

Graficar

Jorge Loría

Oct 11, 2017

Base

Cargue los datos que se le entregaron, con el comando correspondiente.

```
load('Datos_Credito.RData')
```

Para esta clase vamos a ocupar dos librerías:

- ▶ dplyr

Base

Cargue los datos que se le entregaron, con el comando correspondiente.

```
load('Datos_Credito.RData')
```

Para esta clase vamos a ocupar dos librerías:

- ▶ dplyr
- ▶ ggplot2

Estructura de estos datos

Vamos a usar el dataframe que se indicó anteriormente, el cual tiene las siguientes columnas:

```
str(creditcardmarketing_bbm)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame': 18000 obs. of 17 variables
## $ Num_cliente      : int 1 2 3 4 5 6 7 8 9 10 ...
## $ Acepta_oferta    : chr "No" "No" "No" "No" ...
## $ Premio           : chr "Air Miles" "Air Miles" "Air Miles" "Air Miles"
## $ Tipo_mensajeria : chr "Letter" "Letter" "Postcard" "Letter" ...
## $ Nivel_ingresos   : chr "High" "Medium" "High" "Medium" ...
## $ Num_cuentas      : int 1 1 2 2 1 1 1 1 1 2 ...
## $ Proteccion_deficit: chr "No" "No" "No" "No" ...
## $ Rating            : chr "High" "Medium" "Medium" "High" ...
## $ Num_tarjetas_credito: int 2 2 2 1 2 3 2 4 2 3 ...
## $ Num_casas         : int 1 2 1 1 1 1 1 1 1 2 ...
## $ Tamano hogar     : int 4 5 2 4 6 4 3 4 4 4 ...
```

Descripción de las variables

```
names(creditcardmarketing_bbm)
```

```
## [1] "Num_cliente"           "Acepta_oferta"          "Premio"  
## [4] "Tipo_mensajeria"       "Nivel_ingresos"        "Num_cuentas"  
## [7] "Proteccion_deficit"    "Rating"                 "Num_tarjetas_credito"  
## [10] "Num_casas"              "Tamano hogar"          "Duenno_hogar"  
## [13] "Balance_promedio"       "Bal_1"                  "Bal_2"  
## [16] "Bal_3"                  "Bal_4"
```

Primer gráfico

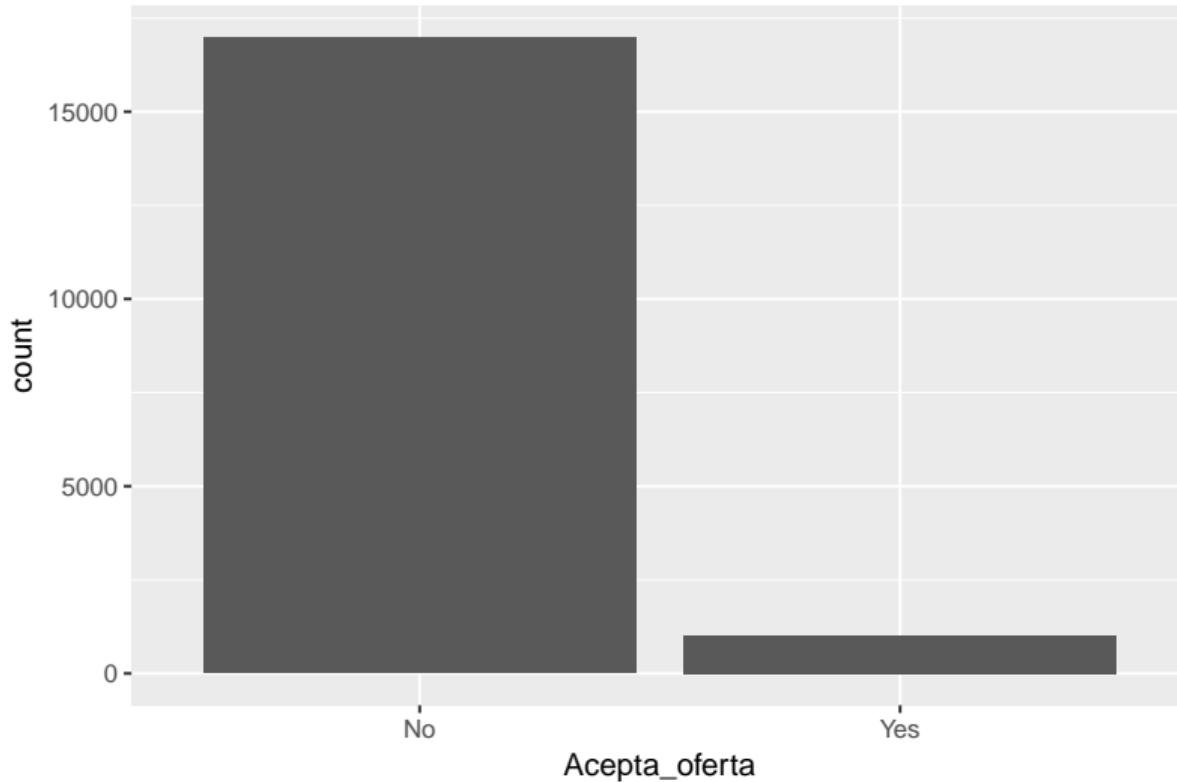
Para describir una variable categórica se puede usar un gráfico de barras, estos le dan una altura correspondiente a la cantidad de observaciones que se observen en cada categoría

Primer gráfico

Para describir una variable categórica se puede usar un gráfico de barras, estos le dan una altura correspondiente a la cantidad de observaciones que se observen en cada categoría

```
ggplot(data = creditcardmarketing_bbm) +  
  geom_bar(mapping = aes(x = Acepta_oferta))
```

Primer gráfico

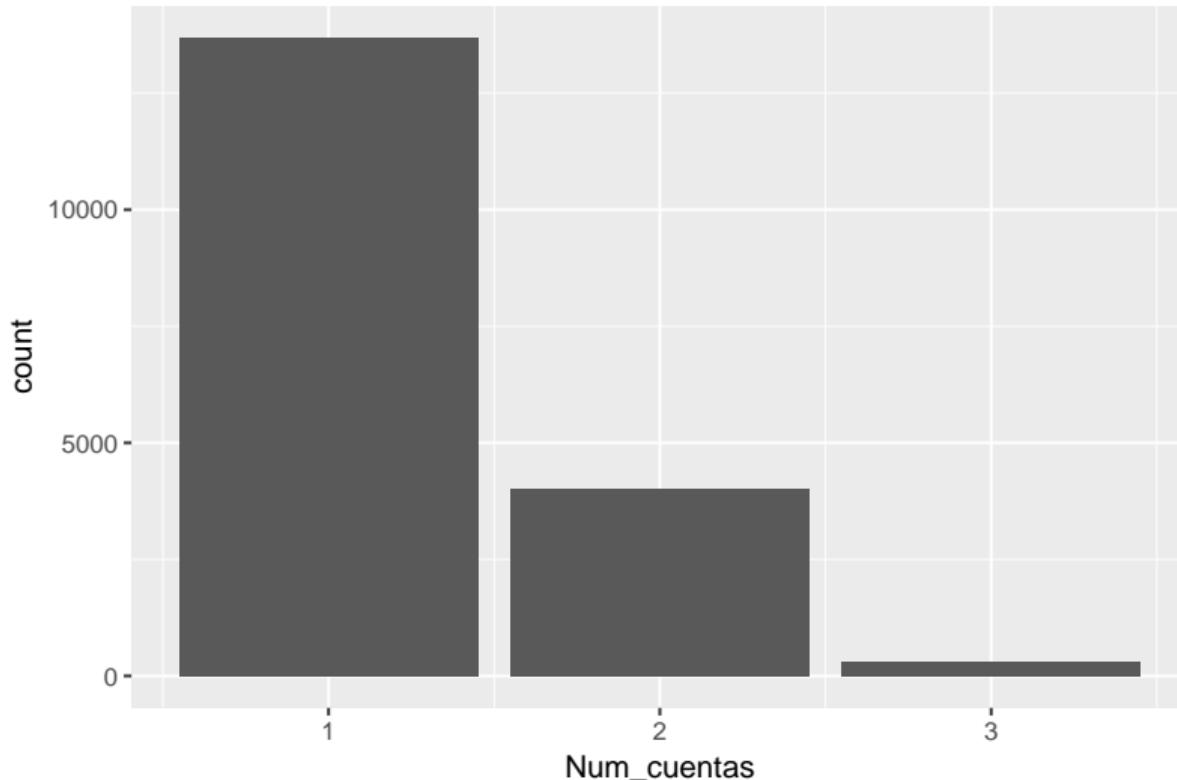


Segundo gráfico

Repita el gráfico anterior, pero usando la variable Num_cuentas

Segundo gráfico

Repita el gráfico anterior, pero usando la variable Num_cuentas



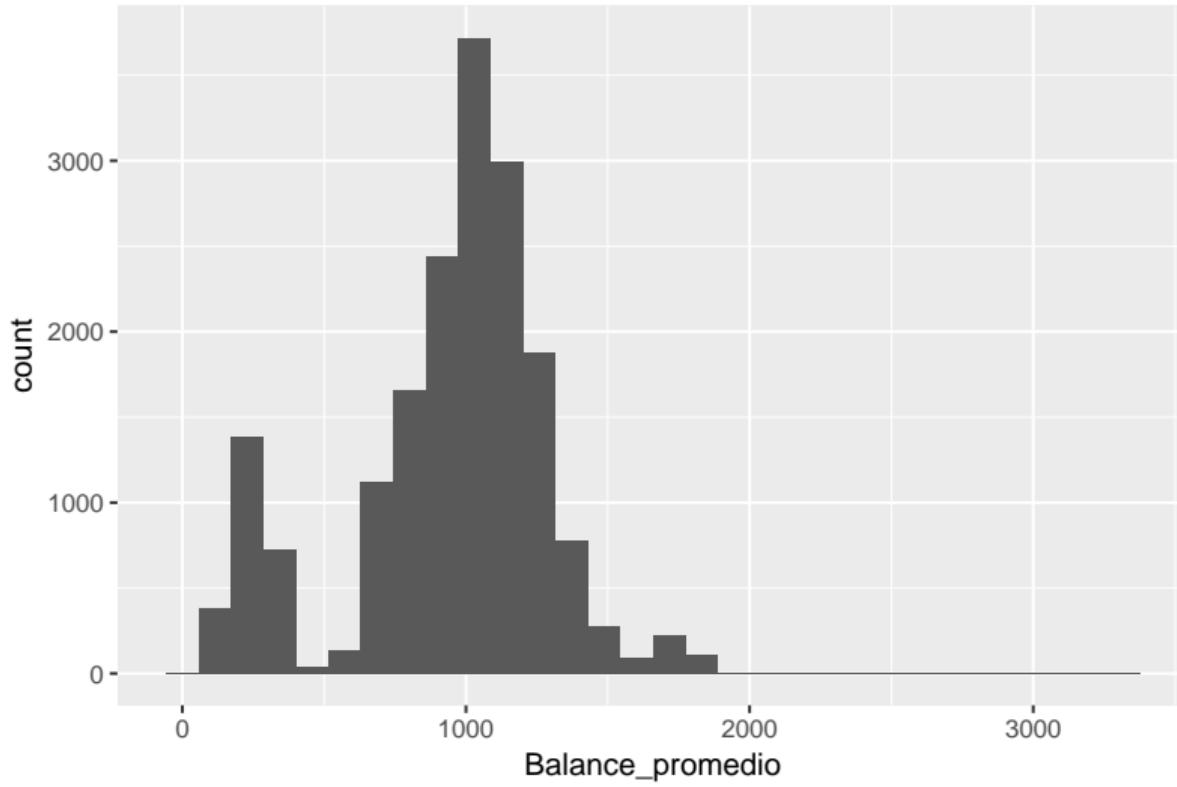
Una variable continua

Para describir una variable continua se usan los histogramas, los cuales hacen “cajitas” de cierto ancho, y dependiendo de cuantas observaciones se encuentren en cada cajita se le asigna esa altura.

```
ggplot(data = creditcardmarketing_bbm) +  
  geom_histogram(mapping = aes(x=Balance_promedio),na.rm=T)
```

Una variable continua

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Una variable continua, 2

Repita el gráfico anterior, pero usando Bal_1.

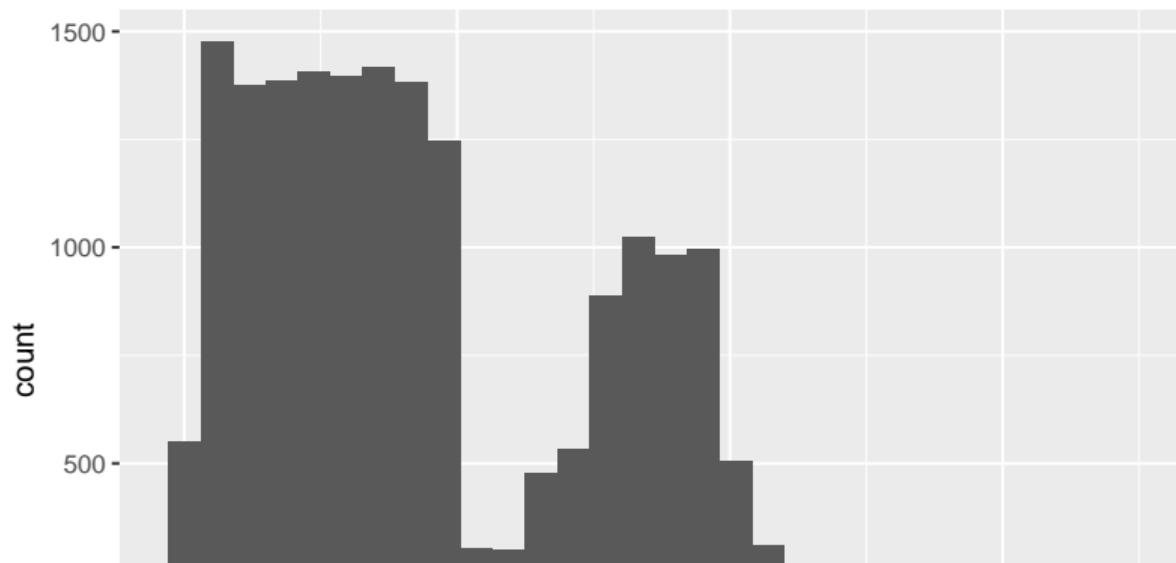
Una variable continua, 2

Repita el gráfico anterior, pero usando Bal_1. ¿Qué pasa si no pone el parámetro na.rm = T?

