UNIVERSIDAD DE LA REPÚBLICA

Facultad de Ciencias Económicas y de Administración

Licenciatura en Estadística

Minimización de pérdida de ingresos al aplicar un plan de simplificación comercial

Miranda, Germán Bizoso, Lucas

Noviembre 2023

Trabajo final de Introducción a la Estadística Computacional

Índice

Introducción **3**

Planteo del problema **3**

Aplicación de métodos **6**

Simulación a partir de Uniformes 6

Simulando solamente precios 6

Simulando precios y cantidades de planes 8

Simulaciones con distribuciones Gamma 9

Conclusiones **10**

Anexo **11**

Paquetes utilizados 11

Script 13

Introducción

Normalmente, cada cierto período de tiempo (mes, semestre, año, etc.) las empresas tienden a actualizar los productos y/o servicios que ofrecen, ya sea por mejoras en los beneficios que brindan, por innovación tecnológica de los mismos, por adaptarse a lo que ofrece el mercado en ese momento, entre otros.

En este contexto, habitualmente sucede que esos productos y/o servicios tienden a acumularse en el catálogo histótico ya que si inicialmente un cliente contrata uno de los productos, este, por contrato, deberá mantenerse hasta que el mismo cese ya sea por voluntad del cliente o decisión unilateral de la empresa.

A simple vista, esto no siempre puede ser un problema, pero en empresas de gran tamaño y de rubro de servicios si puede serlo, por varias razones:

- En caso de mejoras en los nuevos productos, esto supone que los clientes más antigüos, gocen de menos beneficios que los nuevos por el mismo precio. Esto también podría provocar insatisfacción, generando pérdidas de clientes que busquen otras opciones fuera de la compañía.
- Puede suponer un costo de mantenimiento ya que los servicios más antigüos también necesitan cierta actualización, como por ejemplo de precios, etiquetas de facturación, etc., así como también un costo de infraestructura para almacenar la infomación de esos productos que se van acumulando.

Por estos motivos, es habitual que cada cierto período se realicen maniobras de simplificación para reducir la cantidad de servicios activos en los sistemas y también para brindar estas versiones mejoradas a los cientes que quedan desactualizados.

Planteo del problema

Esta maniobra de simplificación puede suponer un riesgo, ya que los clientes contractualmente no pueden abonar más de lo acordado, dejando que la opción sea abonar lo mismo o menos. Dado este problema, se asume que siempre la empresa perderá un cierto margen pero el objetivo en el que ahondaremos en el trabajo es poder minimizar lo más posible la pérdida para que no sea un gran impacto.

Para poder resolver este problema, partimos de la situación inicial definiendo cómo se generan los ingresos y de qué variables dependen.

Podemos ver que los ingresos están definidos como:

$$I_{inicial} = \sum_{n_{ini}=1}^{K} P_{inicial_K} Q_{inicial_K}$$

Donde P_i es el precio del servicio i-ésimo, y Q_i es la cantidad de clientes suscriptos a ese servicio.

De las variables de la ecuación, se asume que la cantidad de clientes no va a variar, al menos debido a la ejecución del plan, y que se espera que $I_{inicial}$ sea menor o igual que I_{final}.

Dicho esto, nuestras variables a analizar para poder minimizar la pérdida son los n planes finales y el propio precio P buscando que dicha combinación sea que minimice $I_{final} - I_{inicial}$.

En este proyecto, trabajaremos sobre un total de 50 productos, cada uno con precios distintos y con distinta cantidad de suscriptores, buscando así, primero trabajar sobre un n fijo y variando sólamente P y luego trabajaremos sobre ambas variables y viendo como ambas aportan a lograr el objetivo.

En la siguiente tabla, podemos observar una muestra de los primeros 20 productos con los que trabajaremos. Contamos con el precio de cada uno de los productos, la cantidad de suscriptores y el ingreso final que representan.

Table 1: Top 20 Productos

Nombre_Producto	Precio_Producto	Suscriptores	Ingreso_Por_Plan
Producto 1	268	3162	847416
Producto 2	287	3471	996177
Producto 3	395	535	211325
Producto 4	450	84	37800
Producto 5	462	106	48972
Producto 6	475	6134	2913650
Producto 7	490	254	124460
Producto 8	521	316	164636
Producto 9	550	143	78650
Producto 10	578	2529	1461762
Producto 11	600	19	11400
Producto 12	615	308	189420
Producto 13	670	12	8040
Producto 14	680	65	44200
Producto 15	766	272	208352
Producto 16	790	1222	965380
Producto 17	850	4381	3723850
Producto 18	874	709	619666
Producto 19	954	1595	1521630
Producto 20	1034	49	50666

Con estos datos, podemos armar un gráfico que nos muestre la dispersión de los ingresos por plan, para ver si inicialmente hay alguna relación entre el precio y el ingreso que aportan.

