,c('age','branch_visit_cnt','online_bank_cn

month_end_balances_4merge = aggregate(. ~ masked_id, month_end_balances_4merge, mean)

Mode <- function(x) {
  ux <- unique(x)
  ux[which.max(tabulate(match(x, ux)))]
}
daily_interactions_WF_4merge = aggregate(. ~ masked_id, daily_interactions_WF,Mode)

head(month_end_balances_4merge)
```

```
##   masked_id  age branch_visit_cnt online_bank_cnt direct_phone_cnt
## 1         1 59.0         9.666667       39.000000        0.1666667
## 2         2 38.0         6.166667        6.333333        0.0000000
## 3         3 57.0         1.333333        1.500000        0.3333333
## 4         4 45.0         2.833333        8.166667        0.3333333
## 5         5 35.5         2.333333       40.500000        1.6666667
## 6         6 65.0        10.666667       22.833333        0.0000000
##   atm_withdrawls_cnt direct_mail_cnt mortgage_flag direct_email_cnt
## 1          5.1666667       0.8333333      1.000000       12.0000000
## 2          1.5000000       0.0000000      1.000000        9.6666667
## 3          0.6666667       1.6666667      2.000000        0.1666667
## 4          4.8333333       0.5000000      1.000000        4.3333333
## 5          0.5000000       0.6666667      1.000000        5.6666667
## 6          0.1666667       1.1666667      1.833333        0.5000000
```

```
head(daily_interactions_WF_4merge)
```

```
##   masked_id     Date                            Des1
## 1         1 07032016                  Advance Reversal
## 2         2 07312016 CIVSALES_PMA_ADD/REMOVE_OWNERS
## 3         3 07072016             CIVSALES_TAB_CLICKER
## 4         4 08132016                        CNA Added
## 5         5 07242016                Credit Adjustment
## 6         6 07072016      CIVSALES_CUSTOMER_SESSION
##                                           Des2                      Des3
## 1 Customer to Customer Relationship Maintained    POS Preauthorization
## 2                                IRA REPORTING Returned Deposited Item
## 3             LIGHT_INTERFACE_WITH_CIVSERVICE     Reverse Fee Request
## 4                                       M/G/F      SCHEDULED TRANSFER
## 5                               ODP Maintenance       STATE_RESTRICTION
## 6          Incompleted Transaction - Cust Init     REPORT_STOP_PAYMENT
```

Now we are ready to merge them together because each masked_id is only once :)

```
merged_data <- merge(month_end_balances_4merge,daily_interactions_WF_4merge,by="masked_id")
summary(merged_data)
```

```
##    masked_id          age        branch_visit_cnt online_bank_cnt
##  Min.   : 1.00   Min.   :13.83   Min.   : 0.000   Min.   :  0.000
##  1st Qu.:13.25   1st Qu.:34.75   1st Qu.: 1.375   1st Qu.:  3.000
##  Median :25.50   Median :50.92   Median : 5.417   Median :  8.833
##  Mean   :25.50   Mean   :50.46   Mean   : 5.723   Mean   : 20.337
##  3rd Qu.:37.75   3rd Qu.:65.00   3rd Qu.: 9.583   3rd Qu.: 28.917
##  Max.   :50.00   Max.   :89.50   Max.   :17.333   Max.   :140.667
##
##  direct_phone_cnt atm_withdrawls_cnt direct_mail_cnt  mortgage_flag
##  Min.   :0.0000   Min.   : 0.000     Min.   :0.0000   Min.   :1.000
##  1st Qu.:0.0000   1st Qu.: 0.000     1st Qu.:0.0000   1st Qu.:1.000
##  Median :0.1667   Median : 0.500     Median :0.3333   Median :1.000
##  Mean   :0.4067   Mean   : 2.223     Mean   :0.5767   Mean   :1.217
##  3rd Qu.:0.5000   3rd Qu.: 2.208     3rd Qu.:0.6667   3rd Qu.:1.000
##  Max.   :2.1667   Max.   :26.000     Max.   :4.1667   Max.   :2.000
##
##  direct_email_cnt       Date
##  Min.   : 0.000   07022016: 4
##  1st Qu.: 0.500   07012016: 3
##  Median : 1.167   07072016: 3
##  Mean   : 3.813   07112016: 3
##  3rd Qu.: 6.917   07132016: 3
##  Max.   :13.000   07212016: 3
##                   (Other) :31
##                                                  Des1
##  Correction                                        : 3
##  Verify WFB Funds                                  : 3
##  Account Open                                      : 2
##  ACH Prenote                                       : 2
##  Business Deposit and Credit flow selected         : 2
##  Business  Deposit Only - Special Relationship flow: 2
##  (Other)                                           :36
##                          Des2                            Des3
##  NEW_ACCT_OWNERS            : 3   SOTA                     : 3
##  Withdrawal Reversal        : 3   TRAVEL PLAN MAINTENANCE   : 3
```

```
##  Customer Maintenance Income: 2   PHOTOCOPY IMAGE RETRIEVAL : 2
##  Customer Name Change       : 2   PMA_MAINTENANCE           : 2
##  DUPLICATE_TIN_OVERRIDE     : 2   REPORT_ACCOUNT_CLOSE      : 2
##  EFORM ACCESS               : 2   REPORT WORKFLOW STATISTICS: 2
##  (Other)                    :36   (Other)                   :36
```

Done! We can look at a single user:

```
merged_data[1,]
```

```
##   masked_id age branch_visit_cnt online_bank_cnt direct_phone_cnt
## 1         1  59         9.666667              39        0.1666667
##   atm_withdrawls_cnt direct_mail_cnt mortgage_flag direct_email_cnt
## 1           5.166667       0.8333333             1               12
##       Date             Des1                                      Des2
## 1 07032016 Advance Reversal Customer to Customer Relationship Maintained
##                   Des3
## 1 POS Preauthorization
```

So the only real question when it comes to merge is how do you aggregate or summarize the data so you can match things up one to one.