Una variable continua, 2

Repita el gráfico anterior, pero usando Bal_1. ¿Qué pasa si no pone el parámetro na.rm = T?

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.  
## Warning: Removed 24 rows containing non-finite values (stat_bin).
```



Interacción de variables

Con dos variables continuas:

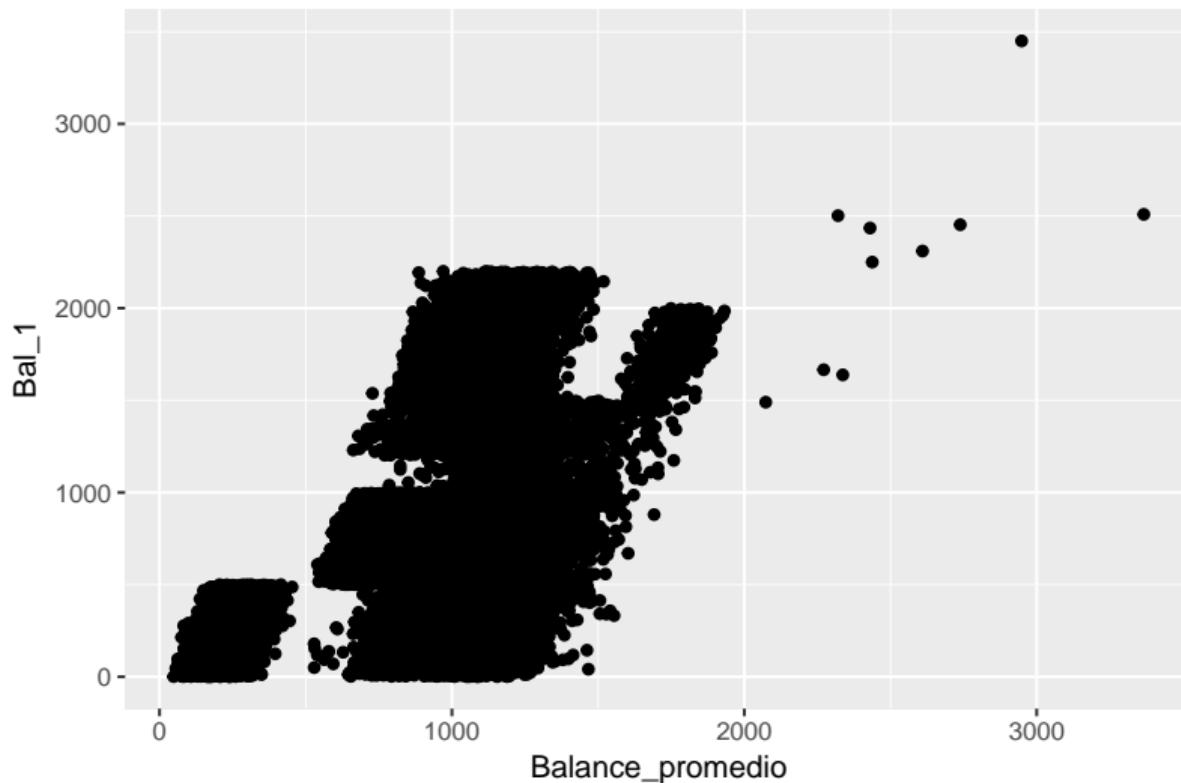
Interacción de variables

Con dos variables continuas:

```
ggplot(data = creditcardmarketing_bbm,  
       mapping = aes(x = Balance_promedio,y = Bal_1)) +  
  geom_point(na.rm =TRUE)
```

Interacción de variables

Con dos variables continuas:

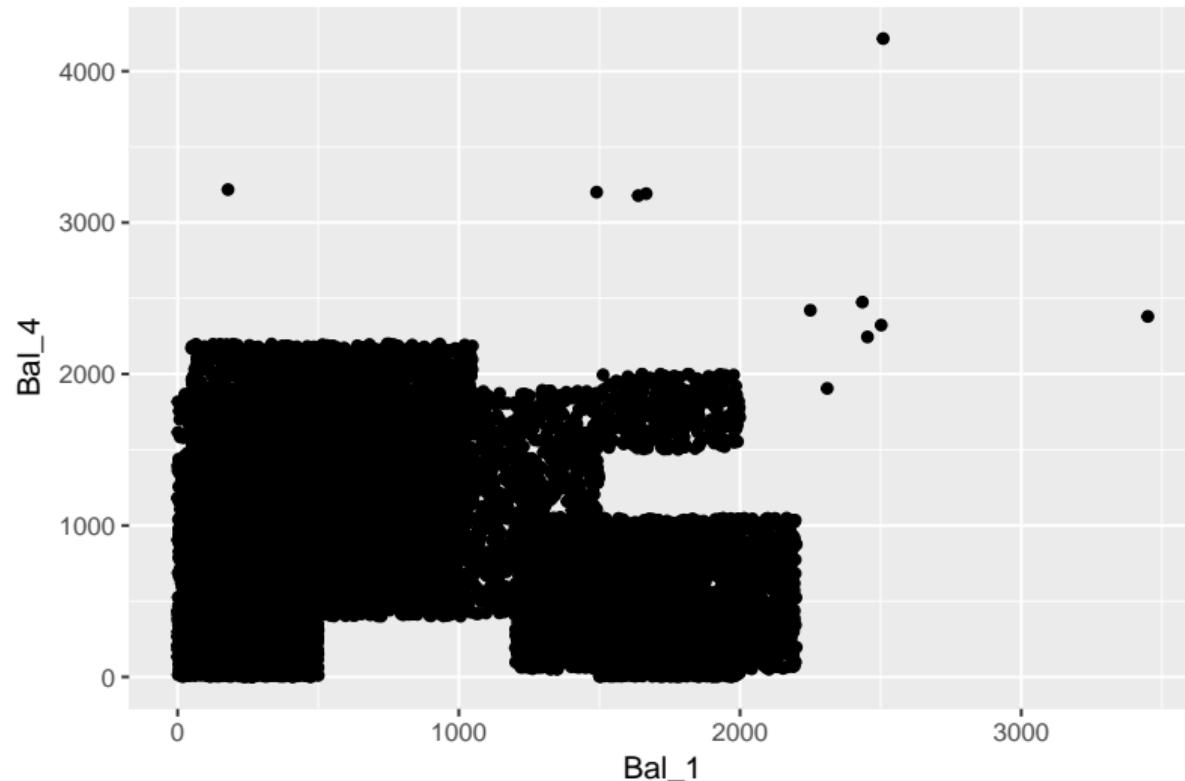


Interacción de variables

Repita el gráfico anterior, pero usando Bal_1 y Bal_4

Interacción de variables

Repita el gráfico anterior, pero usando Bal_1 y Bal_4



Preguntas

¿Qué función de dplyr usaría para identificar al sujeto con Bal_1 mayor a 3000?

Preguntas

¿Qué función de dplyr usaría para identificar al sujeto con Bal_1 mayor a 3000?

¿Cuántos tienen Bal_4 mayor a 3000?

Preguntas

¿Qué función de dplyr usaría para identificar al sujeto con Bal_1 mayor a 3000?

¿Cuántos tienen Bal_4 mayor a 3000?

¿Cuántos tienen alguno de los dos anteriores?

Preguntas

¿Qué función de dplyr usaría para identificar al sujeto con Bal_1 mayor a 3000?

¿Cuántos tienen Bal_4 mayor a 3000?

¿Cuántos tienen alguno de los dos anteriores?