Se puede observar que no hay una relación directa entre los precios de los planes y de los ingresos. Esto se puede deber a que no dependen solamente de su precio, si no de la cantidad de clientes que estén suscriptos.

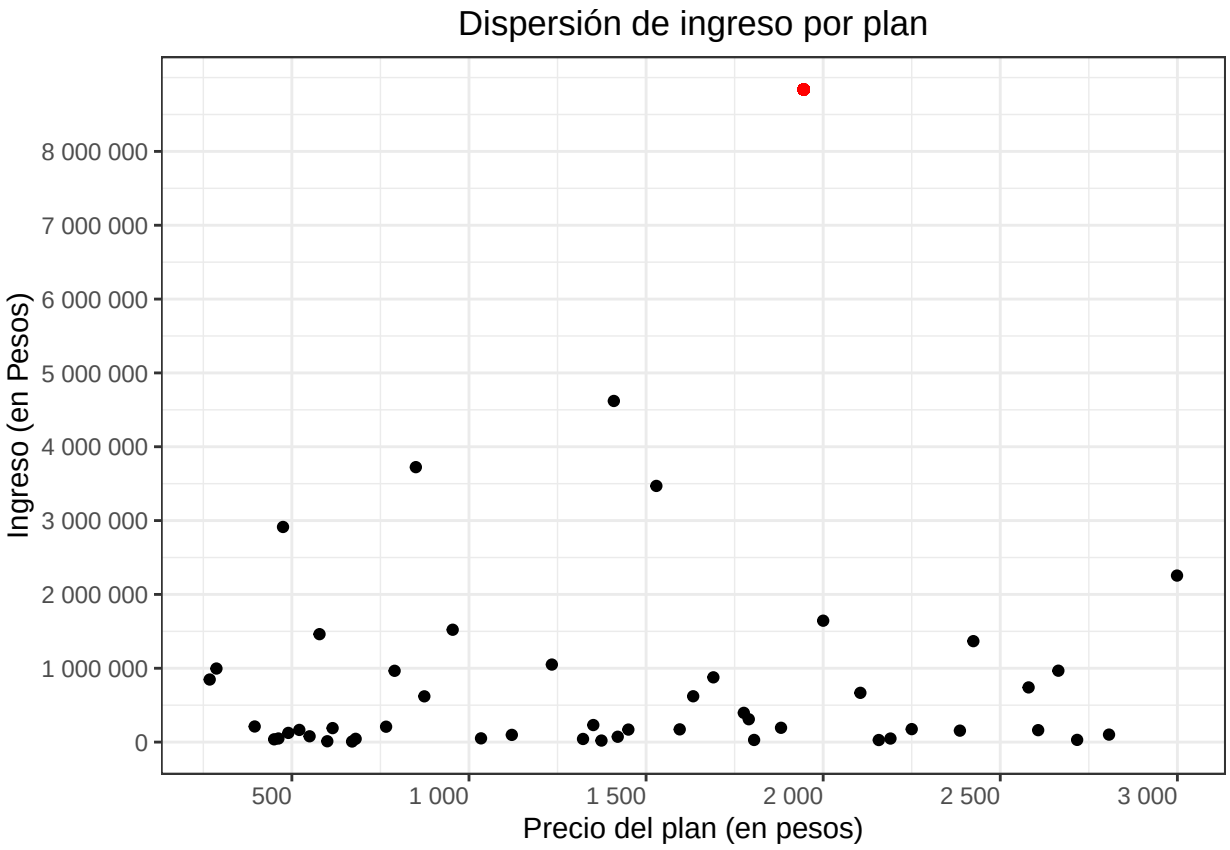


Figure 1: Dispersión de ingresos por plan comercial

Aplicación de métodos

Simulación a partir de Uniformes

Para comenzar a aplicar el método, primero definimos una función "planes_nuv" que genera a partir de distribuciones uniformes, los precios de los planes candidatos, dado un n. En principio buscamos acotar de 50 planes iniciales a 20, por lo que nuestro $n_{inicial} = 20$. En esta función también se puede definir un precio máximo y mínimo para acotar la simulación.

