CAC TO CLV, HISTORIC LIFETIME VALUE, AND LIFECYCLE GRIDS

Step 1 – load the dataframe and libraries needed for analysis. For CLV we need three critical measures - Recency, Frequency and Monetary often simplified under the label RFM. These RFM parameters help us cluster customers into groups (segments) so that we may better understand their behavior and make strategic decisions on how to handle each segment. Any ordering system would collect customer id, the order dates and the order amounts. This data represents all the data we need to get started on CLV. The dataset used here (cacclvdataset.csv) has the above information on a 300 customers who have ordered multiple times from the catalog of a specialty ergonomics store that only sells one product – a standing desk that comes in three variations (product a, b, and c). The customer acquisition cost (CAC) is retrieved from a customer relationship management (CRM) software that happens to be Marketo in this case. The CAC is tagged to each customerID which allows us to know how much was spent on each customer. Looking at the summary data it appears the company spent between 10-15 dollars to acquire each customer. For simplicity, we are setting the gross margin for each product a, b and c to be 1, 2 and 3 dollars respectively. Therefore we make a gross margin of 1 dollar for product a, 2 dollars for product b and 3 for product c.

Step 1 – Load the libraries and the data

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(reshape2)
library(ggplot2)

orders <- read.csv ("cacclvdataset.csv") # load the file cacclvdataset.csv

cac <- read.csv ("cac.csv") # load the file cac.csv

gr.margin <- read.csv("grossmargin.csv") # load the file grossmargin.csv
```

Step 2 – get the maximum date from the dataset and set the current date to one day past that day. The data may be in another format, so use the appropriate adjustments to convert from characters to date or date to characters as necessary. You may just simply print out the maximum date and then set today to be one day after if the column is in the right format. Be careful to match the formatting of the dates as different countries use different notations. In our dataset, the date has the month followed by day and then year. So, as a first step I am converting the date into a standard date format.

```
orders$orderdate <- as.Date(orders$orderdate, format = "%m/%d/%Y")

max(orders$orderdate) # this happens to be "2012-04-10"
```

```
## [1] "2012-04-10"
```

```
today <- as.Date('2012-04-11', format='%Y-%m-%d') # set from the line above
```

Step 3 – start calculating the customer lifetime value by merging the information from the gross margin and customer acquisition cost datasets into one dataset. The calculate CLV for each clientId by summing the gross margin for each item for each order placed by the clientId. This is the historic CLV value for each clientId as it stands based on the today variable we just created. In this case it happens to be Apr 10 2012.

```
orders <- merge(orders, gr.margin, by='product') #merge orders & gross margin

clv <- orders %>% group_by(clientId) %>% summarise(clv=sum(grossmarg)) %>% ungroup()
```

Step 4 – convert the data we have into a recency, frequency and monetary type dataset.

```
orders <- dcast(orders, orderId + clientId + gender + orderdate ~ product, value.var='product', fun.agg

orders <- orders %>%
     group_by(clientId) %>%
     mutate(frequency=n(),
            recency=as.numeric(today-orderdate)) %>%
     filter(orderdate==max(orderdate)) %>%
```

```
    filter(orderId==max(orderId)) %>%
    ungroup()
```

Step 5 – In the prior exercise we used a k-means to cluster the RFM parameters, here we just want to segment data based on frequency of ordering (2,3,4,5, > 5) and recency of ordering (less than 1 week, 14-19 days, 20-45 days, 46-80 days and > 80 days).

```
orders.segm <- orders %>%
    mutate(segm.freq=ifelse(between(frequency, 1, 1), '1',
            ifelse(between(frequency, 2, 2), '2',
                ifelse(between(frequency, 3, 3), '3',
                   ifelse(between(frequency, 4, 4), '4',
                       ifelse(between(frequency, 5, 5), '5', '>5'))))) %>%
    mutate(segm.rec=ifelse(between(recency, 0, 6), '0-6 days',
        ifelse(between(recency, 7, 13), '7-13 days',
          ifelse(between(recency, 14, 19), '14-19 days',
           ifelse(between(recency, 20, 45), '20-45 days',
            ifelse(between(recency, 46, 80), '46-80 days', '>80 days'))))) %>%
mutate(cart=paste(ifelse(a!=0, 'a', ''),
                  ifelse(b!=0, 'b', ''),
                  ifelse(c!=0, 'c', ''), sep='')) %>%
arrange(clientId)
 # creating last cart feature, ignore errors here if you have missing SKU
# defining order of boundaries
orders.segm$segm.freq <- factor(orders.segm$segm.freq, levels=c('>5', '5', '4', '3', '2', '1'))

orders.segm$segm.rec <- factor(orders.segm$segm.rec, levels=c('>80 days', '46-80 days', '20-45 days', '
```

Step 5 – we need to merge orders.segm with the CAC and CLV data, and combine the data with the segments. We will calculate total CAC and CLV to date, as well as their average with the following code