```
filter(Bal_1 > 3000)
```

Preguntas

¿Qué función de dplyr usaría para identificar al sujeto con Bal_1 mayor a 3000?

¿Cuántos tienen Bal_4 mayor a 3000?

¿Cuántos tienen alguno de los dos anteriores?

`filter(Bal_1 > 3000)`

`filter(Bal_4 > 3000)`

Preguntas

¿Qué función de dplyr usaría para identificar al sujeto con Bal_1 mayor a 3000?

¿Cuántos tienen Bal_4 mayor a 3000?

¿Cuántos tienen alguno de los dos anteriores?

`filter(Bal_1 > 3000)`

`filter(Bal_4 > 3000)`

`filter(Bal_1 > 3000 | Bal_4 > 3000)`

Preguntas

¿Qué función de dplyr usaría para identificar al sujeto con Bal_1 mayor a 3000?

¿Cuántos tienen Bal_4 mayor a 3000?

¿Cuántos tienen alguno de los dos anteriores?

`filter(Bal_1 > 3000)`

`filter(Bal_4 > 3000)`

`filter(Bal_1 > 3000 | Bal_4 > 3000)`

Agregar colores

Agregar colores



Figure 1: A pintar!

Agregar colores

Si se quieren agregar colores, se agrega en el aes() el parámetro color = <VARIABLE_A_USAR> (o colour):

Agregar colores

Si se quieren agregar colores, se agrega en el aes() el parámetro color = <VARIABLE_A_USAR> (o colour):

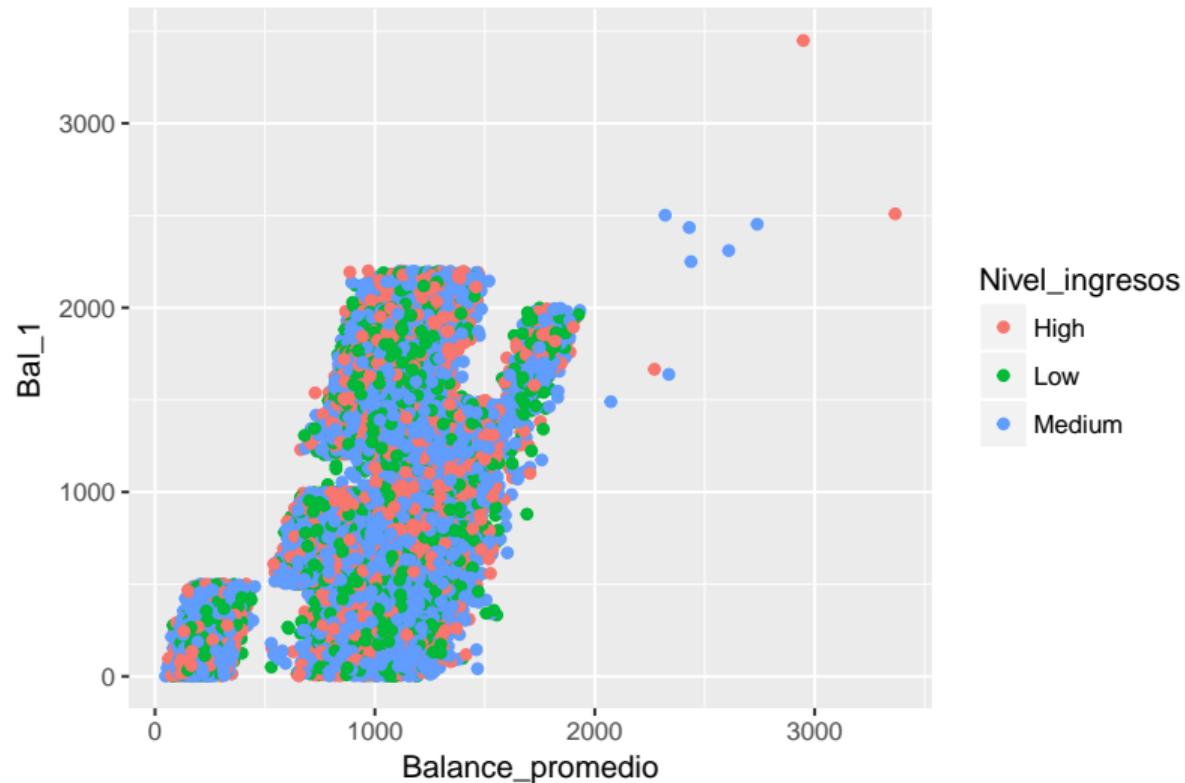
```
ggplot(data = creditcardmarketing_bbm) +
  geom_point(mapping = aes(x = Balance_promedio,
                           y = Bal_1,
                           color = Nivel_ingresos),
             na.rm = TRUE)
```

Agregar colores

Y queda coloreado...

Agregar colores

Y queda coloreado...

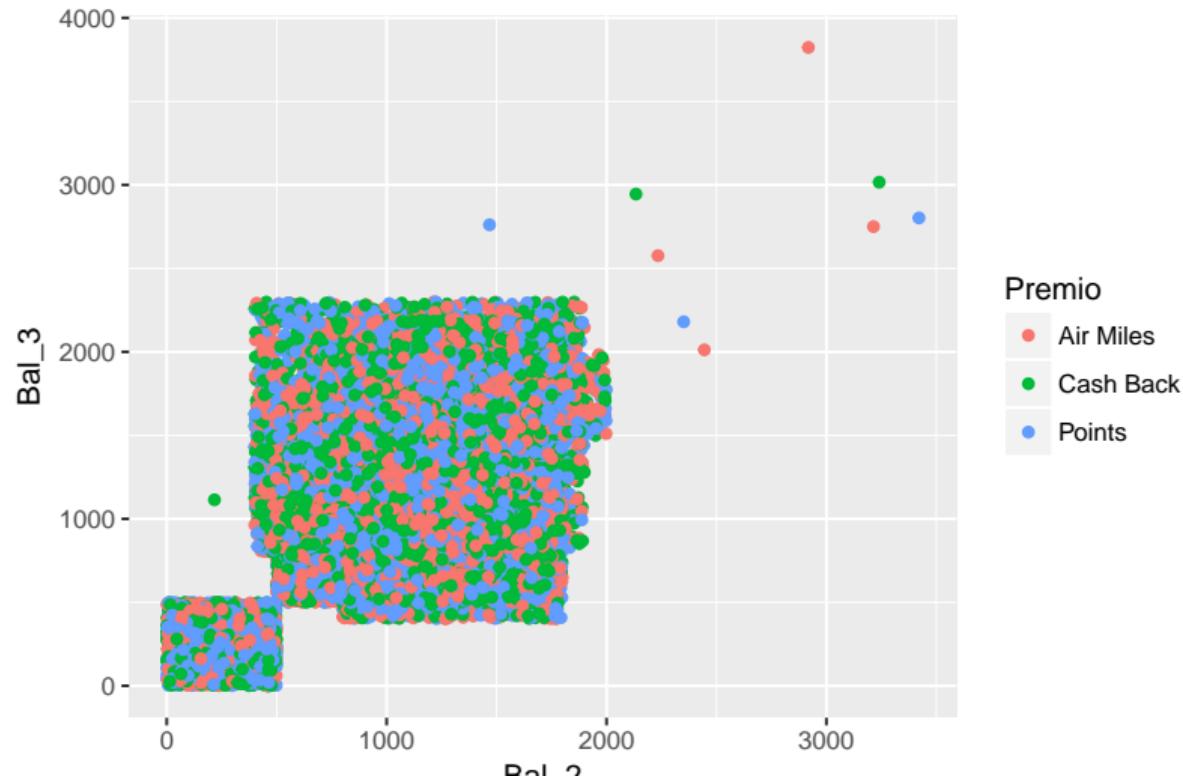


Ejercicio colores

Repita el gráfico anterior, pero usando Bal_2, en el eje X, Bal_3 en el eje Y, y coloreando con la columna Premio.

Ejercicio colores

Repita el gráfico anterior, pero usando Bal_2, en el eje X, Bal_3 en el eje Y, y coloreando con la columna Premio.



Transparencia



RENDICIÓN DE CUENTAS

Figure 2: Pero no política

Transparencia

Muchas veces quedan imágenes como la anterior en la que quedan manchas de colores en la imagen y no se entiende bien

Transparencia

Muchas veces quedan imágenes como la anterior en la que quedan manchas de colores en la imagen y no se entiende bien, para esto se usa el parámetro alpha, para indicar el nivel de transparencia que se quiere en la imagen a usar:

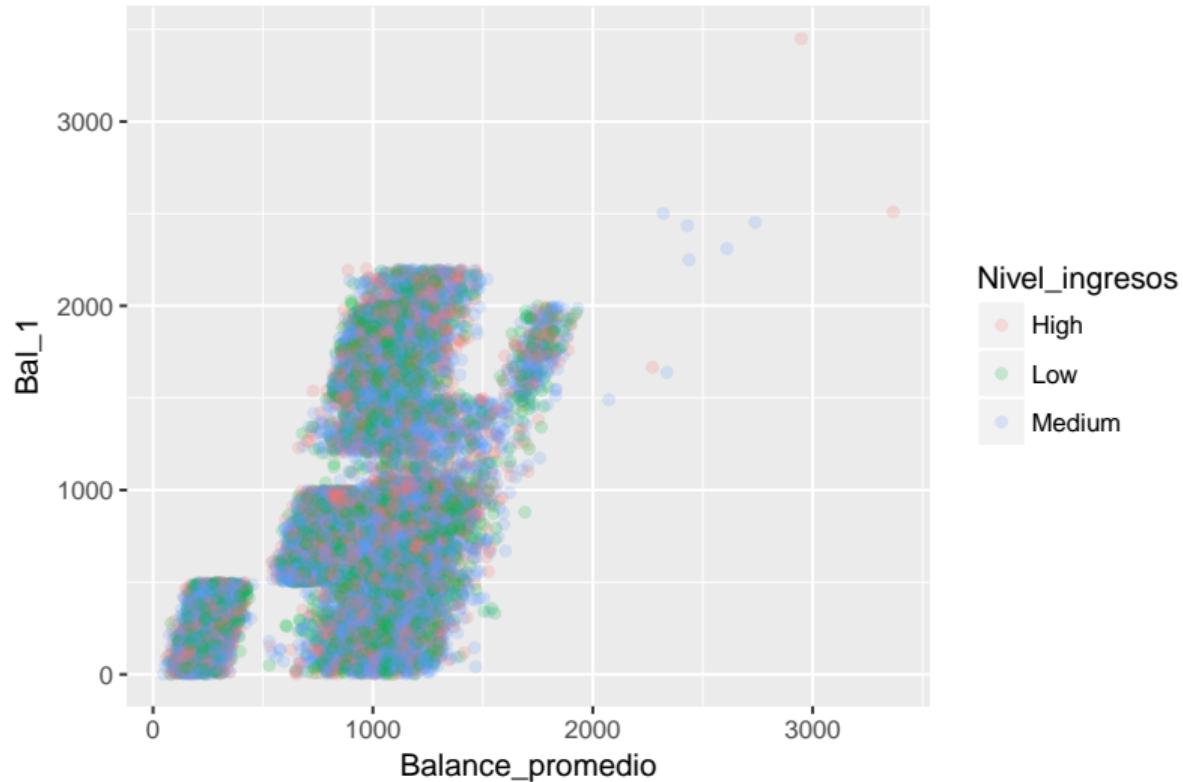
Transparencia

Muchas veces quedan imágenes como la anterior en la que quedan manchas de colores en la imagen y no se entiende bien, para esto se usa el parámetro alpha, para indicar el nivel de transparencia que se quiere en la imagen a usar:

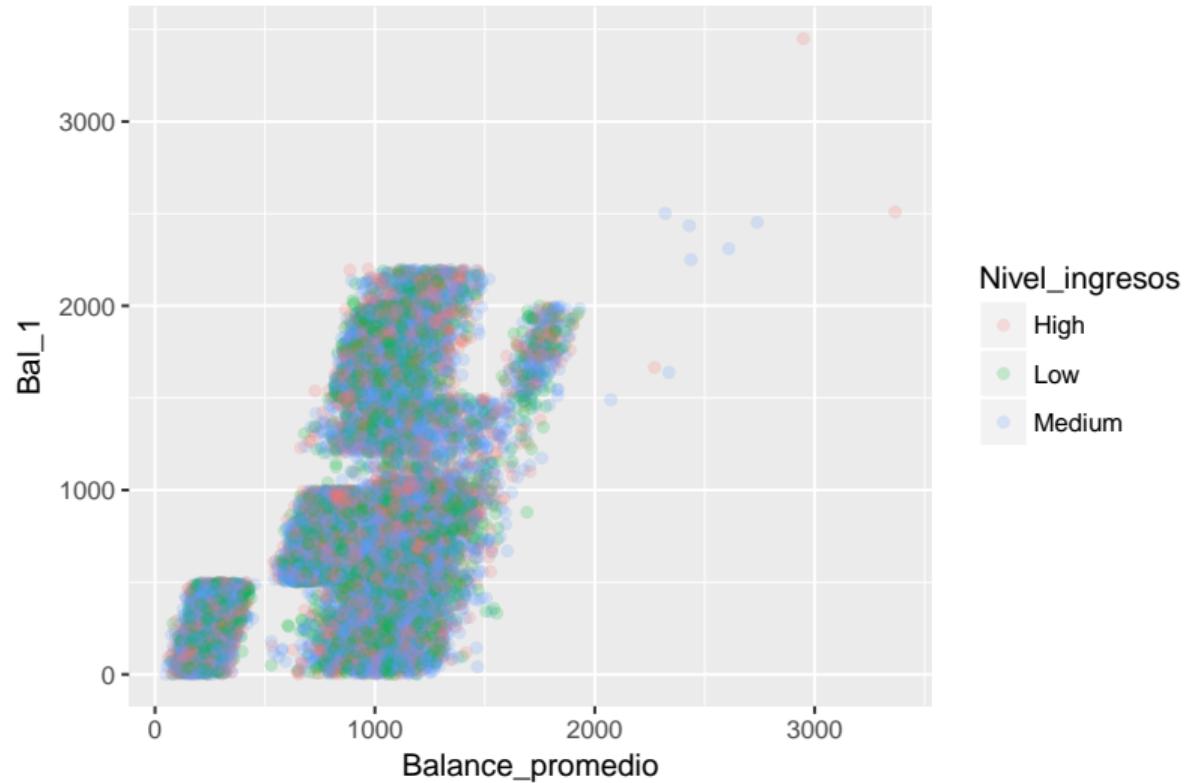
```
ggplot(data = creditcardmarketing_bbm) +  
  geom_point(mapping = aes(x = Balance_promedio,  
                           y = Bal_1,color = Nivel_ingresos),  
             alpha = 1/5,  
             na.rm = TRUE)
```

Note que alpha es un parámetro de la función geom_point, no de la función aes

Transparencia



Transparencia



Con esto se interpreta mejor en *cuales* lugares están concentrados los balances.

Tamaño

Tamaño

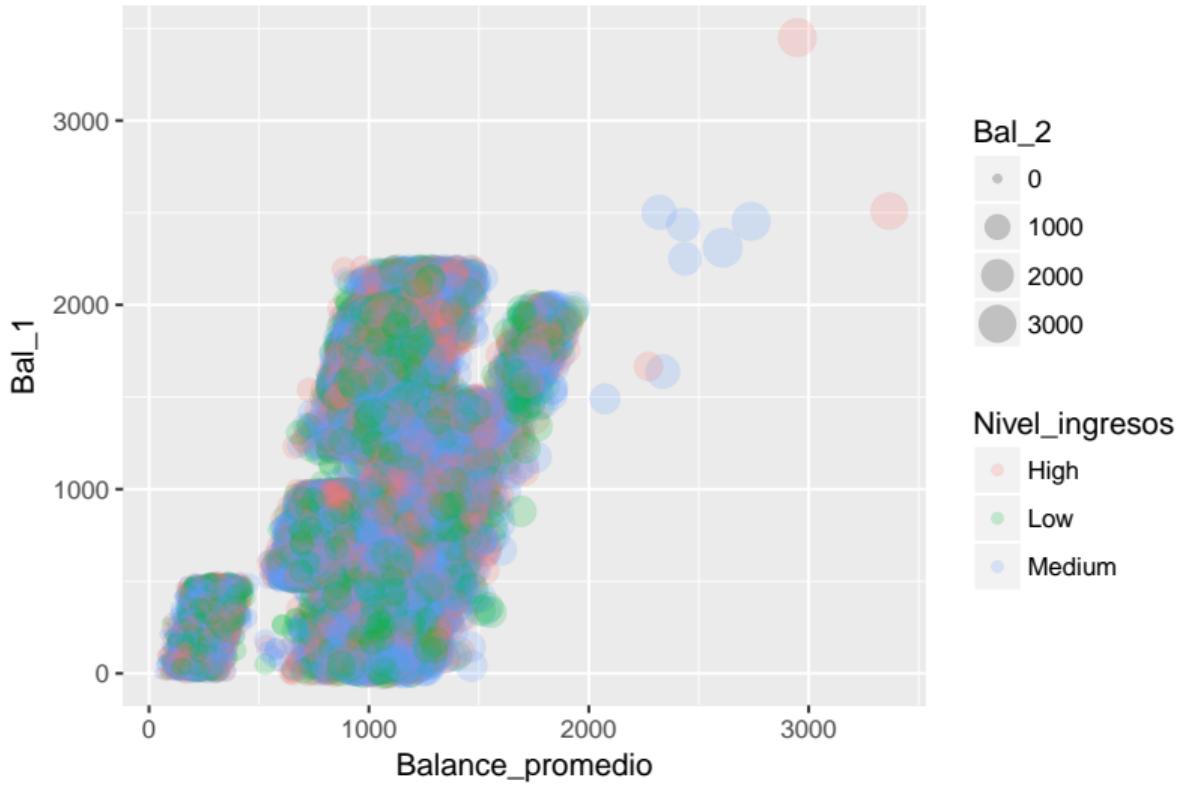
Importa

Tamaño

Existen bastantes más parámetros que se pueden incluir, entre estos está `size`, que puede definirse como una constante o bien, que dependa de otra variable:

```
ggplot(data = creditcardmarketing_bbm) +  
  geom_point(mapping = aes(x = Balance_promedio,  
                           y = Bal_1,color = Nivel_ingresos,  
                           size = Bal_2),  
             alpha = 1/5,  
             na.rm = TRUE)
```

Tamaño

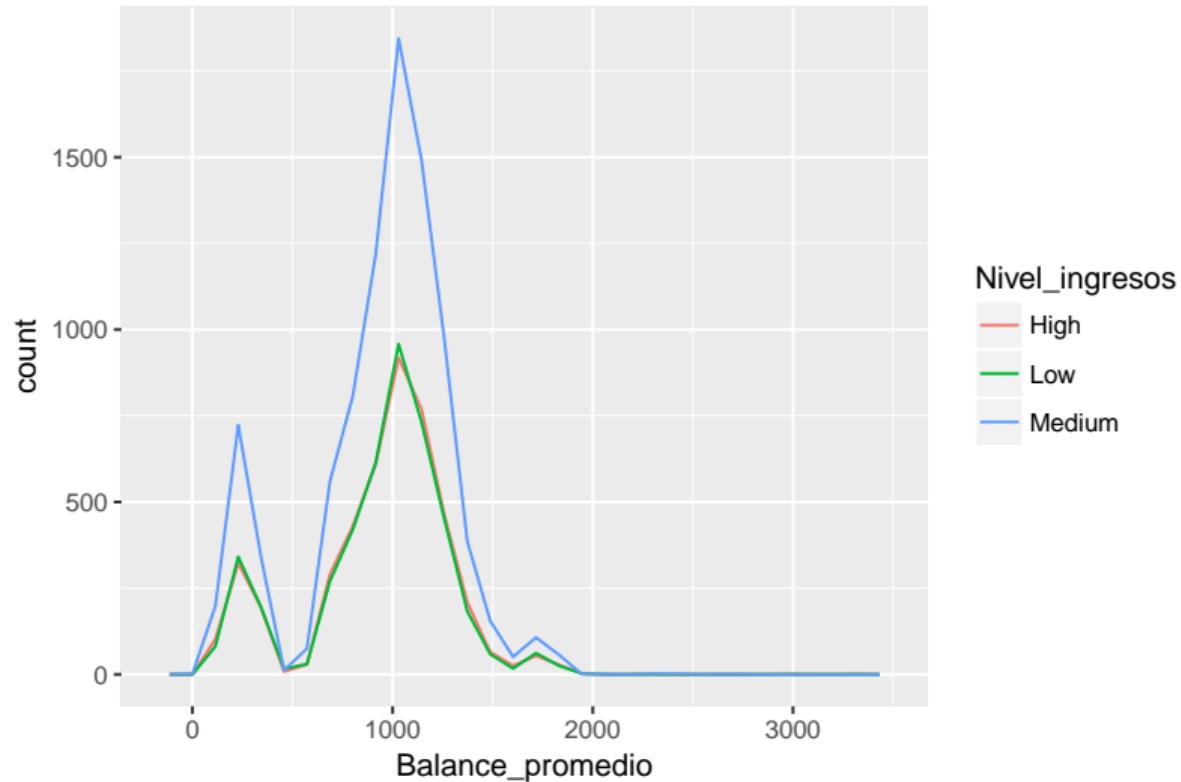


Variables continuas vs discretas

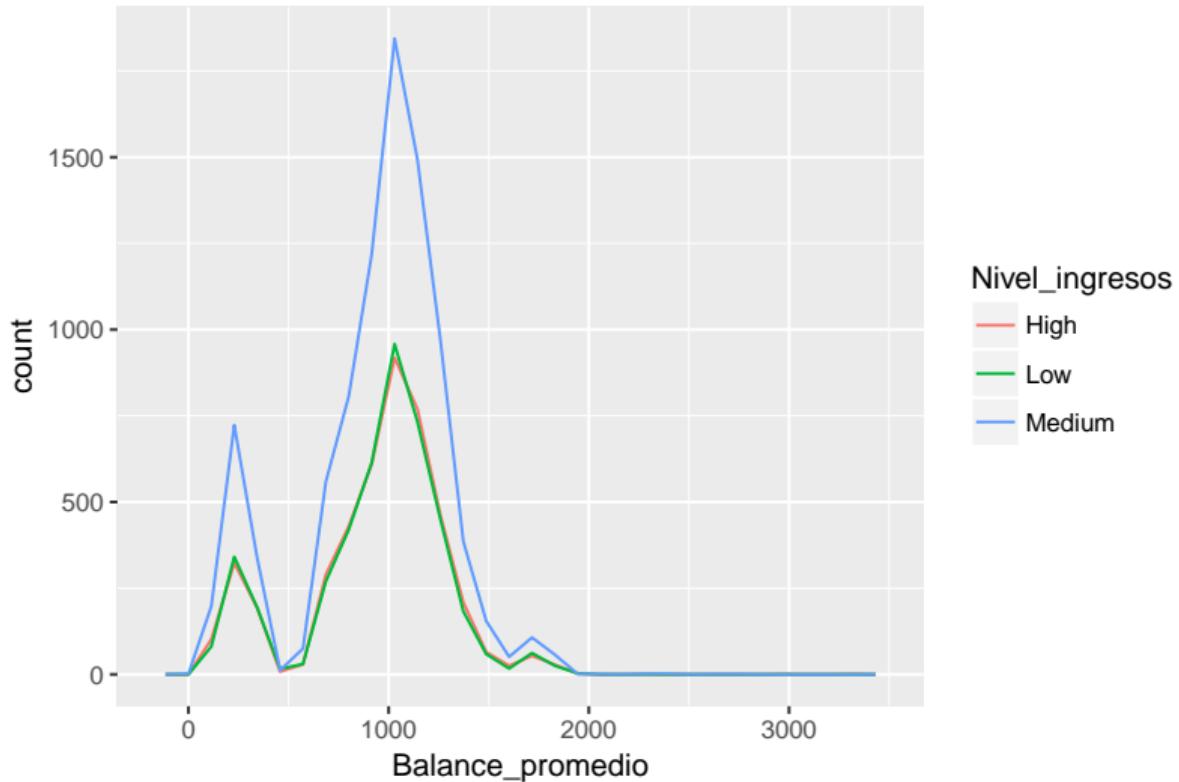
Para graficar variables continuas vs las variables discretas se puede usar la función `geom_freqpoly`, que es muy similar a la función del `histogram` en el sentido de que muestra cual es la distribución de las variables. Solo que es más claro para agrupar en distintos grupos que en el histograma.

```
ggplot(data = creditcardmarketing_bbm) +
  geom_freqpoly(mapping = aes(x = Balance_promedio,
                               colour = Nivel_ingresos),
                na.rm = TRUE, bins = 30)
```

Variables continuas vs discretas



Variables continuas vs discretas



Parece que las distribuciones son similares entre los niveles de ingreso.

Verlo como porcentaje:

Verlo como porcentaje:

Para graficarlo como porcentaje se incluye el parámetro `y=..density..`

Verlo como porcentaje:

Para graficarlo como porcentaje se incluye el parámetro `y=..density..`

```
ggplot(data = creditcardmarketing_bbm) +  
  geom_freqpoly(mapping = aes(x = Balance_promedio,  
                               colour = Nivel_ingresos,  
                               y = ..density..),  
                na.rm = TRUE,  
                bins = 30)
```

El parámetro `bins`, está predefinido como 30, y da un aviso de que vale eso. Por lo que lo ponemos explícito de que vale 30. En cursos avanzados se ven métodos para elegir este parámetro apropiadamente.

Verlo como porcentaje:

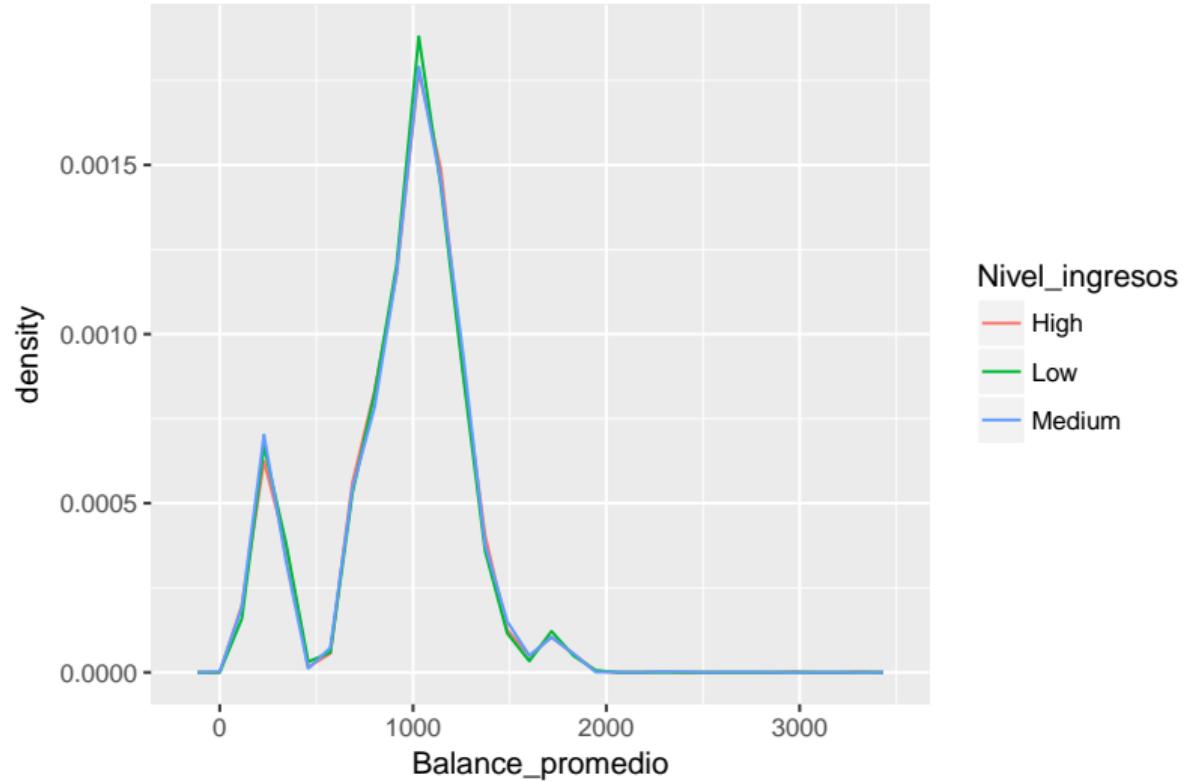
Para graficarlo como porcentaje se incluye el parámetro `y=..density..`