Luego, creamos otra función que aplica la lógica de "precio menor o igual" que se comenta en la introducción. Es decir, en base a los P_{final} generados por la función "planes_nuv", se evalúa que cantidad de clientes $Q_{inicial}$ cumplen con esa condición y serían cantidatos a pasarse a ese nuevo producto. Esta función usa como parámetros los nuevos precios, los precios antiguos y la cantidad de clientes antiguos que hay por precio.

Ejecutamos una primera simulación y obtenemos un primer vector de P_{final} candidato.

```
##  [1]  268  486  513  927 1178 1213 1378 1708 1888 1904 1926 2076 2144 2147 2150
## [16] 2167 2529 2684 2852 2999
```

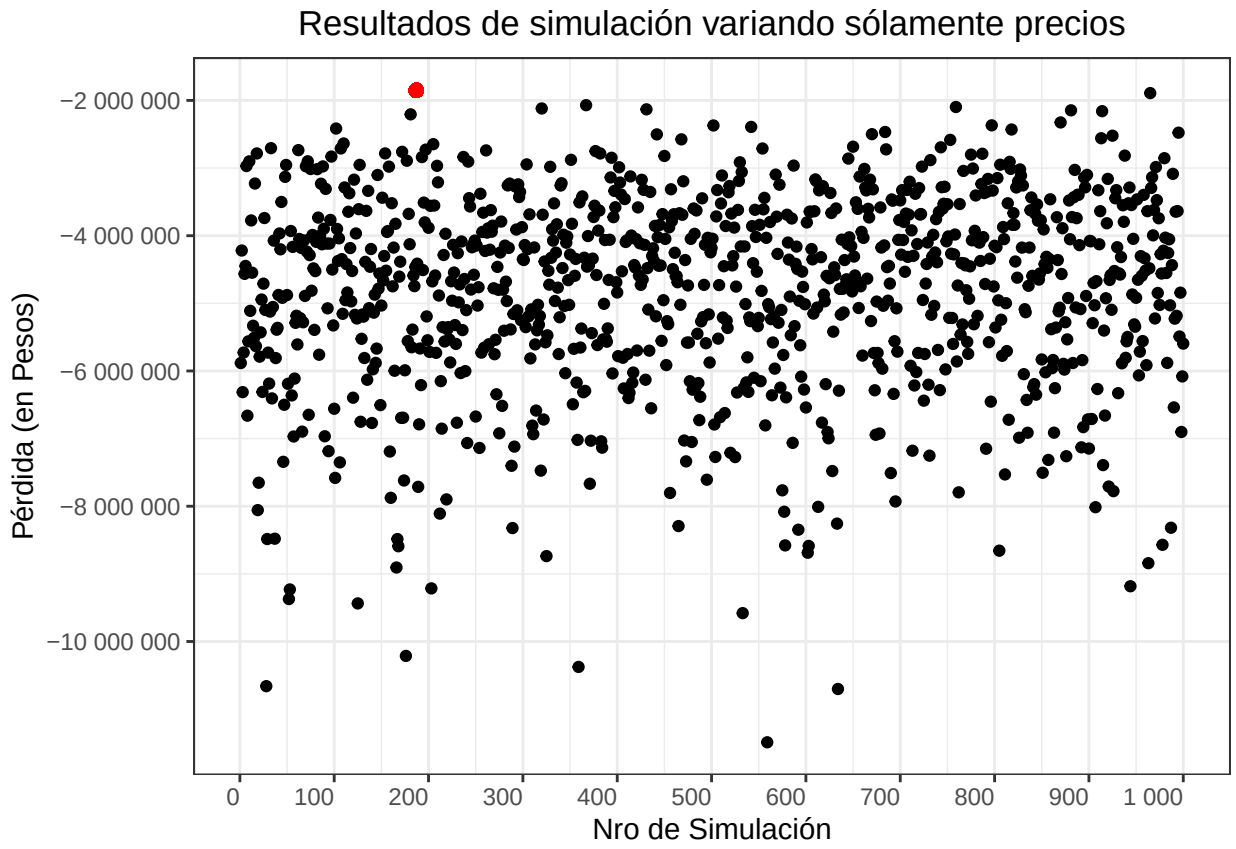
Simulando solamente precios

Para poder ejecutar varias simulaciones con las funciones que creamos en la parte anterior, creamos una nueva función "simul_L" que en base a la cantidad de simulaciones que se quieran hacer (por defecto 1000) y a los parámetros iniciales, nos devuelve una matríz con los $P_{finales}$ y la diferencia de ingreso $I_{final} - I_{inicial}$ que generan. Esa matriz la pasamos a data frame para poder manipular y visualizar mejor los datos. En la Tabla 2 se puede ver una muestra del resultado final, donde tenemos una columna por plan, una fila por simulación.

