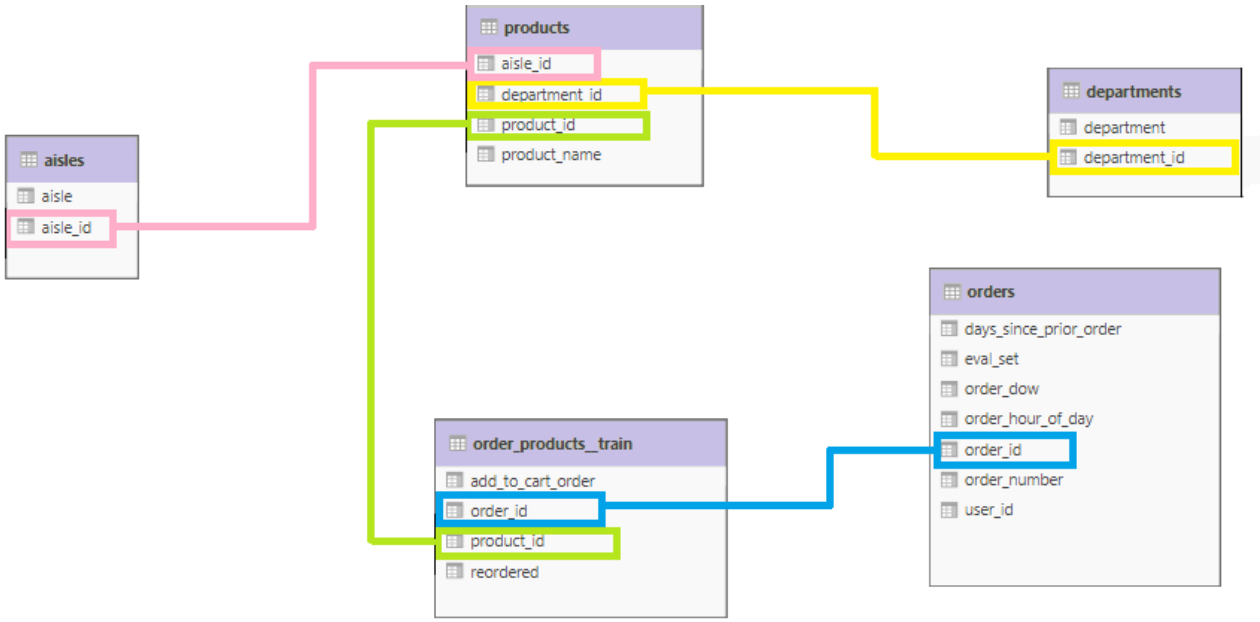
www.nataliejulian.com
Natalie Julian

Market Basket Analysis

Instacart es una empresa que ofrece el servicio de despacho de alimentos en los Estados Unidos y Canadá. Los usuarios piden el despacho a través de su sitio web o de la aplicación móvil. La información de compras y entregas de Instacart se encuentra en las bases de datos `aisles`, `departaments`, `order_products_train`, `orders`, `products`. La información se estructura en un modelo relacional como sigue:



A Instacart le interesa saber cuáles son los productos que suelen despacharse en un mismo pedido, pues así, le será de ayuda para su página web y aplicación que dada la canasta de compra de un cliente le sugiera productos para añadir a su compra.

Cruce de tablas

Las tablas que se poseen entregan la siguiente información:

- `aisles` Entrega clasificación de los productos y el ID de dicha clasificación (ejemplo: *prepared soups salads, bakery desserts, energy granola bars, etcétera*)

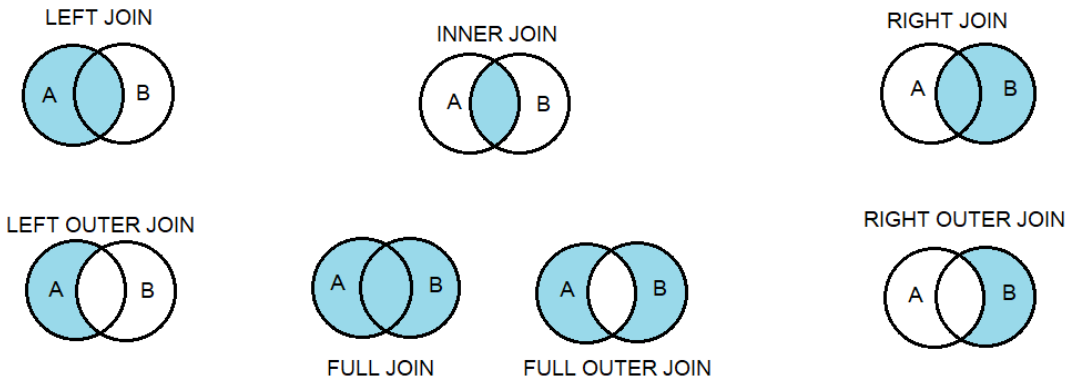
- `products` Contiene información de los productos, nombre de los productos e ID de los productos
- `departments` Indica el departamento que provee el producto (ejemplo: *pets, frozen, bakery, etcétera*)
- `order_products_train` Contiene por orden (pedido) los productos despachados
- `orders` Entrega detalles de la orden (pedido), id del cliente y detalles logísticos de la entrega