```
ggplot(data = creditcardmarketing_bbm) +  
  geom_freqpoly(mapping = aes(x = Balance_promedio,  
                               colour = Nivel_ingresos,  
                               y = ..density..),  
                na.rm = TRUE,  
                bins = 30)
```

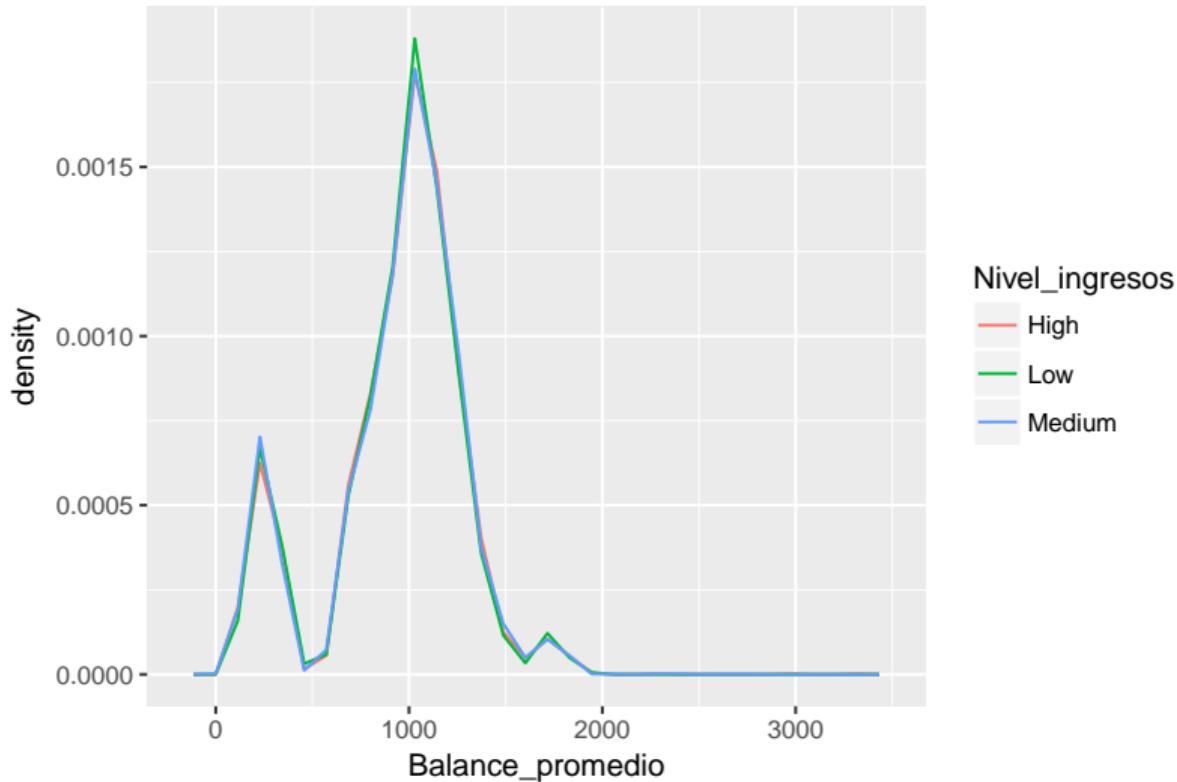
El parámetro `bins`, está predefinido como 30, y da un aviso de que vale eso. Por lo que lo ponemos explícito de que vale 30. En cursos avanzados se ven métodos para elegir este parámetro apropiadamente. Si intenta no incluirlo, se indica que se pone como 30.

Verlo como porcentaje

Verlo como porcentaje



Verlo como porcentaje



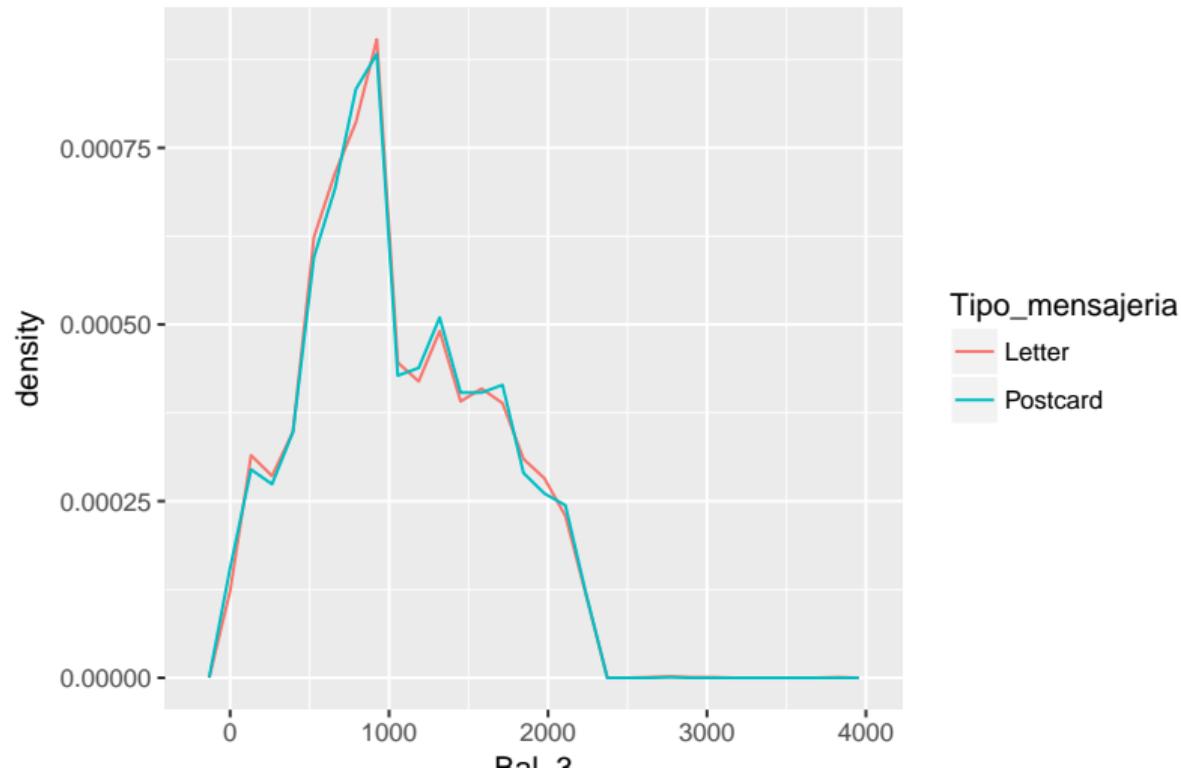
Esto nos indica que el balance promedio no depende del nivel de ingresos.

Ejercicio

Repita el gráfico anterior, usando el Bal_3 agrupando con el tipo de mensajería.

Ejercicio

Repita el gráfico anterior, usando el Bal_3 agrupando con el tipo de mensajería. ¿Hay mucha diferencia?



Otro ejercicio

En el gráfico anterior, ¿qué pasa si pone el parámetro `bins = 20`? ¿Igual a 40? ¿Igual a 2?

Dos variables categóricas

Dos variables categóricas

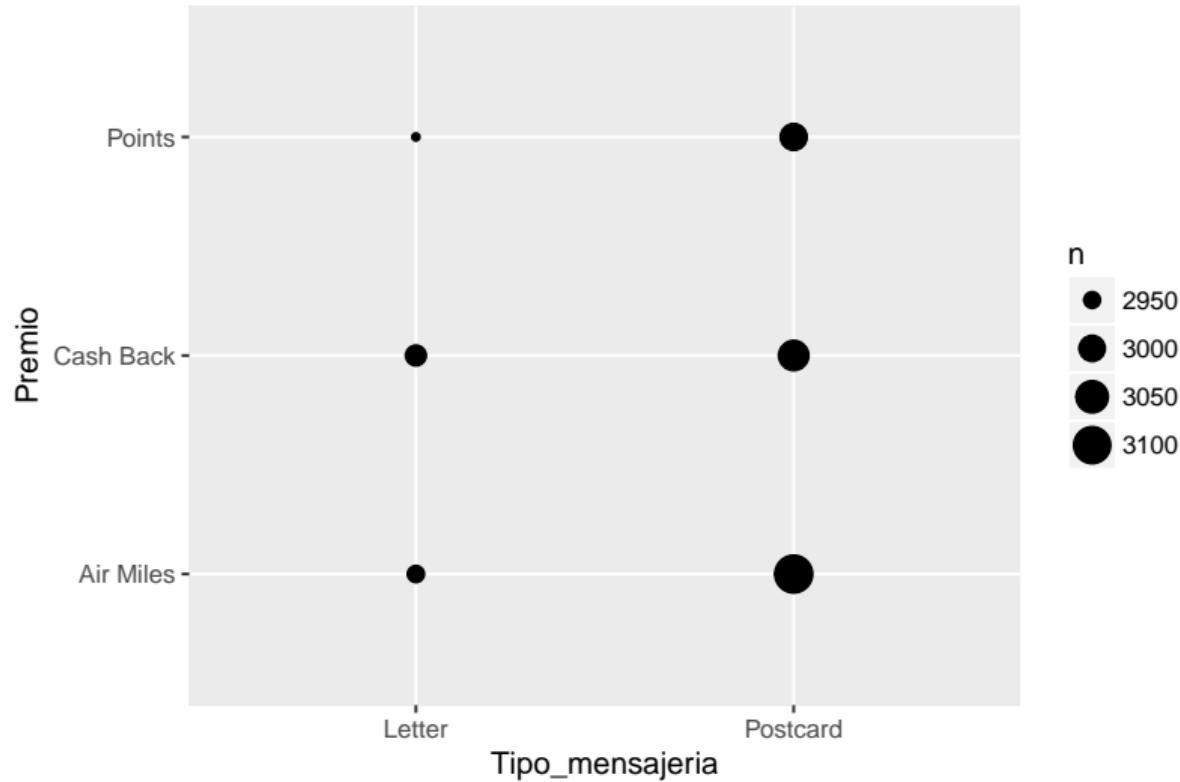
Otro tipo de relación que se puede explorar es entre dos variables categóricas, usando la función `geom_count`, la cual agrupa entre ambas categorías y cuenta cuantos hay en cada combinación:

Dos variables categóricas

Otro tipo de relación que se puede explorar es entre dos variables categóricas, usando la función `geom_count`, la cual agrupa entre ambas categorías y cuenta cuantos hay en cada combinación:

```
ggplot(creditcardmarketing_bbm) +
  geom_count(aes(Tipo_mensajeria,Premio))
```

Dos variables categóricas

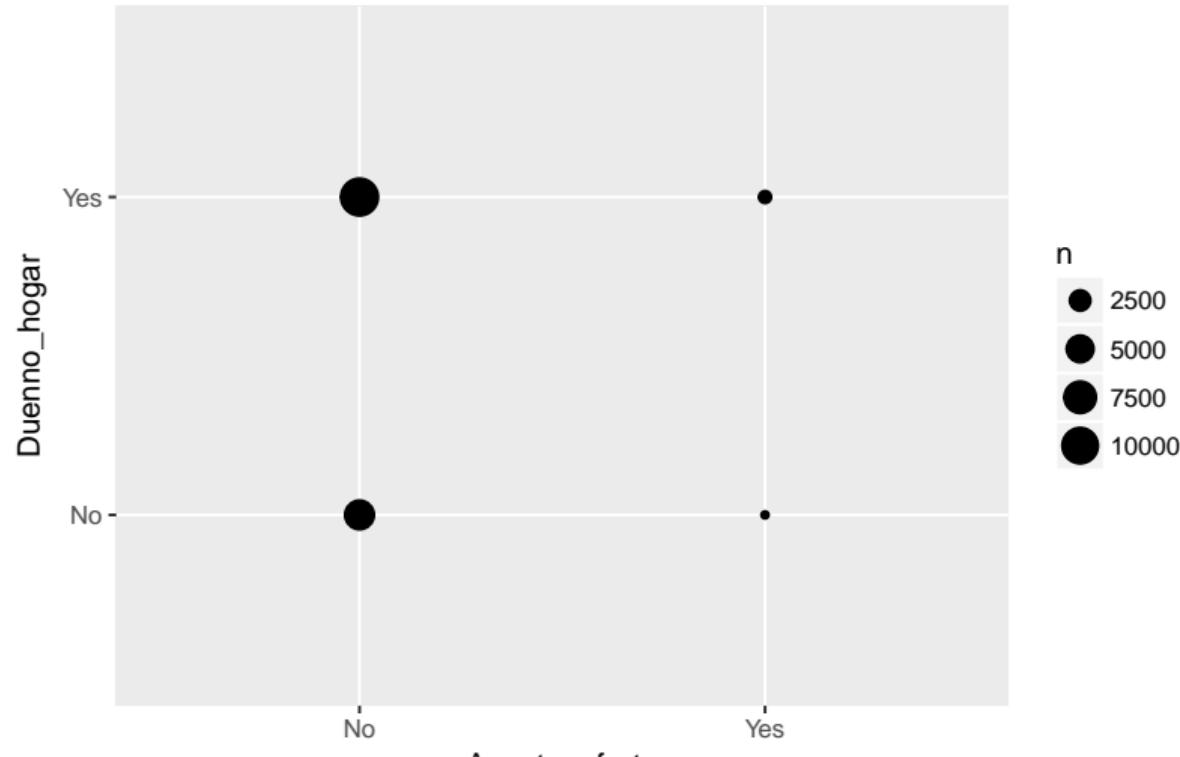


Ejercicio count

Repita el ejercicio anterior, pero revisando la relación entre Acepta_oferta y Duenno_hogar.

Ejercicio count

Repita el ejercicio anterior, pero revisando la relación entre Acepta_oferta y Duenno_hogar.



Sintaxis general de un ggplot

Todos los gráficos que se hacen en ggplot, siguen la siguiente sintaxis:

Sintaxis general de un ggplot

Todos los gráficos que se hacen en ggplot, siguen la siguiente sintaxis:

```
ggplot(data = <Mis_Datos>) +  
<GEOM_FUNCION>(mapping = aes(<EXPLICACION_VARIABLES_A_USAR>))
```

Sintaxis general de un ggplot

Todos los gráficos que se hacen en ggplot, siguen la siguiente sintaxis:

```
ggplot(data = <Mis_Datos>) +  
<GEOM_FUNCION>(mapping = aes(<EXPLICACION_VARIABLES_A_USAR>))
```

Se le pueden agregar más *capas*, como vamos a ver proximamente...

Lineas

Para mostrar progreso a través del tiempo de alguna variable o algo por el estilo, se pueden usar líneas, con la función geom_line:

```
library(tidyr)

creditos_Mes <- creditcardmarketing_bbm %>%
  gather(key = Mes,value = Balance,num_range('Bal_',1:4)) %>%
  separate(col = Mes,into = c('Bal','Mes'),sep = '_',convert = TRUE) %>%
  select(-Bal) %>%
  filter(Num_cliente < 10)

ggplot(data = creditos_Mes,
        aes(x = Mes,y = Balance,
            color = factor(Num_cliente))) +
  geom_line()
```

Lineas

Para mostrar progreso a través del tiempo de alguna variable o algo por el estilo, se pueden usar líneas, con la función geom_line:

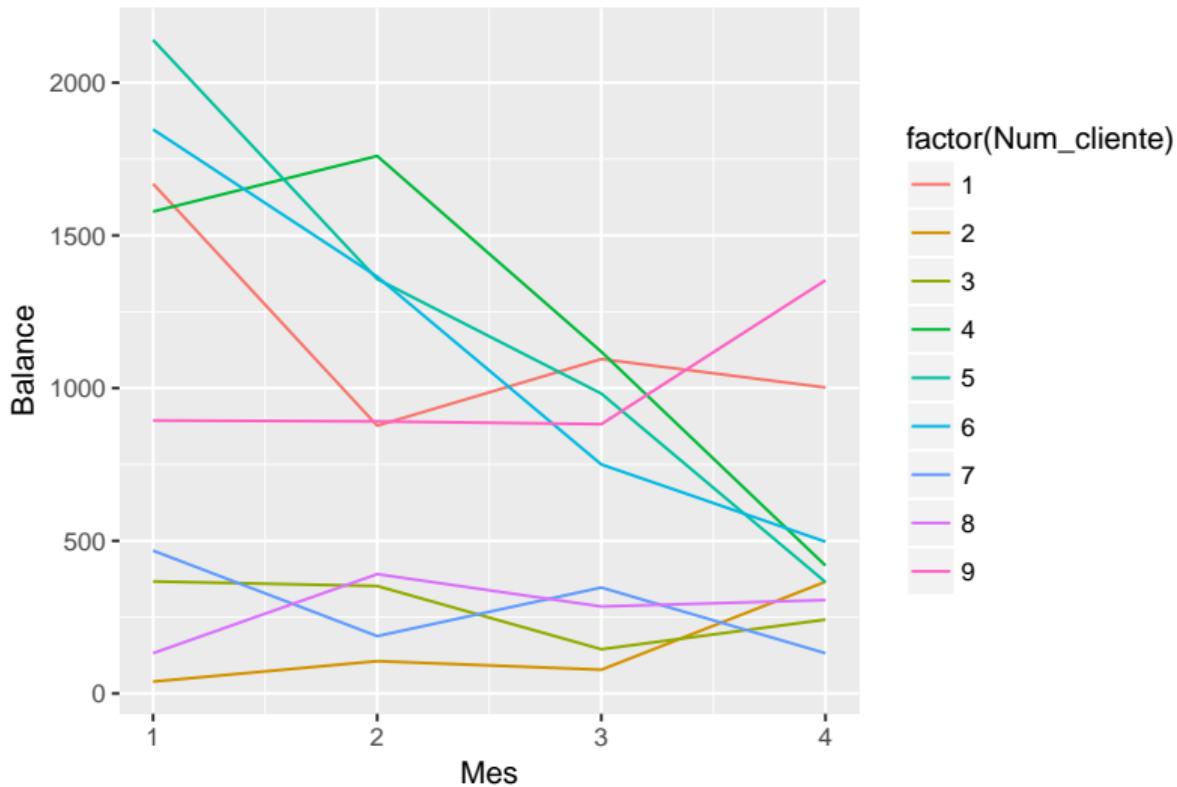
```
library(tidyr)