Table 2: Muestra de la tabla final de resultados

	Producto 1	Producto 2	Producto 19	Producto 20	Diferencia	Flag_Tolerancia
Simul. 1	268	770	2978	2999	-5883023	0
Simul. 2	268	329	2945	2999	-4219217	0
Simul. 3	268	375	2978	2999	-6312917	0
Simul. 4	268	639	2984	2999	-5727751	0
Simul. 5	268	481	2754	2999	-4564507	0
Simul. 6	268	464	2955	2999	-4447405	0

Estos resultados, podemos graficarlos para ver cuál de todas las simulaciones logra obtener la menor diferencia.



En el gráfico podemos ver qué perdida obtuvimos para cada una de las simulaciones de precios y destacado en rojo, la simulación que logró menor diferencia de ingresos, que está dada por el siguiente listado de precios:

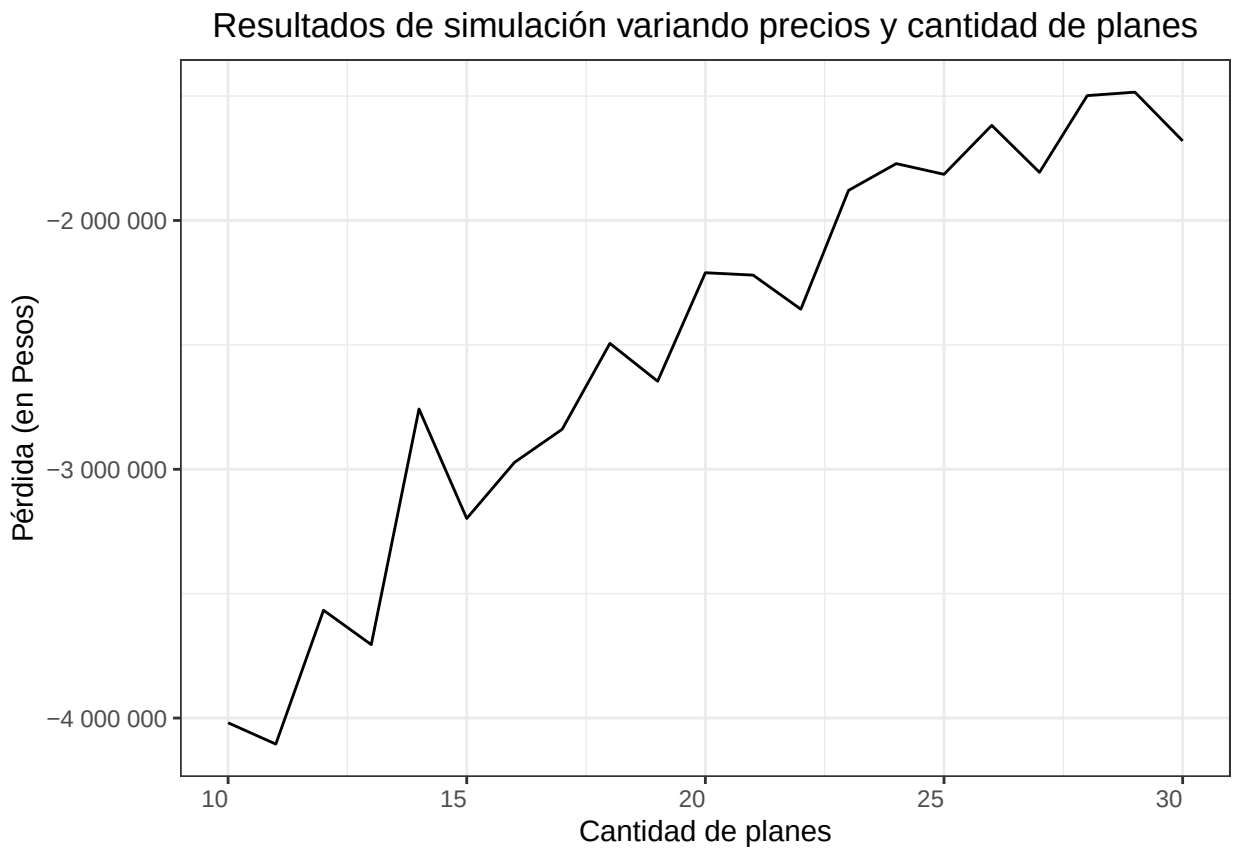
Table 3: Listado de precios para la menor diferencia

	Producto 1	Producto 2	Producto 19	Producto 20	Diferencia	Flag_Tolerancia
Simul. 187	268	444	2943	2999	-1850676	0

Simulando precios y cantidades de planes

Ahora queremos probar simular tanto los precios como la cantidad de planes final, para ver si encontramos la mejor combinación que logre minimizar la pérdida.