Es posible cargar las 5 bases de datos de manera sencilla:

```
#Defino el workspace o directorio

setwd("C:/Users/HP/Desktop/Miner a de Datos/Basket Analysis")


#Obtiene un vector con el nombre de los archivos csv en el directorio

nombres<- list.files(path = getwd() , recursive = TRUE,
                     pattern = "\\.csv$",
                     full.names = FALSE)

nombres
[1] "aisles.csv"                "departments.csv"
[3] "order_products__train.csv" "orders.csv"
[5] "products.csv

Datas<-lapply(nombres,"read_csv")  #Carga todos los archivos csv

names(Datas)<-substr(nombres,1,nchar(nombres)-4) #A ade nombre a cada data

Datas
$aisles
# A tibble: 134 x 2
   aisle_id aisle
      <dbl> <chr>
 1        1 prepared soups salads
 2        2 specialty cheeses
 3        3 energy granola bars
 4        4 instant foods
 5        5 marinades meat preparation
 6        6 other
 7        7 packaged meat
 8        8 bakery desserts
 9        9 pasta sauce
10       10 kitchen supplies
# ... with 124 more rows

$departments
# A tibble: 21 x 2
   department_id department
           <dbl> <chr>
 1             1 frozen
 2             2 other
 3             3 bakery
 4             4 produce
 5             5 alcohol
 6             6 international
 7             7 beverages
 8             8 pets
 9             9 dry goods pasta
10            10 bulk
# ... with 11 more rows

$order_products__train
# A tibble: 1,384,617 x 4
   order_id product_id add_to_cart_order reordered
      <dbl>      <dbl>              <dbl>     <dbl>
 1        1      49302                  1         1
 2        1      11109                  2         1
 3        1      10246                  3         0
 4        1      49683                  4         0
 5        1      43633                  5         1
```

```
 6        1       13176                  6         0
 7        1       47209                  7         0
 8        1       22035                  8         1
 9       36       39612                  1         0
10       36       19660                  2         1
# ... with 1,384,607 more rows

$orders
# A tibble: 3,421,083 x 7
   order_id user_id eval_set order_number order_dow order_hour_of_d¬
      <dbl>   <dbl> <chr>           <dbl>     <dbl> <chr>
 1  2539329       1 prior               1         2 08
 2  2398795       1 prior               2         3 07
 3   473747       1 prior               3         3 12
 4  2254736       1 prior               4         4 07
 5   431534       1 prior               5         4 15
 6  3367565       1 prior               6         2 07
 7   550135       1 prior               7         1 09
 8  3108588       1 prior               8         1 14
 9  2295261       1 prior               9         1 16
10  2550362       1 prior              10         4 08
# ... with 3,421,073 more rows, and 1 more variable:
#   days_since_prior_order <dbl>

$products
# A tibble: 49,688 x 4
   product_id product_name                                aisle_id department_id
        <dbl> <chr>                                          <dbl>         <dbl>
 1          1 Chocolate Sandwich Cookies                        61            19
 2          2 All-Seasons Salt                                 104            13
 3          3 Robust Golden Unsweetened Oolong Tea              94             7
 4          4 Smart Ones Classic Favorites Mini Rigato¬         38             1
 5          5 Green Chile Anytime Sauce                         5            13
 6          6 Dry Nose Oil                                     11            11
 7          7 Pure Coconut Water With Orange                   98             7
 8          8 Cut Russet Potatoes Steam N' Mash                116             1
 9          9 Light Strawberry Blueberry Yogurt                120            16
10         10 Sparkling Orange Juice & Prickly Pear Be¬        115             7
# ... with 49,678 more rows
```