creditos_Mes <- creditcardmarketing_bbm %>%
  gather(key = Mes,value = Balance,num_range('Bal_',1:4)) %>%
  separate(col = Mes,into = c('Bal','Mes'),sep = '_',convert = TRUE) %>%
  select(-Bal) %>%
  filter(Num_cliente < 10)

ggplot(data = creditos_Mes,
        aes(x = Mes,y = Balance,
            color = factor(Num_cliente))) +
  geom_line()
```

¿Qué hace el código anterior?

Lines



Ejercicio lineas

Ejercicio lineas

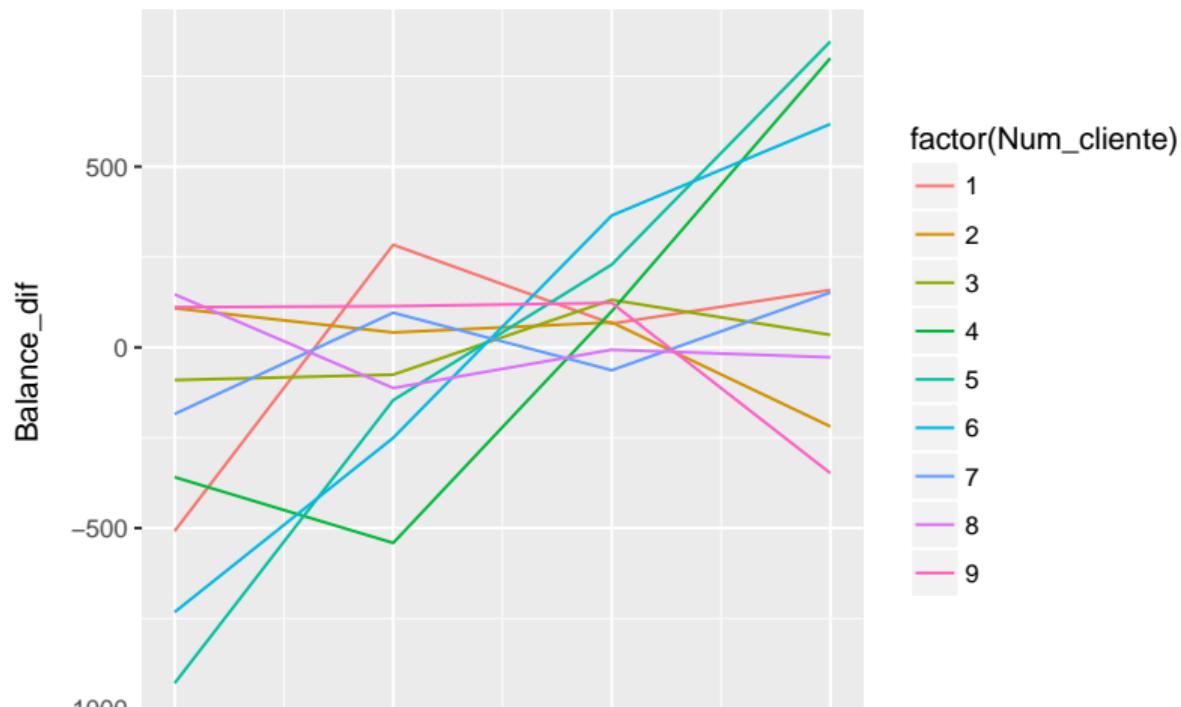
¿Qué pasa si se llama sin poner el factor alrededor de Num_cliente?

Ejercicio lineas

¿Qué pasa si se llama sin poner el factor alrededor de Num_cliente? Haga una nueva columna, que indique la diferencia entre Balance_promedio y Balance. Grafique esta diferencia respecto a estos meses.

Ejercicio lineas

¿Qué pasa si se llama sin poner el factor alrededor de Num_cliente? Haga una nueva columna, que indique la diferencia entre Balance_promedio y Balance. Grafique esta diferencia respecto a estos meses.



Capas

Capas



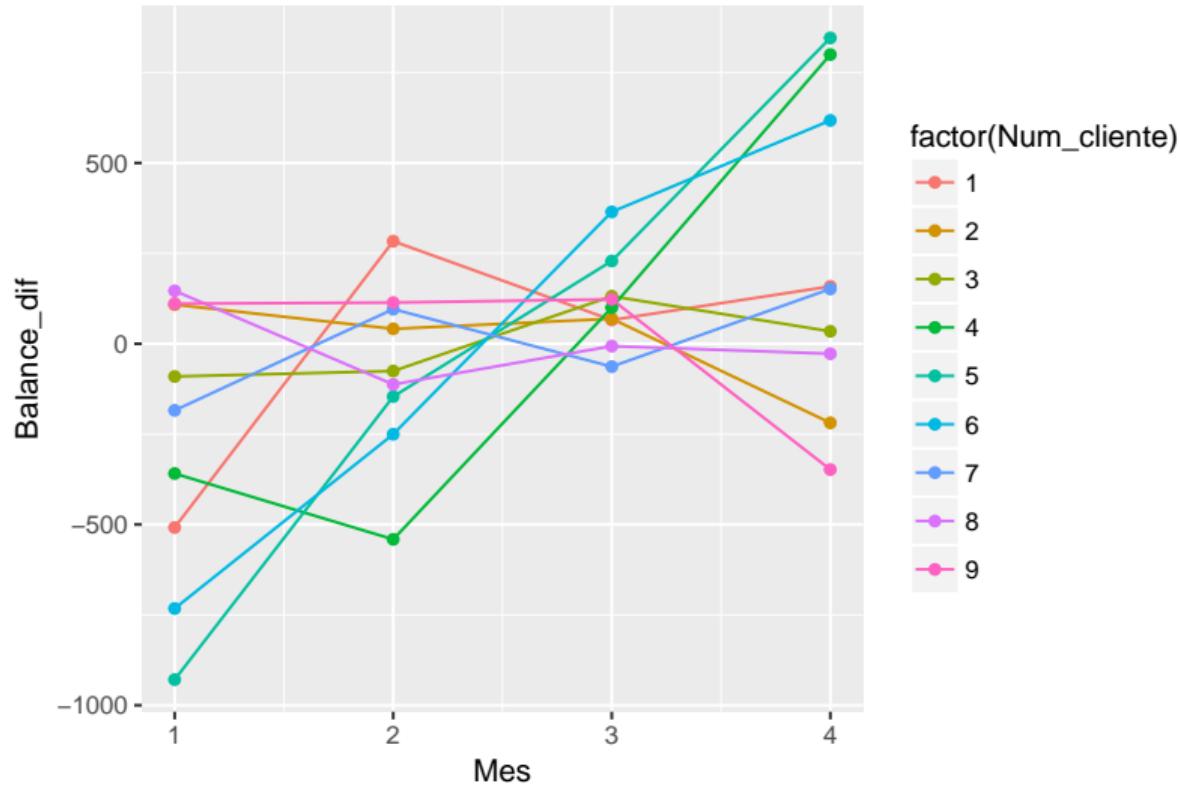
Figure 3: Pero no para la lluvia

Capas:

Hasta el momento nos hemos limitado a sumar una capa, pero podemos agregarle más capas, para recalcar ciertos valores:

```
creditos_Mes %>%
  mutate(Balance_dif = Balance_promedio - Balance) %>%
  ggplot(mapping = aes(x = Mes, y = Balance_dif,
                        color = factor(Num_cliente))) +
  geom_line() +
  geom_point()
```

Capas:

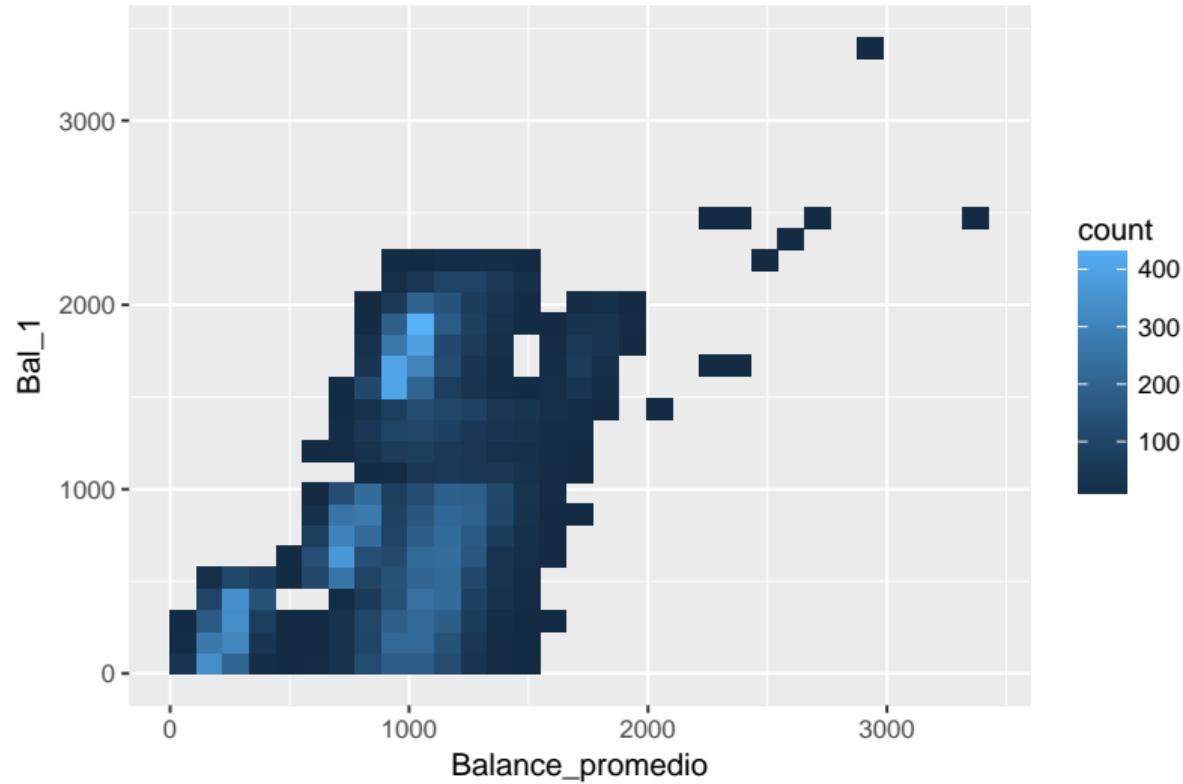


Otro tipo de gráfico:

Para comprender mejor la relación entre dos variables continuas, se puede usar la función geom_bind2d:

```
creditcardmarketing_bbm %>%
  ggplot(aes(x = Balance_promedio, y = Bal_1)) +
  geom_bin2d(na.rm = TRUE)
```

Otro tipo de gráfico:



Ejercicio geom_bin2d

Esta función recibe un parámetro que se llama `binwidth`, que es un vector que indica cual es el grosor de los bins que va a hacer (similar al histograma), para contar.

Ejercicio geom_bin2d

Esta función recibe un parámetro que se llama `binwidth`, que es un vector que indica cual es el grosor de los `bins` que va a hacer (similar al histograma), para contar. Repita 3 veces el gráfico anterior, e incluya este parámetro usando los siguientes pares:

- ▶ `c(20,20)`
- ▶ `c(100,20)`
- ▶ `c(30,50)`

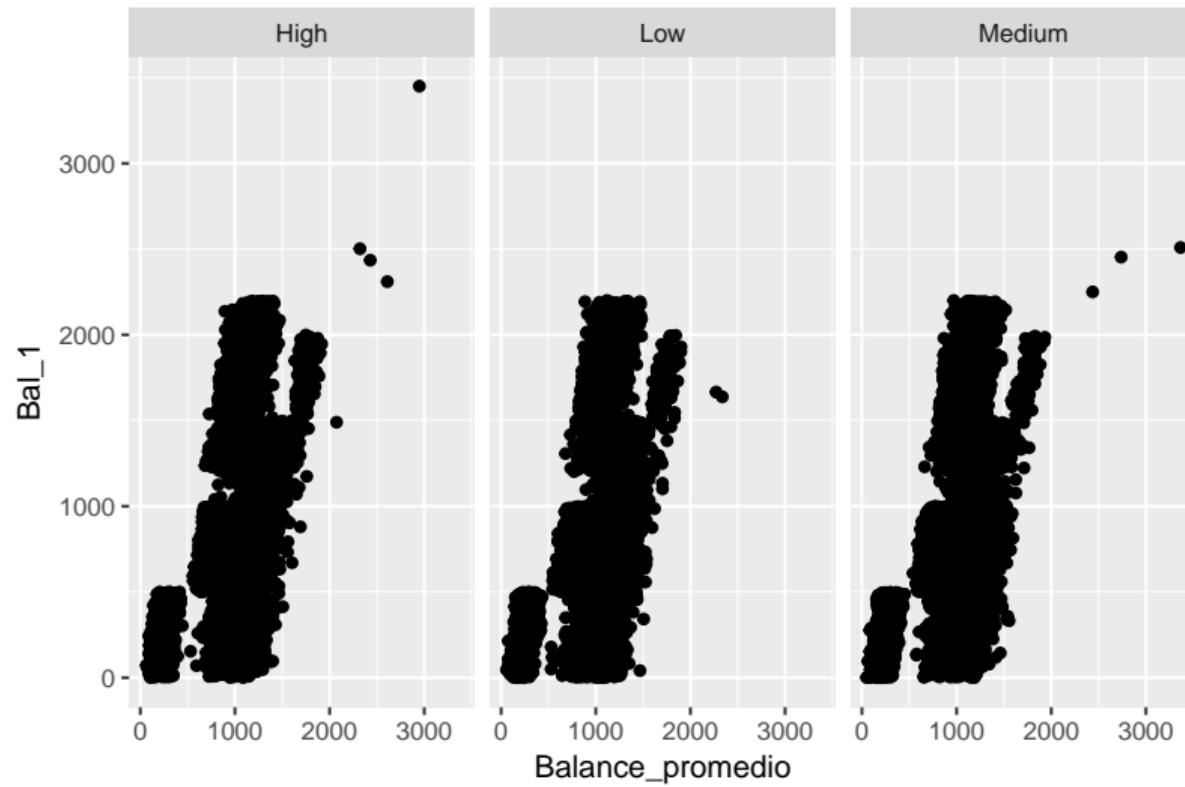
Más variables...

Más variables...

Para que el análisis descriptivo sea más claro, se pueden incluir más variables a la hora de hacer gráficos. Para esto se puede usar la capa facet_grid

```
creditcardmarketing_bbm %>%
  ggplot(aes(x = Balance_promedio,y = Bal_1)) +
  geom_point(na.rm = TRUE) +
  facet_grid(.~Rating)
```

Más variables...



Ejercicio más variables

Repita el gráfico anterior, pero usando `facet_grid(Tipo_mensajeria~.)`.

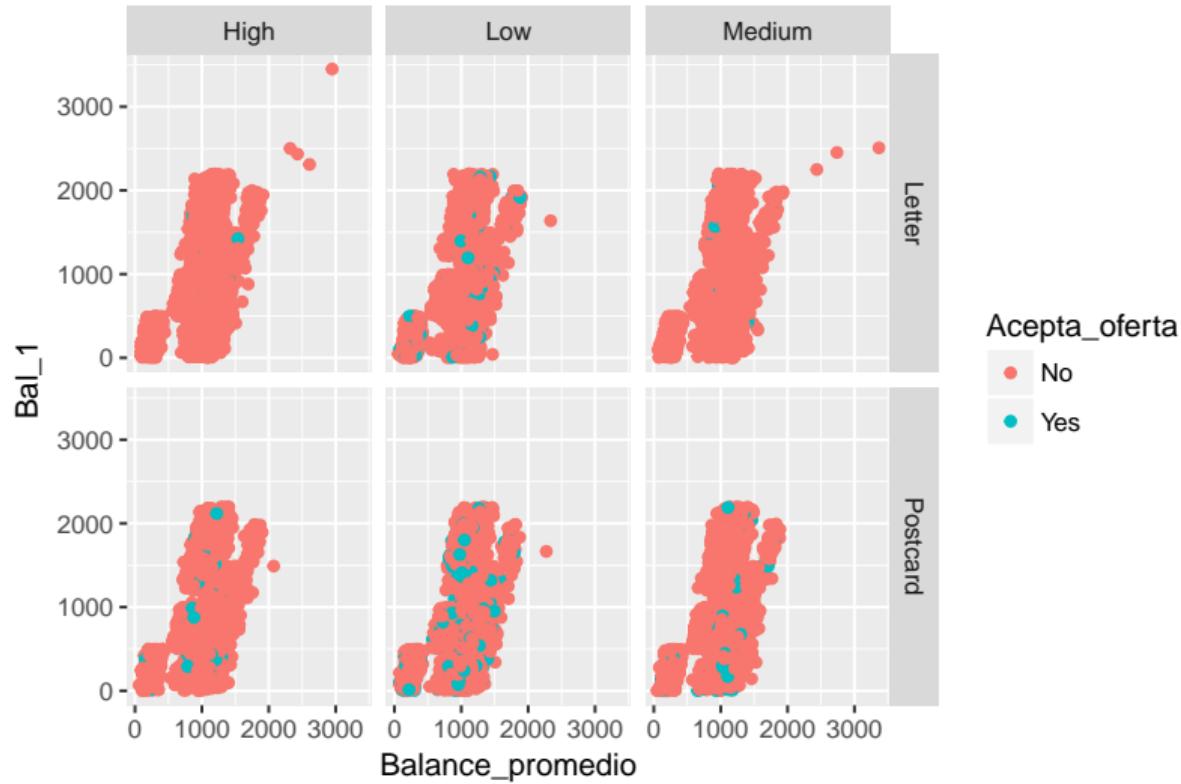
Ejercicio más variables

Repita el gráfico anterior, pero usando `facet_grid(Tipo_mensajeria~.)`. Repita el gráfico anterior pero usando `facet_grid(Tipo_mensaeria~Rating)`

Ejercicio más variables

Repita el gráfico anterior, pero usando `facet_grid(Tipo_mensajeria~.)`. Repita el gráfico anterior pero usando `facet_grid(Tipo_mensaeria~Rating)`, y si además colorea con `Acepta_oferta?`

Ejercicio más variables



Objetos

Todos los ggplots son objetos, por lo que no solo se obtiene un gráfico, sino que es fácil de conservar:

```
plot_1 <- ggplot(data = creditcardmarketing_bbm) +
  geom_histogram(mapping = aes(Balance_promedio),
                 bins = 30,na.rm = TRUE)
```

Objetos

Todos los ggplots son objetos, por lo que no solo se obtiene un gráfico, sino que es fácil de conservar:

```
plot_1 <- ggplot(data = creditcardmarketing_bbm) +
  geom_histogram(mapping = aes(Balance_promedio),
                 bins = 30,na.rm = TRUE)
```

Si llama plot_1 desde la consola se despliega en la pestaña de Plots.

Objetos

Todos los ggplots son objetos, por lo que no solo se obtiene un gráfico, sino que es fácil de conservar:

```
plot_1 <- ggplot(data = creditcardmarketing_bbm) +
  geom_histogram(mapping = aes(Balance_promedio),
                 bins = 30,na.rm = TRUE)
```

Si llama plot_1 desde la consola se despliega en la pestaña de Plots.

Graficar agregando capas

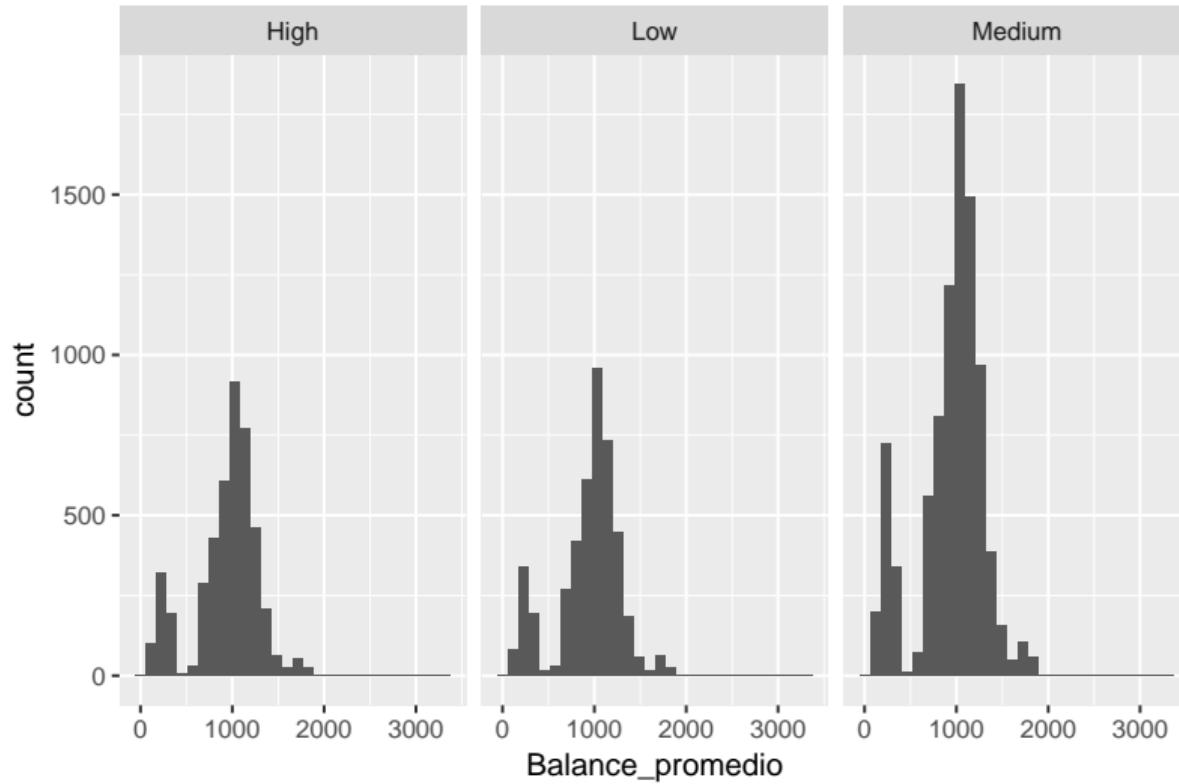
Graficar agregando capas

Si quiere realizar modificaciones, usa la misma sintaxis de siempre, agregando capas:

```
plot_1 +  
  facet_grid(.~Nivel_ingresos)
```

Esto se presta para ir desarrollando progresivamente los gráficos, y efectivamente ir explorando las relaciones que se puede presentar

Graficar agregando capas

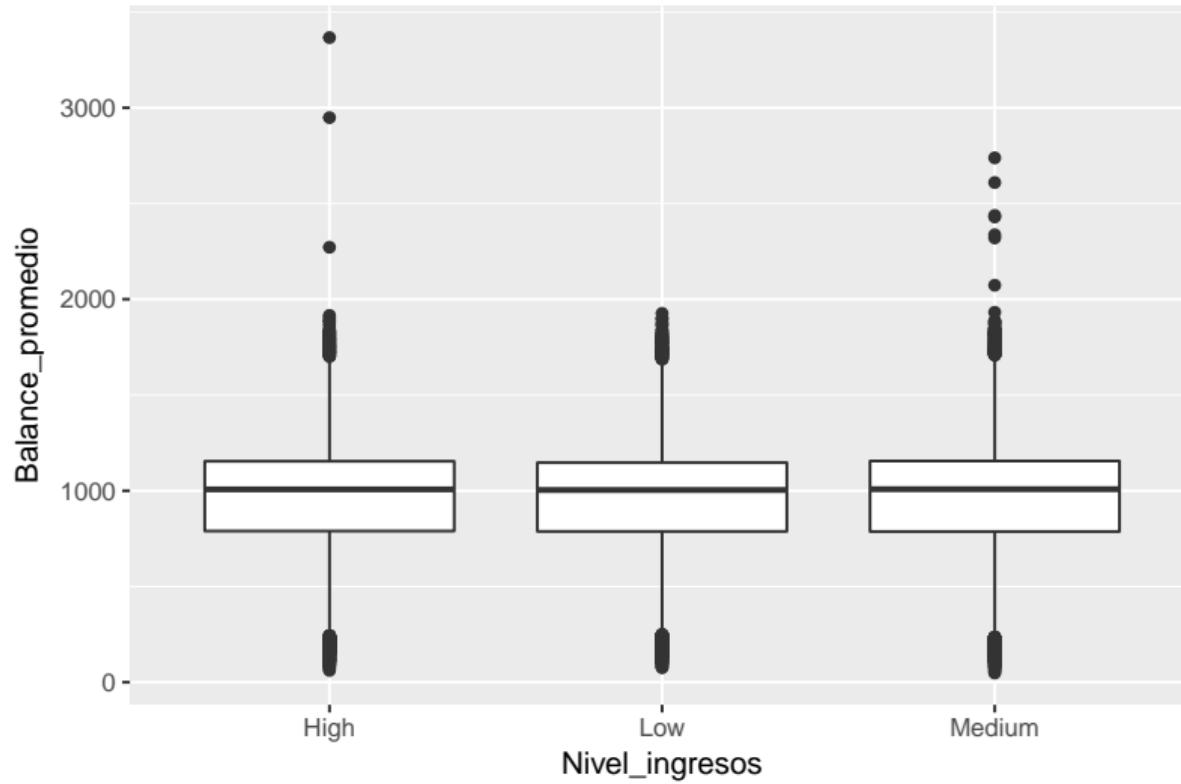


Box-plots

Una herramienta muy útil para graficar, son los boxplots, que ayudan a entender como están distribuidas las variables y cuales son “raras”