Para esto definimos ahora la función "simul_cant" que toma como parámetros, los n entre los que vamos a simular, el listado de precios antiguos, la cantidad de clientes asociadas a cada uno y la cantidad de simulaciones que va a realizar (por defecto 1000).



Simulaciones con distribuciones Gamma

Otro punto que podemos analizar es ver si cambiando las distribuciones que usamos al generar los precios, logramos obtener un resultado distinto, ya que puede que otra distribucuón de adapte mejora a la fijación de precios actuales que hay.

Dicho esto, vamos a probar ahora simular sólamente los precios pero con distribuciones gamma aleatorias.

Table 4: Diferencia de precio según Alfa y Sigma

Alfa	Sigma	Diferencia
0.8	0.09	-2510298
4.8	0.35	-2738946
4.2	0.32	-2793567
4.2	0.28	-2803434
2.6	0.19	-2858642

En la tabla se puede ver que de las simulaciones realizadas, para los valores $\alpha = 2,2$ y $\sigma = 0,27$ logramos obtener la menor diferencia posible de prérdida, logrando también una optimización con este tipo de distribuciones.

Conclusiones

Dada la complejidad del problema planteado que implica una gran cantidad de variables y distintas restricciones para las mismas, los métodos de simulación nos ayudan a poder aproximarnos a un resultado óptimo incluso en estas condiciones.

Comparando los métodos y los distintos resultados, si variamos sólamente precios con uniformes, podemos llegar a obtener entre las tantas simulaciones, un resultado óptimo, aunque sólamente son este método, no logramos obtener alguna relación de las mismas si no que con este método solo intentaríamos conseguir a los $P_{finales}$ que menos pérdida me generen.

En cambio si variamos los precios y a su vez cantidad de planes, vemos una cierta relación de que al aumentar la cantidad de planes, la pérdida disminuye. Esto nos da sentido ya que hay más planes en los que los clientes se "encasillan" logrando así menos diferencia entre los planes viejos y los nuevos. Esto puede ayudar a la hora de fijar el n en caso de que por regla de negocio se quiera restringir.

A su vez, comparando el método de uniformes contra el de distribuciones gamma, se ve que es una mejor forma de poder definir los precios ya que encuentra valores más ajustados a las condiciones planteadas. Si bien no es la mejor forma de fijar los precios ya que los mismos tienen cierta estructuta (enteros y/o redondeados), puede ser un muy buen punto de partida a la hora de tomar una decisión.