La lista `Datas` contiene las 5 bases de datos (es una lista de bases de datos). Sin embargo, note que las bases de datos que nos interesan para realizar Análisis de Asociación de productos son `order_products_train` (pues es la que indica cuáles productos se pidieron en cada pedido) y `products` pues indica el nombre de los productos. Realizamos el cruce entre estas dos tablas usando como llave `product_id`. Algunos tipos de cruces de tablas son:



Utilizaremos un *left join* pues nos interesa añadir por cada transacción (`order_id`) el

nombre del producto:

```
names(Datas)
[1] "aisles"                "departments"           "order_products__train"
[4] "orders"                "products"


cruce<-left_join(Datas[[3]],Datas[[5]],by="product_id")

print(cruce)
# A tibble: 1,384,617 x 7
   order_id product_id add_to_cart_ord¬ reordered product_name aisle_id
      <dbl>      <dbl>             <dbl>     <dbl> <chr>           <dbl>
 1        1      49302                 1         1 Bulgarian Y¬       120
 2        1      11109                 2         1 Organic 4% ¬       108
 3        1      10246                 3         0 Organic Cel¬        83
 4        1      49683                 4         0 Cucumber Ki¬        83
 5        1      43633                 5         1 Lightly Smo¬        95
 6        1      13176                 6         0 Bag of Orga¬        24
 7        1      47209                 7         0 Organic Has¬        24
 8        1      22035                 8         1 Organic Who¬        21
 9       36      39612                 1         0 Grated Peco¬         2
10       36      19660                 2         1 Spring Water       115
# ... with 1,384,607 more rows, and 1 more variable: department_id <dbl>
```

Market Basket Analysis

La idea de realizar este análisis es descubrir asociaciones entre artículos o productos de modo de establecer comportamientos de compra. Se buscan combinaciones de artículos que son en su mayoría comprados en conjunto, busca responder las siguientes preguntas:

- Dado que compré el producto X, ¿cuáles son los otros productos que probablemente también llevé en esta compra?
- ¿Qué productos se asocian tal que *si llevo X e Y, llevo Z*?

Reglas de Asociación

Miremos los productos que se compraron en algunas transacciones:

```
set.seed(5)

boletas<-sample(cruce$order_id,5)

(compra1<-cruce$product_name[which(cruce$order_id==boletas[1])])
 [1] "Total 2% All Natural Greek Strained Yogurt with Honey"
 [2] "Organic Avocado"
 [3] "Light Chestnut Brown 5N Permanent Hair Color"
 [4] "Organic Garlic"
 [5] "Total 2% Lowfat Plain Greek Yogurt"
 [6] "Organic Raw Agave Nectar"
 [7] "Seedless Red Grapes"
 [8] "Italian Sparkling Mineral Water"
 [9] "Globe Eggplant"
[10] "Jalapeno Peppers"
[11] "Red Vine Tomato"
> (compra2<-cruce$product_name[which(cruce$order_id==boletas[2])])
[1] "Bag of Organic Bananas"           "Large Lemon"
[3] "Organic Italian Salad"            "Organic Italian Parsley Bunch"
[5] "Organic Garnet Sweet Potato (Yam)" "Russet Potato"
```

```
 [7] "Organic Sour Cream"

(compra3<-cruce$product_name[which(cruce$order_id==boletas[3])])
[1] "Yuba Tofu Skin"
[2] "Original Real Vegetable Chips"
[3] "Organic Blue Corn Tortilla Chips"
[4] "Organic Cilantro"
[5] "Organic Sliced Provalone Cheese"
[6] "Organic Traditional Plain Kefir"
[7] "Almonds & Sea Salt in Dark Chocolate"
[8] "Passionberry Bliss Kombucha Drink"
[9] "Sparkling Natural Mineral Water"

(compra4<-cruce$product_name[which(cruce$order_id==boletas[4])])
 [1] "Original No Pulp 100% Florida Orange Juice"
 [2] "Organic Reduced Fat 2% Milk"
 [3] "Banana"
 [4] "Baked Aged White Cheddar Rice and Corn Puffs"
 [5] "VitaminWater Zero XXX Acai Blueberry Pomegranate"
 [6] "Organic Blackberries"
 [7] "Organic Blue Corn Tortilla Chips"
 [8] "Original Hummus"
 [9] "Organic Extra Large Brown Eggs"
[10] "Lowfat Small Curd Cottage Cheese"
[11] "Cucumber & Garlic Tzatziki"
[12] "Organic Wheat-Free & Gluten-Free Original Crackers"
[13] "Organic Red Bell Pepper"

(compra5<-cruce$product_name[which(cruce$order_id==boletas[5])])
 [1] "Organic Garlic"
 [2] "Super Berry Fusion"
 [3] "Snack Sticks Chicken & Rice Recipe Dog Treats"
 [4] "Limes"
 [5] "Organix Chicken & Brown Rice Recipe"
 [6] "Organic Yellow Peaches"
 [7] "Organic Lemon"
 [8] "Hibiscus Organic Raw Kombucha"
 [9] "Organic Yellow Onion"
[10] "Mint Chip Almond Milk Non-Dairy Frozen Dessert"
[11] "Passionberry Bliss Kombucha Drink"
[12] "Organic Du Puy Lentils"
[13] "Veganic Sprouted Brown Rice Cacao Crisps"
[14] "Organic Mango Acai Fruit Leather, 12 Ct"
[15] "Organic Garnet Sweet Potato (Yam)"
[16] "Chocolate Hazelnut Non-Dairy Beverage"
[17] "Guava Goddess Organic Kambucha"
[18] "Organic Mint"
[19] "Organic Rainbow Carrots"
[20] "Organic Italian Parsley Bunch"
[21] "Organic Small Bunch Celery"
[22] "Organic Sunday Bacon"
[23] "Apple & Cherry Fruit Bar"
[24] "Marcona Almonds"
[25] "Dried Mangoes"
[26] "Sea Salt Macadamias"
[27] "Organic Tree Ripened Plums"
[28] "Organic Cinnamon Apple Chips"
[29] "Organic Dill"
[30] "Ccnut Raw Cocoaminos"
[31] "Fruit & Vegetable Wash"
[32] "Blackberry Cucumber Sparkling Water"
```

