Pentathlon II

Swagata Chakraborty

Setup

Create an rmarkdown document in which you calculate the LTV for each of the plans over a period of 104 weeks using the average subscription churn and revenue numbers provided on belowb. Generate a line plot to compare the five LTV curves. Are your results consistent with your answer to question 3 in the pentathlon-II.pdf file on Dropbox? Why (or why not)?

Hints

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com. Go to http://commonmark.org/help/ for a 10-minute interactive Markdown tutorial

When you click the **Knit** button in Rstudio, a document will be generated that includes both the text you type as well as the output of any embedded R-code chunks within the document.

Analysis

The code chunk below sets the values for the different variables you should use in your analysis. Note that you still have to calculate the weekly discount rate based on the annual discount rate mentioned below.

```
weeks <- 1:104
discount <- .1 ## annual discount rate
discount_weekly <- discount/52
cost <- .6
net_factor <- 0.4


## Pentathlon II (1 email per week)
schurn1 <- .005650      ## subscription churn
rev_sub1 <- .29           ## average revenue from subscribers
rev_unsub1 <- .1225     ## average revenue from non-subscribers


## Pentathlon II (2 emails per week)
schurn2 <- .00755
rev_sub2 <- .34125
rev_unsub2 <- .12375

## Pentathlon II (3 emails per week)
schurn3 <- 0.02345
rev_sub3 <- .35375
rev_unsub3 <- .15

## Pentathlon II (4 emails per week)
schurn4 <- 0.040388
rev_sub4 <- .355
rev_unsub4 <- .1325
```

```
## Pentathlon II (5 emails per week)
schurn5 <- 0.063025
rev_sub5 <- .358750
rev_unsub5 <- .13
```

Create a code chunk that contains a function that you can call to calculate the LTV for each of the 5 email plans.

```
## insert ltv function below
## determine what arguments the function needs to generate the required
## return value
ltv <- function(schurn, rev_sub, rev_unsub) {
  ## weekwise effective subscribers after churning through unsubscription
  subscriber <- rep(100,104)
    for(i in 2:104){
      subscriber[i] <- subscriber[i-1] -  subscriber[i-1]*(schurn)
    }
  ## weekwise effective unsubscribers  being added after churning through unsubscription
  unsubscriber <- rep(0,104)
  total_cust <- rep(100,104)

    for(i in 2:104){
      unsubscriber[i] <- unsubscriber[i-1] +  (subscriber[i-1]- subscriber[i])
      total_cust[i] <- unsubscriber[i] +  subscriber[i]

    }
    ## total customers will stay the same over the weeks as everybody is buying
    ## and churning implies conversion from subscriber to non- subscriber only

    ## profit = (1-0.6)* revenue (respective subscriber or unsubscriber category)
    rev_sub_post_attr <- rev_sub* subscriber
    profit_sub_post_attr <- 0.4*rev_sub_post_attr

    rev_unsub_post_attr <- rev_unsub*unsubscriber
    profit_unsub_post_attr <- 0.4*rev_unsub_post_attr

    ## effective profit after converting to NPV;
    ## weekly discount rate=(annual discount rate)/(52 * 100)
    effective_subs_profit <- rep(0,104)
    for(i in 1:104){
     effective_subs_profit[i] <- profit_sub_post_attr[i] /(1+discount_weekly)**i
    }

    effective_unsubs_profit <- rep(0,104)
    for(i in 1:104){
     effective_unsubs_profit[i] <- profit_unsub_post_attr[i] /(1+discount_weekly)**i
    }

   ## customer overall profit = weekly eff profit from subscribers and unsubscribers.
    total_cust_profit <- effective_subs_profit + effective_unsubs_profit
    total_cust_profit_std <- total_cust_profit/100


   ## Cumulated LTV vector = previos weeks LTV + addtional effective profit expected at the end of
```

```
   ## current week LTV[week i] = LTV[week i-1] + additional value for last week
    LTV_vect <- rep(total_cust_profit_std[1],104)
    for( i in 2:104){
      LTV_vect[i] <- LTV_vect[i-1]+ total_cust_profit_std[i]
    }


    LTV_sub_104 <- sum(effective_subs_profit)
    LTV_unsub_104 <- sum(effective_unsubs_profit)
    LTV_total_104 <- LTV_sub_104 + LTV_unsub_104

    ##For 100 customer --> per customer
    LTV_total_std_104 <- LTV_total_104/100
    print(LTV_total_std_104)
    return(LTV_vect)
}
```