Anexo

Paquetes utilizados

```
## To cite package 'tidyverse' in publications use:
##
##   Wickham H, Averick M, Bryan J, Chang W, McGowan LD, François R,
##   Grolemund G, Hayes A, Henry L, Hester J, Kuhn M, Pedersen TL, Miller
##   E, Bache SM, Müller K, Ooms J, Robinson D, Seidel DP, Spinu V,
##   Takahashi K, Vaughan D, Wilke C, Woo K, Yutani H (2019). "Welcome to
##   the tidyverse." _Journal of Open Source Software_, *4*(43), 1686.
##   doi:10.21105/joss.01686 <https://doi.org/10.21105/joss.01686>.
##
## A BibTeX entry for LaTeX users is
##
##   @Article{,
##     title = {Welcome to the {tidyverse}},
##     author = {Hadley Wickham and Mara Averick and Jennifer Bryan and Winston Chang and Lucy D'Agosti
##     year = {2019},
##     journal = {Journal of Open Source Software},
##     volume = {4},
##     number = {43},
##     pages = {1686},
##     doi = {10.21105/joss.01686},
##   }

## To cite package 'here' in publications use:
##
##   Müller K (2020). _here: A Simpler Way to Find Your Files_. R package
##   version 1.0.1, <https://CRAN.R-project.org/package=here>.
##
## A BibTeX entry for LaTeX users is
##
##   @Manual{,
##     title = {here: A Simpler Way to Find Your Files},
##     author = {Kirill Müller},
##     year = {2020},
##     note = {R package version 1.0.1},
##     url = {https://CRAN.R-project.org/package=here},
##   }

## To cite package 'readxl' in publications use:
##
##   Wickham H, Bryan J (2023). _readxl: Read Excel Files_. R package
##   version 1.4.3, <https://CRAN.R-project.org/package=readxl>.
##
## A BibTeX entry for LaTeX users is
##
##   @Manual{,
##     title = {readxl: Read Excel Files},
##     author = {Hadley Wickham and Jennifer Bryan},
##     year = {2023},
##     note = {R package version 1.4.3},
```

```
##     url = {https://CRAN.R-project.org/package=readxl},
##   }

## To cite package 'dplyr' in publications use:
##
##   Wickham H, François R, Henry L, Müller K, Vaughan D (2023). _dplyr: A
##   Grammar of Data Manipulation_. R package version 1.1.4,
##   <https://CRAN.R-project.org/package=dplyr>.
##
## A BibTeX entry for LaTeX users is
##
##   @Manual{,
##     title = {dplyr: A Grammar of Data Manipulation},
##     author = {Hadley Wickham and Romain François and Lionel Henry and Kirill Müller and Davis Vaughan
##     year = {2023},
##     note = {R package version 1.1.4},
##     url = {https://CRAN.R-project.org/package=dplyr},
##   }

## To cite ggplot2 in publications, please use
##
##   H. Wickham. ggplot2: Elegant Graphics for Data Analysis.
##   Springer-Verlag New York, 2016.
##
## A BibTeX entry for LaTeX users is
##
##   @Book{,
##     author = {Hadley Wickham},
##     title = {ggplot2: Elegant Graphics for Data Analysis},
##     publisher = {Springer-Verlag New York},
##     year = {2016},
##     isbn = {978-3-319-24277-4},
##     url = {https://ggplot2.tidyverse.org},
##   }

## To cite package 'scales' in publications use:
##
##   Wickham H, Seidel D (2022). _scales: Scale Functions for
##   Visualization_. R package version 1.2.1,
##   <https://CRAN.R-project.org/package=scales>.
##
## A BibTeX entry for LaTeX users is
##
##   @Manual{,
##     title = {scales: Scale Functions for Visualization},
##     author = {Hadley Wickham and Dana Seidel},
##     year = {2022},
##     note = {R package version 1.2.1},
##     url = {https://CRAN.R-project.org/package=scales},
##   }

## To cite package 'ggeasy' in publications use:
##
```

```
##   Carroll J, Schep A, Sidi J (2023). _ggeasy: Easy Access to 'ggplot2'
##   Commands_. R package version 0.1.4,
##   <https://CRAN.R-project.org/package=ggeasy>.
##
## A BibTeX entry for LaTeX users is
##
##   @Manual{,
##     title = {ggeasy: Easy Access to 'ggplot2' Commands},
##     author = {Jonathan Carroll and Alicia Schep and Jonathan Sidi},
##     year = {2023},
##     note = {R package version 0.1.4},
##     url = {https://CRAN.R-project.org/package=ggeasy},
##   }
```