Y se puede resumir en una tabla:

Boleta	Productos
1	"Sparkling Natural Mineral Water" "Toma" "Banana" "Total 0% Greek Yogurt" "Organic Blackberries" "Organic Strawberries" "Ancient Grain Original Granola"
2	"Gluten Free Whole Grain Bagels" "Gluten Free Whole Grain Bread" "Wild Non-Parel Capers, Sunkissed in the Mediterranean" "Organic Roma Tomato" "Vanilla Coconut Milk Yogurt" ....
3	Boneless Skinless Chicken Breasts" "Macaroni & Cheese" "Russet Potato" "Bartlett Pears" [5] "Banana" "Cream Cheese" "Gluten and Dairy Free Rice Macaroni and Cheeze"...
4	"Chicken Curry Salad" "Organic Large Extra Fancy Fuji Apple" "Large Alfresco Eggs" "Organic Strawberries" "Kids Mac-N-Cheezv Fish Shaped Ravioli" "Kids Sensible Foods Broccoli Littles" "Organic Garlic" ...
5	"Organic Heavy Whipping Cream" "Exquisitely Rich 85% Dark Chocolate" "Curate Melon Pomelo Sparking Water" "Organic Zucchini" "Red Peppers" "Organic Tomato Paste"

De estas boletas podemos definir un conjunto I de productos:

$$I = \{i_1, i_2, ..., i_k\}$$

Utilizando 5 boletas, los productos serían:

```
unique(c(compra1,compra2, compra3,compra4,compra5))
 [1] "Sparkling Natural Mineral Water"
 [2] "Toma"
 [3] "Banana"
 [4] "Total 0% Greek Yogurt"
 [5] "Organic Blackberries"
 [6] "Organic Strawberries"
 [7] "Ancient Grain Original Granola"
 [8] "Gluten Free Whole Grain Bagels"
 [9] "Gluten Free Whole Grain Bread"
[10] "Wild Non-Pareil Capers, Sunkissed in the Mediterranean"
[11] "Organic Roma Tomato"
[12] "Vanilla Coconut Milk Yogurt"
[13] "Wheat Gluten Free Waffles"
[14] "Organic Gluten & Wheat Free Homestyle Waffles"
[15] "Light Sea Salt Plentils"
[16] "Dairy Free Unsweetened Coconut Milk"
[17] "Organic Unsweetened Soy Milk Beverage"
[18] "Organic Apple Juice"
[19] "Organic Chocolate Flavored Syrup"
[20] "Organic Broccoli"
[21] "Organic Celery Hearts"
[22] "Organic Carrot Bunch"
[23] "Apple Cinnamon Instant Oatmeal"
```

```
[24] "Free & Clear All-Purpose Natural Cleaner"
[25] "Italian Sparkling Mineral Water"
[26] "Peppermint"
[27] "Tilapia Filet"
[28] "Organic Large Green Asparagus"
[29] "Boneless Skinless Chicken Breasts"
[30] "Macaroni & Cheese"
[31] "Russet Potato"
[32] "Bartlett Pears"
[33] "Cream Cheese"
[34] "Gluten and Dairy Free Rice Macaroni and Cheeze"
[35] "Organic Agave Nectar"
[36] "Organic Sunflower Seed Spread"
[37] "Wild Blueberry Preserves"
[38] "Gluten Free Cinnamon Rolls"
[39] "Large Lemon"
[40] "Limes"
[41] "Large Yellow Flesh Nectarine"
[42] "Black Plum"
[43] "White Peach"
[44] "Organic Hot House Tomato"
[45] "Chicken Curry Salad"
[46] "Organic Large Extra Fancy Fuji Apple"
[47] "Large Alfresco Eggs"
[48] "Kids Mac-N-Cheezy Fish Shaped Ravioli"
[49] "Kids Sensible Foods Broccoli Littles"
[50] "Organic Garlic"
[51] "Organic Iceberg Lettuce"
[52] "Shredded Mild Cheddar Cheese"
[53] "Organic Pancake Syrup"
[54] "Himalania Fine Pink Salt"
[55] "Organic Chicken & Apple Sausage"
[56] "Fresh Pasta Sauce, Marinara Sauce"
[57] "Organic Gluten-Free Quinoa Spaghetti"
[58] "Organic Greek Plain Lowfat Yogurt"
[59] "Organic Raspberries"
[60] "Cocoa Spread with Hazelnuts"
[61] "Lemon & Eucalyptus Hand Soap"
[62] "Avocados"
[63] "Bag of Organic Bananas"
[64] "Organic Rainbow Carrots"
[65] "Organic Ginger Root"
[66] "Organic Small Bunch Celery"
[67] "Organic Sweet Onion"
[68] "Parsley"
[69] "Organic Tomato Cluster"
[70] "Organic Cucumber"
[71] "Organic Hummus"
[72] "Sour Cream & Onion Potato Chips"
[73] "Double Chocolate Chip Cookies"
[74] "Mini Chocolate Croissant"
[75] "Organic Heavy Whipping Cream"
[76] "Exquisitely Rich 85% Dark Chocolate"
[77] "Curate Melon Pomelo Sparking Water"
[78] "Organic Zucchini"
[79] "Red Peppers"
[80] "Organic Tomato Paste"
```

El conjunto de transacciones/boletas/compras/pedidos se puede definir:

$$T = \{t_1, t_2, ..., t_n\}$$

Por ejemplo, en la boleta/transacción 1 se tiene:

$t_1 = \{\texttt{Sparkling Natural Mineral Water, Toma, Banana, Total 0\% Greek Yogurt, } \ldots\}$

Luego una regla de asociación es $X \Rightarrow Y$, where $X \subset I$, $Y \subset I$ and $X \cap Y = 0$. Por ejemplo, encontrar asociaciones de tipo:

$$\{\texttt{Limes, Red Peppers}\} \Rightarrow \{\texttt{Red Vine Tomato}\}$$

Soporte

$$supp(X, Y) = \frac{|X \cap Y|}{n}$$

Es la cantidad de transacciones en las que aparecen conjuntamente X e Y dividido en el total de transacciones. Veamos si hay productos que se repiten en las 5 boletas seleccionadas:

```
intersect(compra1, compra2)
character(0)

intersect(compra1, compra3)
[1] "Banana"                "Organic Blackberries"

intersect(compra1, compra4)
[1] "Organic Strawberries"

intersect(compra1, compra5)
character(0)

intersect(compra2, compra3)
character(0)

intersect(compra2, compra4)
character(0)

intersect(compra2, compra5)
character(0)

intersect(compra3, compra4)
character(0)

intersect(compra3, compra5)
character(0)
```

Podemos ver que:

$$supp(\texttt{Banana}, \texttt{Organic Blackberries}) = \frac{1}{5} = 20\%$$

Para cualquier otro par de productos el soporte es 0% pues éste es el único par que se repite.