Next, create a tibble (or data.frame) with 6 columns. Five columns for the LTV series for each of the plans and one column for the `weeks` variable defined above. Call the `ltv` function you created above to fill-in the columns.

```
## LTV vectors for 5 different email schemes:
e1 <- ltv(schurn1, rev_sub1, rev_unsub1)
```

```
## [1] 9.442921
```

```
e2 <- ltv(schurn2, rev_sub2, rev_unsub2)
```

```
## [1] 10.42634
```

```
e3 <- ltv(schurn3, rev_sub3, rev_unsub3)
```

```
## [1] 8.639796
```

```
e4 <- ltv(schurn4, rev_sub4, rev_unsub4)
```

```
## [1] 7.071249
```

```
e5 <- ltv(schurn5, rev_sub5, rev_unsub5)
```

```
## [1] 6.304762
```

```
## create the tibble (data.frame)
LTV_Tibble <- data.frame(weeks, e1, e2,e3,e4,e5)

print("First 6 weeks")
```

```
## [1] "First 6 weeks"
```

```
head(LTV_Tibble)
```

```
##   weeks        e1        e2        e3        e4        e5
## 1     1 0.1157774 0.1362380 0.1412284 0.1417274 0.1432246
## 2     2 0.2309554 0.2715602 0.2802819 0.2796021 0.2804296
## 3     3 0.3455381 0.4059745 0.4172129 0.4137826 0.4119989
## 4     4 0.4595295 0.5394888 0.5520724 0.5444209 0.5382917
## 5     5 0.5729334 0.6721109 0.6849102 0.6716624 0.6596438
## 6     6 0.6857539 0.8038484 0.8157748 0.7956466 0.7763691
```

```
print("Last 6 weeks")
```

```
## [1] "Last 6 weeks"
```

```
tail(LTV_Tibble)
```

```
##     weeks       e1       e2       e3       e4       e5
## 99     99 9.086127 10.05654 8.362672 6.847710 6.090504
## 100   100 9.158114 10.13129 8.418604 6.892685 6.133534
## 101   101 9.229785 10.20564 8.474277 6.937525 6.176475
## 102   102 9.301142 10.27960 8.529698 6.982231 6.219326
## 103   103 9.372186 10.35316 8.584869 7.026805 6.262088
## 104   104 9.442921 10.42634 8.639796 7.071249 6.304762
```

Next, generate three line plots that compare the LTV curves for each of the five plans. The first graphs should plot weeks 1 through 8. The second graph should plot weeks 1 through 52 and the final graph should show weeks 1 through 104.