```
orders.segm <- merge(orders.segm, cac, by='clientId')
orders.segm <- merge(orders.segm, clv, by='clientId')


lcg.clv <- orders.segm %>%
    group_by(segm.rec, segm.freq) %>%
    summarise(quantity=n(),

cac=sum(cac),
clv=sum(clv)) %>%
    ungroup() %>%
    # calculating CAC and CLV per client
```

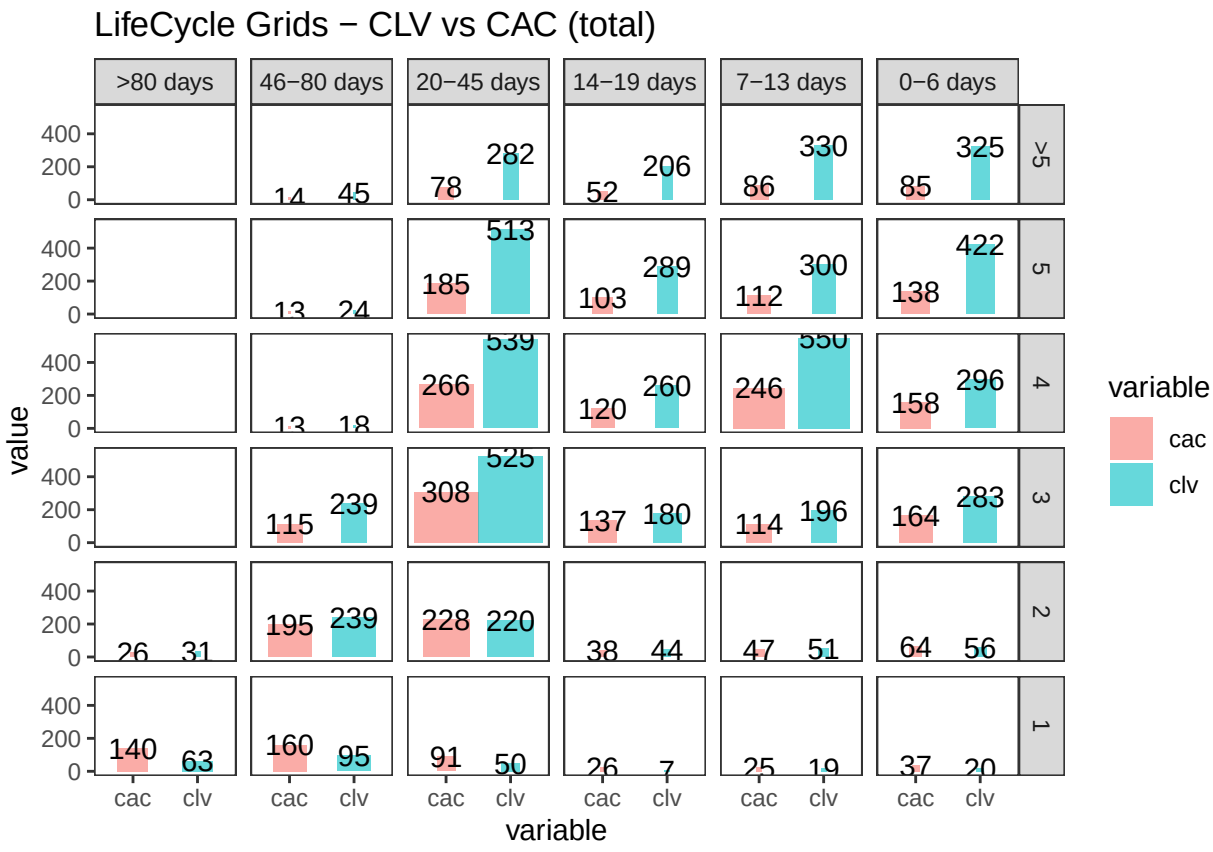
```
     mutate(cac1=round(cac/quantity, 2),
            clv1=round(clv/quantity, 2))

lcg.clv <- melt(lcg.clv, id.vars=c('segm.rec', 'segm.freq', 'quantity'))
```

Step 7 – Plot the two curves to view CLV vs. CAC thus far – one using total (first) and one using average (second)