Confianza

La confianza muestra la probabilidad condicional $P(Y|X)$ de que si llevo X lleve Y. Y se define:

$$conf(X \Rightarrow Y) = \frac{supp(X, Y)}{supp(X)}$$

Por ejemplo, Banana se compra en 2 de las 5 transacciones, por lo que:

$$conf(\texttt{Banana} \Rightarrow \texttt{Organic Blackberries}) = \frac{supp(\texttt{Banana}, \texttt{Organic Blackberries})}{supp(\texttt{Banana})} = \frac{1/5}{2/5} = 50\%$$

Lift

El lift o ganancia es la relación entre el soporte observado y el esperado si X e Y si fueran independientes:

$$lift(X \Rightarrow Y) = \frac{supp(X,Y)}{supp(X)supp(Y)}$$

Organic Blackberries se compra en 2 de las 5 transacciones, por lo que:

$$lift(\texttt{Banana} \Rightarrow \texttt{Organic Blackberries}) = \frac{supp(\texttt{Banana}, \texttt{Organic Blackberries})}{supp(\texttt{Banana})supp(\texttt{Organic Blackberries})} = \frac{1/5}{(2/5)(2/5)}$$

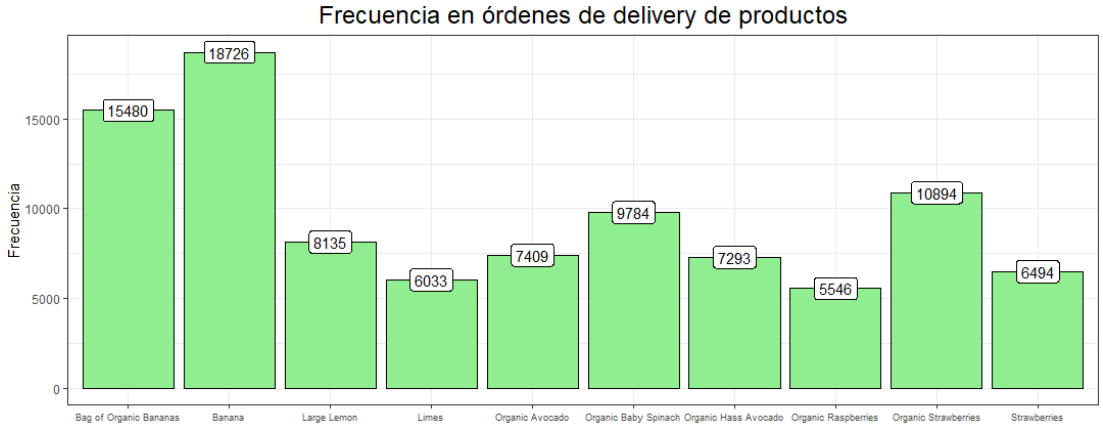
$$lift(\texttt{Banana} \Rightarrow \texttt{Organic Blackberries}) = 1.25\%$$

Si el lift=1 los productos son independientes, no habría asociación. Si lift>1 hay dependencia entre los productos y si lift<1 existe un efecto negativo en la compra de uno versus el otro.

Frecuency plot

Podríamos estar interesados en detectar inicialmente cuáles son los productos que más se piden para despacho:

```
frecuencias<-data.frame(table(cruce$product_name))

fivenum(frecuencias$Freq) #Minimo, 1quantil, Mediana, 3quantile, Maximo
[1]     1     2     5    18 18726

frecuencias<-frecuencias[frecuencias$Freq>5000,]

library(ggplot2)
frecuencias  %>% ggplot(aes(x=Var1, y=Freq)) +
  geom_bar(stat="identity", fill="palegreen2",
           show.legend=FALSE, colour="black") +
  geom_label(aes(label=Freq)) +
  labs(title="Frecuencia en  rdenes  de delivery de productos") +
  theme_bw()+xlab("")+ylab("Frecuencia")+
  theme(axis.text.x.bottom = element_text(size=7),
  plot.title = element_text(hjust=0.5,size=18))
```