PLOTS

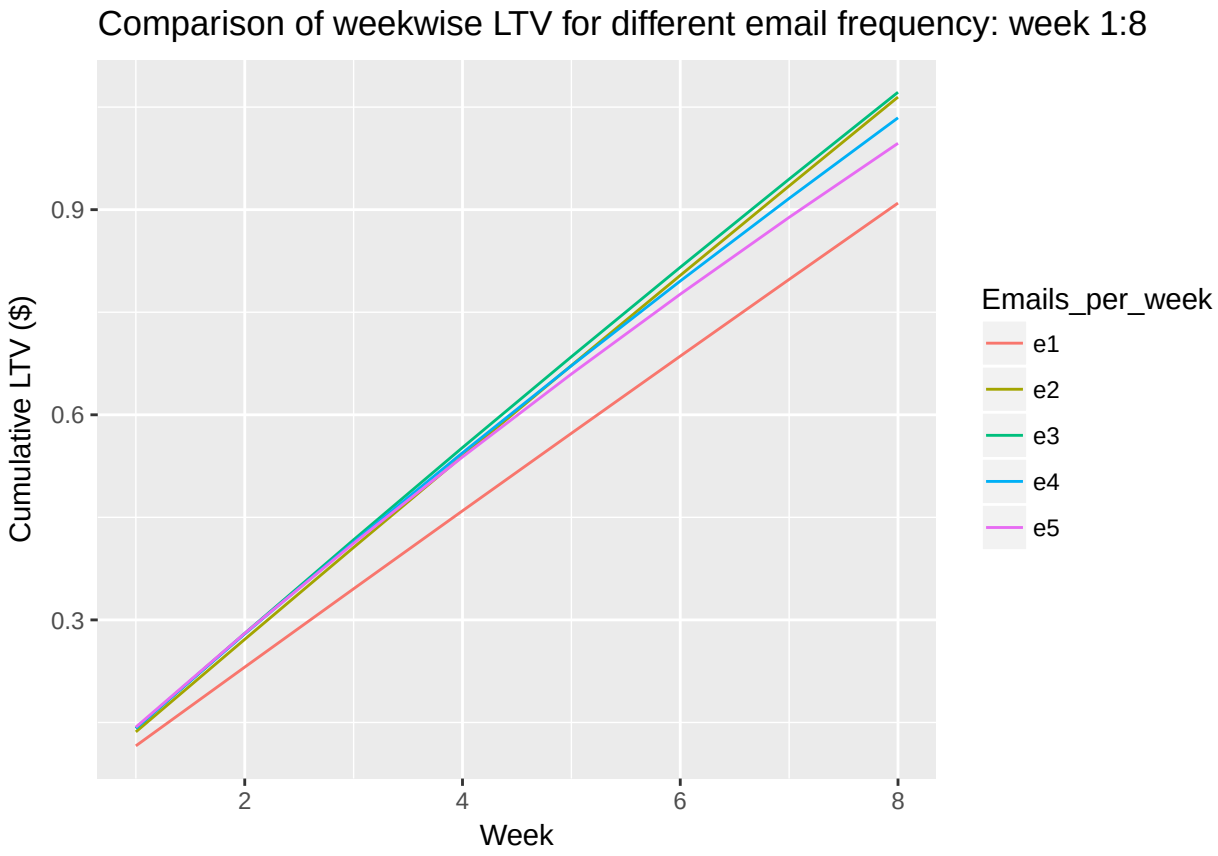
- Comparison of weekwise LTV for different email frequency for week 1 to week 8

```
## Loading tidyverse: tibble
## Loading tidyverse: tidyr
## Loading tidyverse: readr
## Loading tidyverse: purrr
## Loading tidyverse: dplyr
```

```
## Conflicts with tidy packages ---------------------------------------------
```