Script

```
library(tidyverse)
library(here)
library(readxl)
library(dplyr)
library(ggplot2)
library(scales)
library(ggeasy)

db = read_xlsx(here("Entrega Final","Base Productos.xlsx"))

db = db %>% mutate (Ingreso_Por_Plan = Suscriptores*Precio_Producto)   %>%
  arrange(Precio_Producto)

knitr::kable(db %>% slice(1:20) , caption = "Top 20 Productos") %>% print()

db = read_xlsx(here("Entrega Final","Base Productos.xlsx"))

db = db %>% mutate (Ingreso_Por_Plan = Suscriptores*Precio_Producto)   %>%
  arrange(Precio_Producto)

x_max=as.numeric(db %>% filter(Ingreso_Por_Plan==max(db$Ingreso_Por_Plan)) %>% select(Precio_Producto))
y=unlist(db$Ingreso_Por_Plan)
y_max=max(y)
x=unlist(db$Precio_Producto)
m_max=c(x_max,y_max)

pl=ggplot(data=db,aes(x=Precio_Producto,y=Ingreso_Por_Plan))+geom_point()
pl+ geom_point(aes(x=x_max,y=y_max),col="red")+
  theme_bw() + theme(axis.text.x = element_text( hjust = 1))+
  labs(x='Precio del plan (en pesos)',y='Ingreso (en Pesos)',title = "Dispersión de ingreso por plan") +
  scale_y_continuous(label=label_number(),breaks = c(seq(0,8000000,1000000)))+
  scale_x_continuous(label=label_number(),breaks = c(seq(0,3000,500)))+
  ggeasy::easy_center_title()

planes_nuv <- function(n_final,min=200,max=3000) {
  p_nuevos_L = sort(rdunif(n_final-2,min,max))
```

```
  p_nuevos_L = c(min,p_nuevos_L,max)
  return(p_nuevos_L)
}


# new_pc = vector planes nuevos, old_p = vector planes viejos, old_cli = vector actuales

ganancia_nuv <- function(new_p,old_p,old_cli){
  # nueva distribución de los clientes
  new_cli=rep(0,length(new_p))

  for(i in length(old_cli):1) {
    j = length(new_p)
    while(old_p[i]<=new_p[j] && j!=1){
      j=j-1
    }
    new_cli[j] = new_cli[j] + old_cli[i]
  }
  # nuevas ganacias
  new_ingresos <- sum(new_p*new_cli)
  # viejas ganancias
  old_ingresos <- sum(old_p*old_cli)
  #diferencia (asumiendo old_ingresos > new_ingresos)
  diferencia = new_ingresos- old_ingresos
  res <- list(new_p = new_p,
              new_cli = new_cli ,
              new_ingresos = new_ingresos ,
              old_ingresos = old_ingresos ,
              diferencia = diferencia,
              porc_dif = (diferencia)/old_ingresos)
  return(res)
}
# devuelve: - distribucion nueva de los clientes
#           - cuanto es la ganancia del plan nuevo
#           - la ganancia del plan viejo
#           - la diferencia entre ambas
#           - porcentaje de diferencia de los ingresos

p_actuales=unlist(db$Precio_Producto)
q_actuales=unlist(db$Suscriptores)
n=20
min=min(p_actuales)
max=max(p_actuales)

new_p=planes_nuv(n,min=min,max=max)

vec=ganancia_nuv(new_p = new_p,old_p = p_actuales,old_cli=q_actuales)
vec$new_p

simul_L <- function(n_final,min,max,old_p,old_cli,tol,cant_sim = 1000){
  # usa planes_nuv y ganancia_nuv
  # tol valor que hay que superar para ser aceptada la nueva distribución
  # cant_sim cantidad de simulaciones
```

```
  all <- matrix(0 , nrow = cant_sim , ncol = (n_final+2))
  # primeras n_final+2 columnas son los 20 planes
  # penúltima columna es la diferencia con los ingresos viejos
  # última columna es un T/F según se alcance la tolerancia
  for (i in 1:cant_sim) {
    plan_i <- planes_nuv(n_final,min,max)
    gan_i <- ganancia_nuv(plan_i,old_p,old_cli)
    t_f <- as.numeric(gan_i$diferencia>tol)
    vector <- c(plan_i,gan_i$diferencia,t_f)
    all[i,] <- vector
  }
  return(all)
}

n_final=20
res_sim <- simul_L(n_final=n_final,min=min,max=max,p_actuales,q_actuales,tol=-600000)

col_names = c(paste('Producto',seq(1,n_final,1)),'Diferencia','Flag_Tolerancia')
row_names = c(paste('Simul.',seq(1,nrow(res_sim),1)))

df = as.data.frame(res_sim)
colnames(df)=col_names
rownames(df)=row_names

tbl=head(df) %>% select('Producto 1','Producto 2','Producto 19','Producto 20','Diferencia','Flag_Toleran
knitr::kable(tbl , caption = "Muestra de la tabla final de resultados") %>% print()

max = df %>% mutate(Nro_Sim=cur_group_rows()) %>%