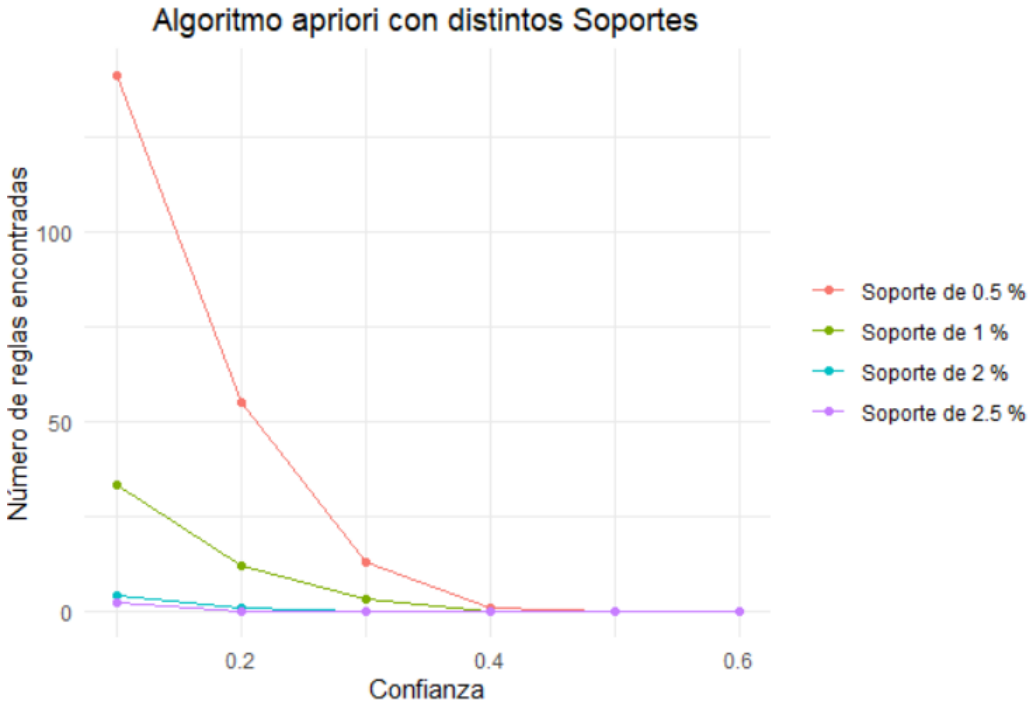
Los productos que más se piden por delivery son vegetales y frutas, Banana es el producto más solicitado, luego fresas, espinaca, palta, etcétera.

Elección del soporte y confianza mínimos

El primer paso para crear un conjunto de reglas de asociación es determinar los umbrales óptimos para el soporte y la confianza. Si establecemos estos valores demasiado bajos, el algoritmo tardará más en ejecutarse y obtendremos muchas reglas (la mayoría de ellas no serán útiles pues dado que el soporte y la confianza son bajos, corresponden a eventos que ventas conjuntas esporádicas, no muy frecuentes). Entonces, ¿qué valores elegimos? Podemos probar diferentes valores de soporte y confianza y ver gráficamente cuántas reglas se generan para cada combinación. En los siguientes gráficos podemos ver el número de reglas generadas con un soporte del 0.05%, 0.1%, 0.2%, 0.3% y confianzas de 60%, 50%, 40%, 30%, 20% y 10%.

```
### Probando umbral de confianza y soporte

supportLevels <- c(0.05, 0.1, 0.20, 0.25)/10
confidenceLevels <- c(0.6, 0.5, 0.4, 0.3, 0.2, 0.1)

rules_sup1 <- rep(NA, length(confidenceLevels))
rules_sup2 <- rep(NA, length(confidenceLevels))
rules_sup3 <- rep(NA, length(confidenceLevels))
rules_sup4 <- rep(NA, length(confidenceLevels))

# Reglas

library(arules)

trans<-data.frame(factor(cruce$order_id),cruce$product_name)

colnames(trans)<-c("Transaction","Item")

basket_data = group_by(cruce, order_id)
basket_data = summarise(basket_data,itens=as.vector(list(product_name)))


trans=as(basket_data$itens, 'transactions')

for (i in 1:length(confidenceLevels)) {

  rules_sup1[i] <- length(apriori(trans, parameter=list(sup=supportLevels[1],
  conf=confidenceLevels[i], target="rules")))

}


for (i in 1:length(confidenceLevels)){

  rules_sup2[i] <- length(apriori(trans, parameter=list(sup=supportLevels[2],
  conf=confidenceLevels[i], target="rules")))

}


for (i in 1:length(confidenceLevels)){

  rules_sup3[i] <- length(apriori(trans, parameter=list(sup=supportLevels[3],
  conf=confidenceLevels[i], target="rules")))

}

for (i in 1:length(confidenceLevels)){

  rules_sup4[i] <- length(apriori(trans, parameter=list(sup=supportLevels[4],
  conf=confidenceLevels[i], target="rules")))

}
```

```
num_rules <- data.frame(rules_sup1, rules_sup2, rules_sup3, rules_sup4,
confidenceLevels)

ggplot(data=num_rules, aes(x=confidenceLevels)) +

  geom_line(aes(y=rules_sup1,colour=paste("Soporte de",supportLevels[1]*100,"%"))) +
  geom_point(aes(y=rules_sup1, colour=paste("Soporte de", supportLevels[1]*100,"%"))) +

  geom_line(aes(y=rules_sup2,colour=paste("Soporte de",supportLevels[2]*100,"%"))) +
  geom_point(aes(y=rules_sup2,colour=paste("Soporte de",supportLevels[2]*100,"%"))) +

  geom_line(aes(y=rules_sup3,colour=paste("Soporte de",supportLevels[3]*100,"%"))) +
  geom_point(aes(y=rules_sup3,colour=paste("Soporte de",supportLevels[3]*100,"%"))) +

  geom_line(aes(y=rules_sup4,colour=paste("Soporte de",supportLevels[4]*100,"%"))) +
  geom_point(aes(y=rules_sup4,colour=paste("Soporte de",supportLevels[4]*100,"%"))) +

  labs(x="Confianza", y="Numero de reglas encontradas",
       title="Algoritmo apriori con distintos Soportes") +
  theme_minimal() +
  theme(legend.title=element_blank(), plot.title = element_text(hjust=0.5))
```