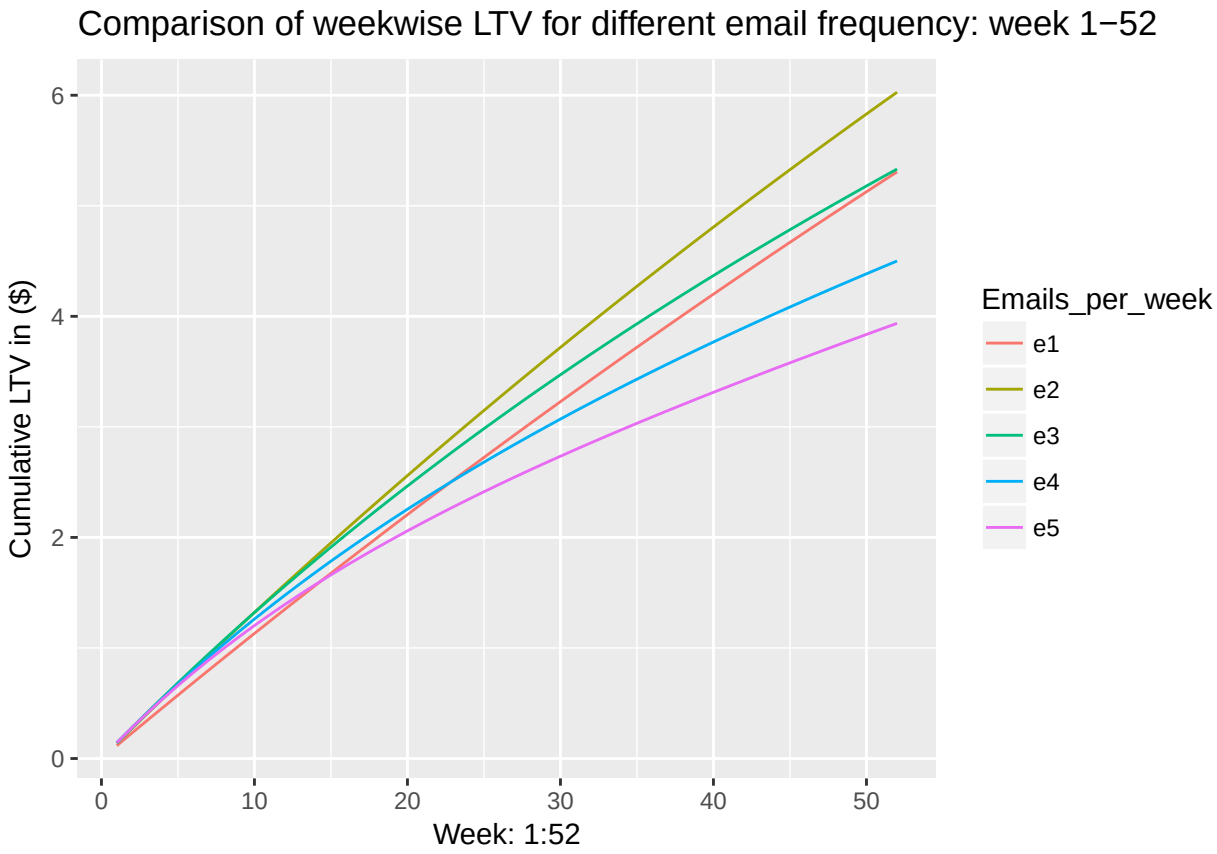
```
## filter(): dplyr, stats
## lag():    dplyr, stats
```

```
## generate graph 1
LTV_gather_8 <- gather(data = LTV_Tibble[1:8,],key = "Email", value = "LTV", -weeks)
colnames(LTV_gather_8) <- c("Week", "Emails_per_week", "Weekwise_LTV")

ggplot(data = LTV_gather_8,mapping = aes(x = Week,y=Weekwise_LTV, color= Emails_per_week))+geom_line()+g
  xlab("Week") + ylab("Cumulative LTV ($)")
```