```
ggplot(lcg.clv[lcg.clv$variable %in% c('clv', 'cac'), ], aes(x=variable, y=value, fill=variable)) +
theme_bw() +
theme(panel.grid = element_blank())+
geom_bar(stat='identity', alpha=0.6, aes(width=quantity/max(quantity))) +
geom_text(aes(y=value, label=value), size=4) +
facet_grid(segm.freq ~ segm.rec) +
ggtitle("LifeCycle Grids - CLV vs CAC (total)")
```

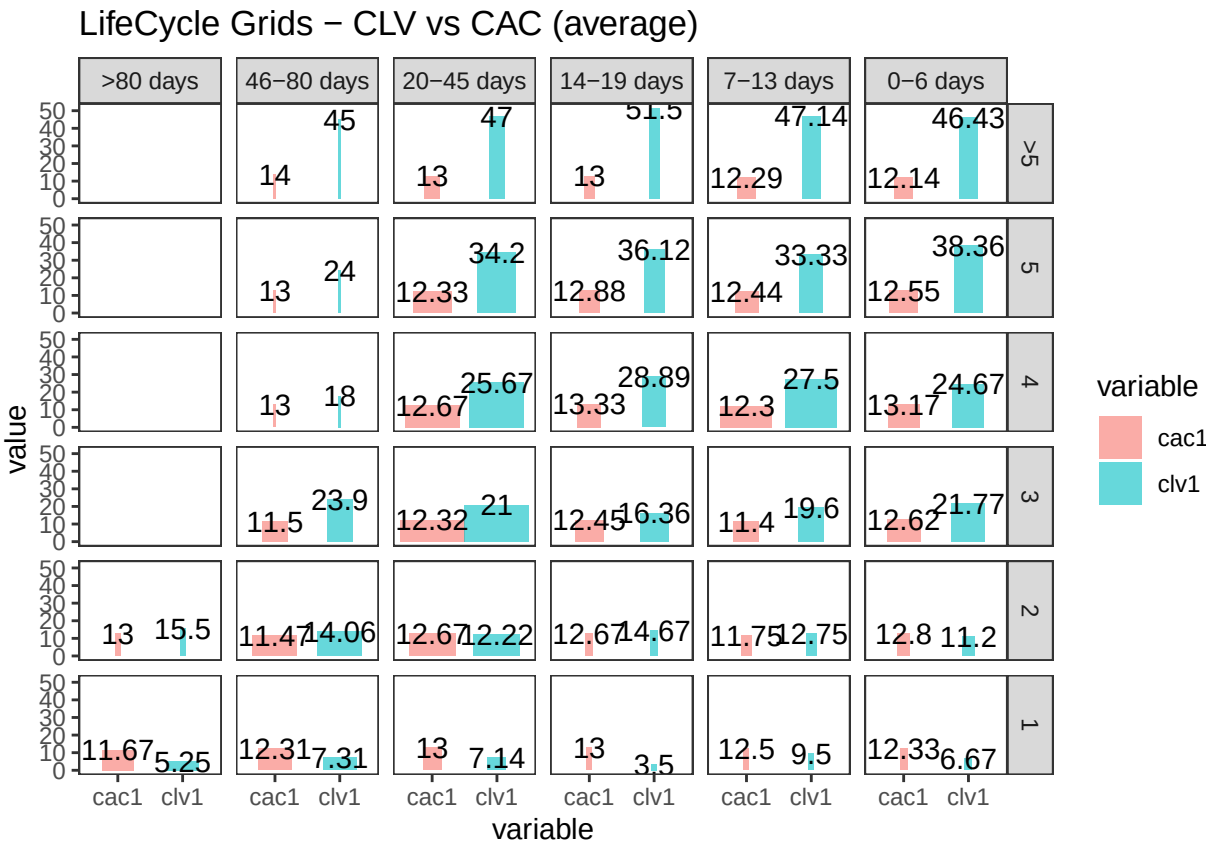
```
## Warning: Ignoring unknown aesthetics: width
```




```
ggplot(lcg.clv[lcg.clv$variable %in% c('clv1', 'cac1'), ], aes(x=variable, y=value, fill=variable)) +
theme_bw() +
theme(panel.grid = element_blank())+
geom_bar(stat='identity', alpha=0.6, aes(width=quantity/max(quantity))) +
geom_text(aes(y=value, label=value), size=4) +
facet_grid(segm.freq ~ segm.rec) +
```

```
ggtitle("LifeCycle Grids - CLV vs CAC (average)")
```

```
## Warning: Ignoring unknown aesthetics: width
```



Interpretation

First to interpret the grid you will need to know that the width of the bars is a representation of the number of customers. Second, you can now see the CAC vs CLV amounts for each block independently. This is better than looking at the entire CLV or CAC totals where different segments are not treated differently. You can now review the difference between CLV to date and CAC and make decisions about your paid campaigns or initiatives

For e.g.

Is it worthwhile to spend the extra money to reanimate some customers such as the ones who only placed one order over 80 days ago?

Should we create an incentive system for the customers in the more than five orders in the 20-45 days cell to make them purchase more?

As I review this, I find the 14-19 day since ordering to be my most profitable customers, should I then ignore this segment since they are doing so well already?