Naturalmente, a menor soporte o menor confianza, mayor número de reglas entre los productos. La elección del soporte dependerá bastante de la cantidad de transacciones que se tenga y de la variedad de productos comprados en dichas transacciones. Note que usando un soporte de 0.5% y una confianza de 0.1 se determinan más de 100 reglas, y la idea es determinar las reglas más útiles. Probemos por ejemplo un soporte del 1% y una confianza de 0.2.

```
rules <- apriori(trans, parameter=list(sup=0.01, conf=0.2, target="rules"))


inspect(sort(rules,by="lift")) #Ordena por ganancia o lift las reglas
    lhs                         rhs                        support
```

```
    [1]  {Limes}                 => {Large Lemon}             0.01215618
    [2]  {Organic Raspberries}   => {Organic Strawberries}    0.01272779
    [3]  {Organic Hass Avocado}  => {Bag of Organic Bananas}  0.01844386
    [4]  {Organic Raspberries}   => {Bag of Organic Bananas}  0.01356614
    [5]  {Organic Hass Avocado}  => {Organic Strawberries}    0.01172938
    [6]  {Organic Strawberries}  => {Bag of Organic Bananas}  0.02342827
    [7]  {Strawberries}          => {Banana}                  0.01484654
    [8]  {Organic Avocado}       => {Banana}                  0.01688909
    [9]  {Organic Baby Spinach}  => {Bag of Organic Bananas}  0.01704151
    [10] {Large Lemon}           => {Banana}                  0.01644704
    [11] {Limes}                 => {Banana}                  0.01014412
    [12] {Organic Baby Spinach}  => {Banana}                  0.01524286
         confidence coverage   lift     count
    [1]  0.2643792  0.04598008 4.264159 1595
    [2]  0.3011179  0.04226844 3.626710 1670
    [3]  0.3318250  0.05558308 2.812560 2420
    [4]  0.3209520  0.04226844 2.720400 1780
    [5]  0.2110243  0.05558308 2.541609 1539
    [6]  0.2821737  0.08302784 2.391714 3074
    [7]  0.2999692  0.04949356 2.101819 1948
    [8]  0.2990957  0.05646716 2.095698 2216
    [9]  0.2285364  0.07456806 1.937082 2236
    [10] 0.2652735  0.06200032 1.858714 2158
    [11] 0.2206199  0.04598008 1.545836 1331
    [12] 0.2044154  0.07456806 1.432294 2000
```

Visualización

```
library(arulesViz) # Visualizacion de reglas de asociacion

plot(rules, method="graph",cex=0.6, main="Lift de Reglas de asociacion")
```

Lift de Reglas de asociación