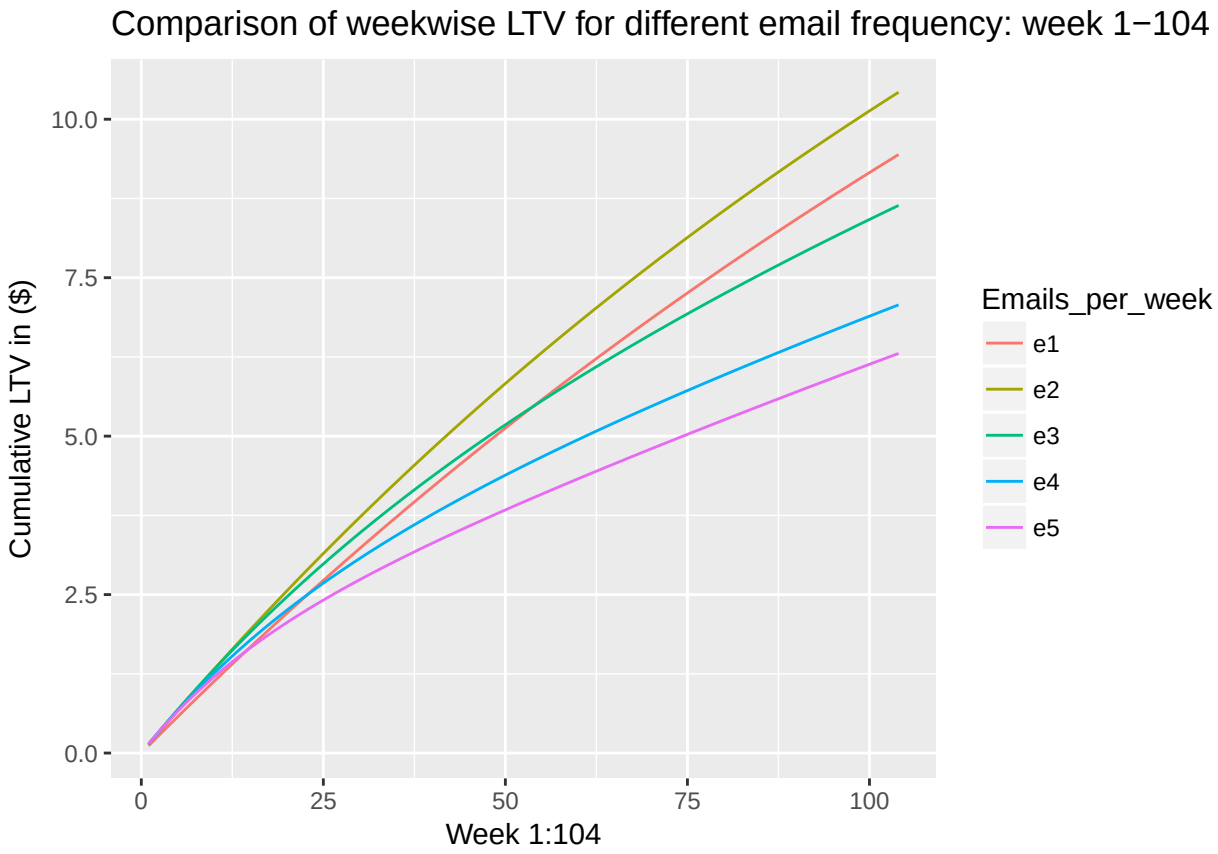
- Comparison of weekwise LTV for different email frequency for week 1 to week 52

```
## generate graph 2
LTV_gather_52 <- gather(data = LTV_Tibble[1:52,],key = "Email", value = "LTV", -weeks)
colnames(LTV_gather_52) <- c("Week", "Emails_per_week", "Weekwise_LTV")

ggplot(data = LTV_gather_52,mapping = aes(x = Week,y=Weekwise_LTV, color= Emails_per_week))+geom_line()+
  xlab("Week: 1:52") + ylab("Cumulative LTV in ($)")
```



Comparison of weekwise LTV for different email frequency for week 1 to week 104

```
## generate graph 3
LTV_gather_104 <- gather(data = LTV_Tibble[1:104,],key = "Email", value = "LTV", -weeks)
colnames(LTV_gather_104) <- c("Week", "Emails_per_week", "Weekwise_LTV")

ggplot(data = LTV_gather_104,mapping = aes(x = Week,y=Weekwise_LTV, color= Emails_per_week))+geom_line()
  xlab("Week 1:104") + ylab("Cumulative LTV in ($)")
```



Please generate a pdf file with your results and upload it to TED together with a document containing your answers to all the questions listed in the pentathlon-II.pdf file on Dropbox.

When you have finished editing this rmarkdown document make sure to save, commit, and push it to GitLab. We will collect all the rmarkdown files from GitLab after the due date.