  filter(Diferencia == max(Diferencia))%>%
  select('Nro_Sim','Diferencia')
y_max = as.numeric( max %>% select('Diferencia'))
x_max = as.numeric( max%>% select('Nro_Sim'))

pl=ggplot(data=df,aes(x=1:1000,y=Diferencia))+geom_point()
pl+ geom_point(aes(x=x_max,y=y_max),col="red",size=2)+
  theme_bw() + theme(axis.text.x = element_text( hjust = 1))+
  labs(x='Nro de Simulación',y='Pérdida (en Pesos)',title = "Resultados de simulación variando sólamente
  scale_y_continuous(label=label_number(),breaks = c(seq(0,-20000000,-2000000)))+
  scale_x_continuous(label=label_number(),breaks = c(seq(0,1000,100)))+
  ggeasy::easy_center_title()


tbl=df %>% filter(Diferencia == max(Diferencia)) %>% select('Producto 1','Producto 2','Producto 19','Pr
knitr::kable(tbl , caption = "Listado de precios para la menor diferencia") %>% print


simu_cant <- function(n_start,n_end,min,max,old_p,old_cli,tol,cant_sim = 1000) {
  mat <- matrix(0, nrow = n_end-n_start+1 , ncol = 4)
  for(n in n_start:n_end){
    datos = simul_L(n,min,max,old_p,old_cli,tol)
    mat[(n-n_start+1),] = c(n,min(datos[,n+1]),max(datos[,n+1]),mean(datos[,n+1]))
  }
  return(mat)
```

```
}


mat <- simu_cant(10,30,200,3000,p_actuales,q_actuales,1000)

col_names = c('Cant_Planes','Min_Perdida','Max_Perdida','Prom_Perdida')

df = as.data.frame(mat)
colnames(df)=col_names

ggplot(data=df,aes(x=Cant_Planes,y=Max_Perdida))+geom_line()+
  theme_bw() + theme(axis.text.x = element_text( hjust = 1))+
  labs(x='Cantidad de planes',y='Pérdida (en Pesos)',title = "Resultados de simulación variando precios
  scale_y_continuous(label=label_number(),breaks = c(seq(0,-20000000,-1000000)))+
  scale_x_continuous(label=label_number(),breaks = c(seq(10,30,5)))+
  ggeasy::easy_center_title()


# El enfoque es ya con la cantidad de planes

min_dif <- function(tabla_planes){
    posicion <- which(res_sim == max(t(tabla_planes[,length(tabla_planes[1,])-1])),arr.ind = TRUE)
    plan <- tabla_planes[posicion[1,1],]
}

simul_gam <- function(min,max,old_p,old_cli,alfa_m = 0,alfa_M = 20,by_alfa=0.2,sig_m=0.01,sig_M = 10,by
  # min es el precio mínimo
  # max es el precio máximo
  # old_p es el vector de precios anterior
  # old_cli es la cantidad de clientes en los precios viejos
  # alfa_m es el primer valor de las secuencia de alfas, alfa_M es el último y by_alfa es de a cuanto v
  # sig_m es el primer valor de las secuencia de sigmas, sig_M es el último y by_sig es de a cuanto va
  # cant_sim es la cantidad de simulaciones por cada combinación de parámetros
  # n_pla la cantidad de planes nuevos (dos de eso son el min y el max)
r <- 1
s_a <- seq(alfa_m,alfa_M,by=by_alfa)
s_s <- seq(sig_m,sig_M,by=by_sig)
all_g <- matrix(0,nrow = (length(s_a)*length(s_s)),ncol = n_pla+3)
for(i in s_a){
    for(j in s_s){
        x <- rgamma(n_pla-2,i,rate = j)*100
        x <- c(min,x,max)
        x = x[order(x)]
        new_cli=rep(0,n_pla)
        for(l in length(old_cli):1) {
            p = n_pla
            while(old_p[l]<x[p] && p!=1){
                p=p-1
            }
            new_cli[p] = new_cli[p] + old_cli[l]
        }
        new_ingresos <- sum(x*new_cli)
        old_ingresos <- sum(old_p*old_cli)
```

```
        diferencia = new_ingresos- old_ingresos
        if(all_g[r,23]==0 || all_g[r,23] > diferencia){
            all_g[r,] <- c(i,j,x,diferencia)
        }
        r=r+1
    }
}
return(all_g)
}

resul <- simul_gam(200,3000,p_actuales,q_actuales)

resul <- resul[order(resul[,23],decreasing = TRUE),]


best_unif <- res_sim[which(res_sim[,21]==max(res_sim[,21])),]
bests <- matrix(NA,nrow = 6,ncol = 23)
bests[1:5,] <- resul[1:5,]
bests[6,3:23] <- best_unif[1:21]

df = as.data.frame(bests)
col_names=c('Alfa','Sigma','Diferencia')

df= df %>% select('V1','V2','V23') %>% filter(V1!=is.nan(V1))

colnames(df)=col_names

knitr::kable(df , caption = "Diferencia de precio según Alfa y Sigma") %>% print()
```