```
ggplot(data = creditcardmarketing_bbm) +
  geom_boxplot(mapping = aes(y = Balance_promedio,x = Nivel_ingresos),
               na.rm = TRUE)
```

Box-plots

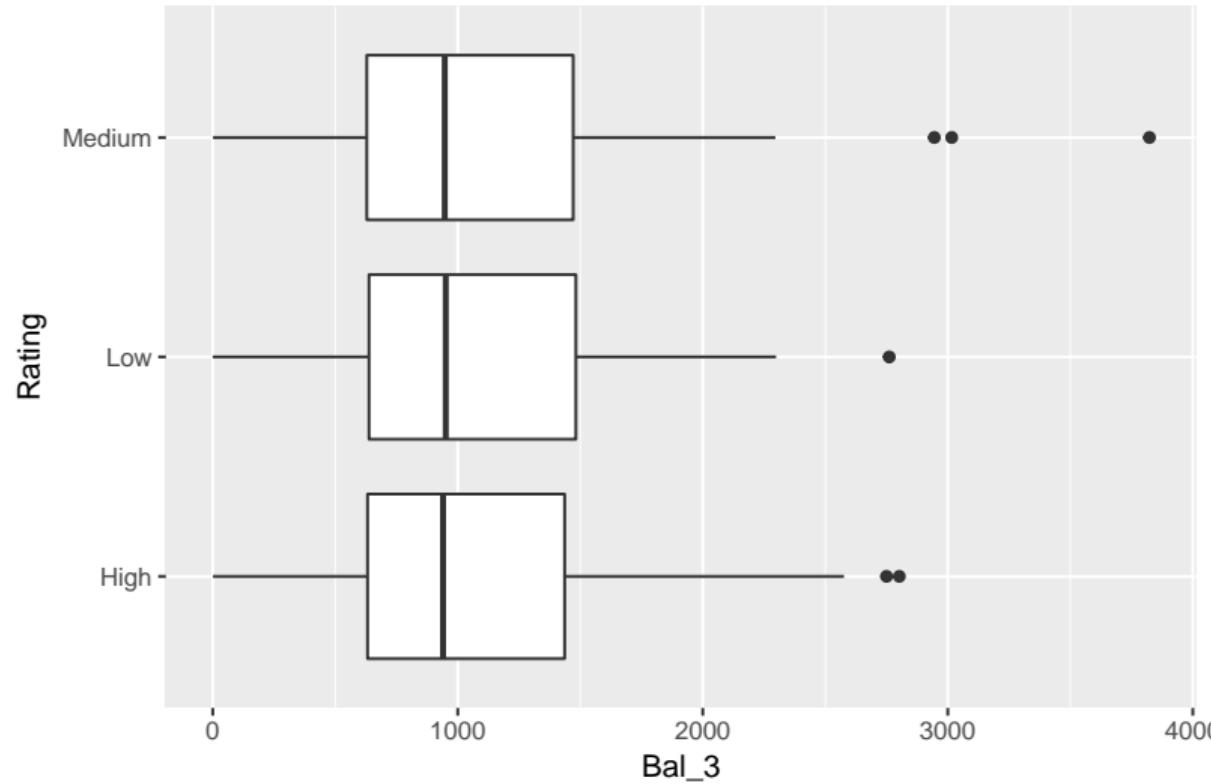


Girar coordenadas

Algunas veces, se tienen nombres muy largos para desplegar en el eje X, por lo que se pueden girar las coordenadas, para que se entiendan mejor, y se reciba la misma información:

```
ggplot(data = creditcardmarketing_bbm) +
  geom_boxplot(mapping = aes(y = Bal_3,x = Rating),
                na.rm = TRUE) +
  coord_flip()
```

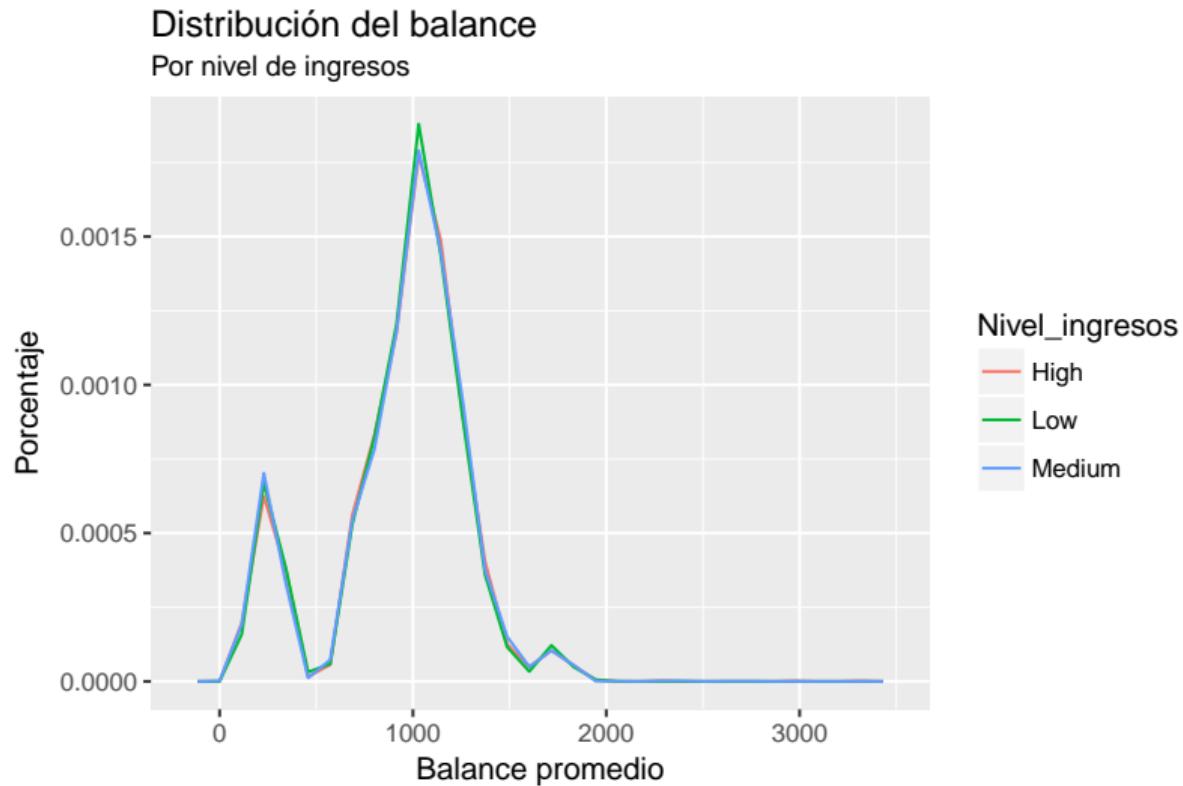
Girar coordenadas



Titulos y ejes

```
plot_titulo <- ggplot(data = creditcardmarketing_bbm) +
  geom_freqpoly(mapping = aes(x = Balance_promedio,
                               colour = Nivel_ingresos,
                               y = ..density..),
                na.rm = TRUE,bins = 30) +
  ggtitle('Distribución del balance',
          subtitle = 'Por nivel de ingresos') +
  xlab('Balance promedio') +
  ylab('Porcentaje')
```

Titulos y ejes



Etiquetas

Bueno, pero también sería bueno poder modificar el nombre del color, y para agregar una pequeña leyenda (caption) se puede usar el parámetro `caption`:

Etiquetas

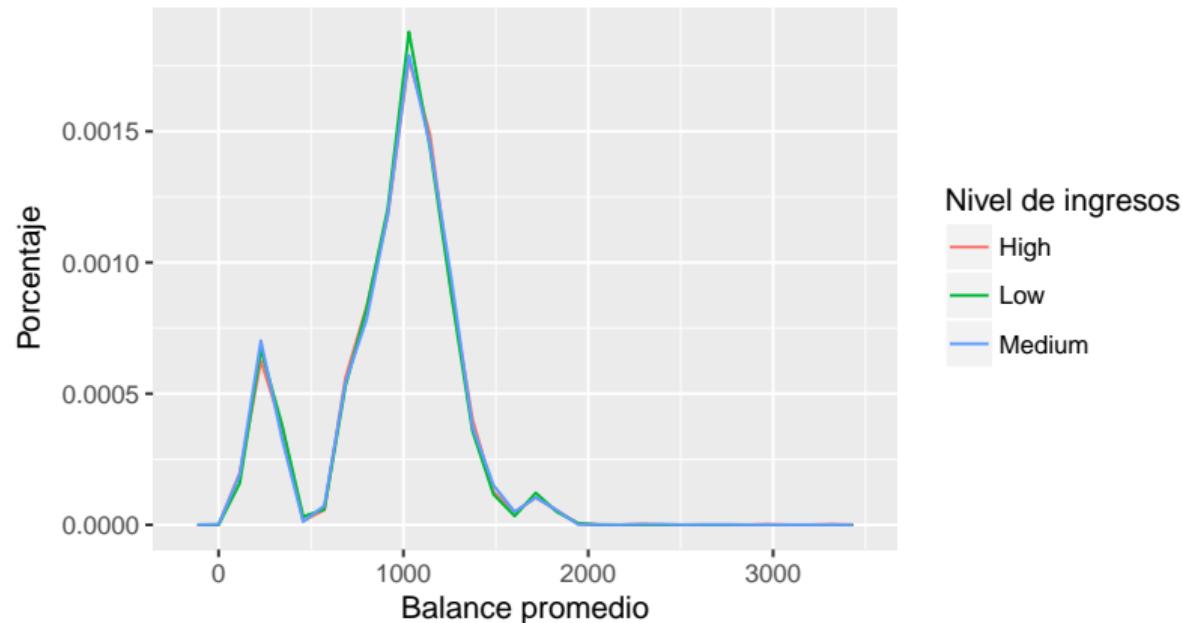
Bueno, pero también sería bueno poder modificar el nombre del color, y para agregar una pequeña leyenda (caption) se puede usar el parámetro `caption`:

```
plot_titulo +  
  labs(color = 'Nivel de ingresos',  
        caption = 'Fuente: data.world.com,\n Creacion propia')
```

Etiquetas

Distribución del balance

Por nivel de ingresos



Nivel de ingresos

- High
- Low
- Medium

Fuente: data.world.com,
Creacion propia

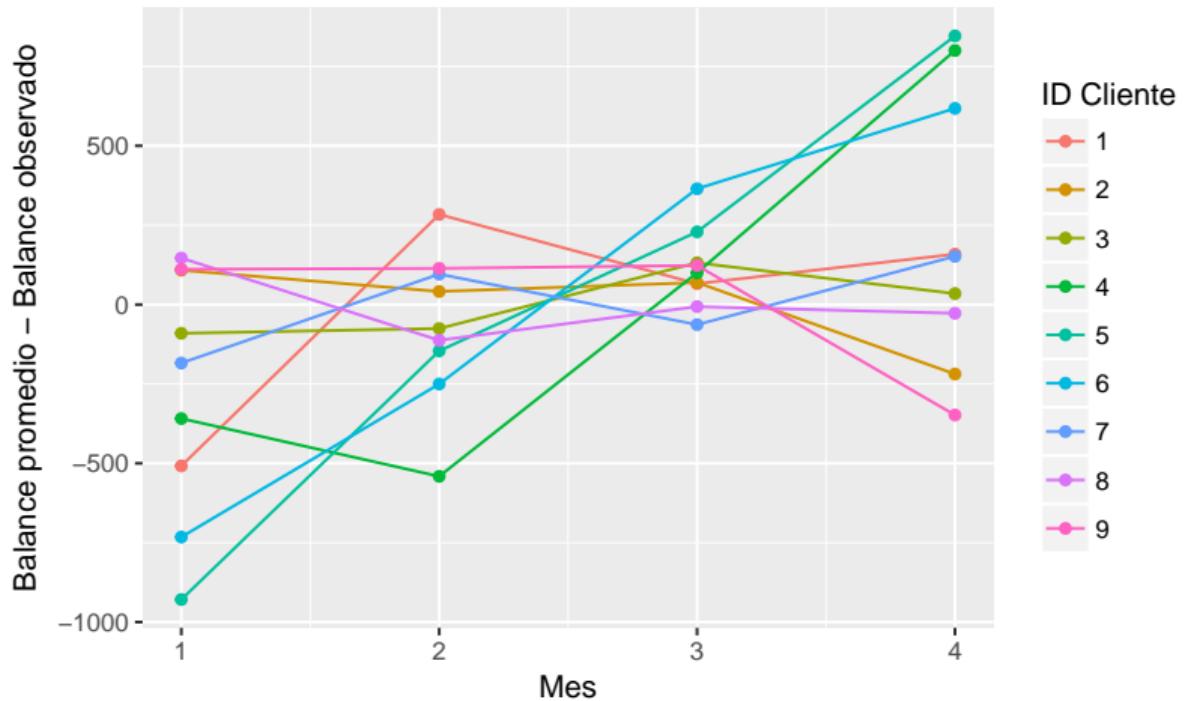
labs

Si se quiere, se pueden incluir todos los parámetros en la capa de labs:

```
creditos_Mes %>%
  mutate(Balance_dif = Balance_promedio - Balance) %>%
  ggplot(mapping = aes(x = Mes, y = Balance_dif,
                        color = factor(Num_cliente))) +
  geom_line() +
  geom_point() +
  labs(title = 'Historia del balance',
       subtitle = 'Por persona',
       x = 'Mes',
       y = 'Balance promedio - Balance observado',
       color = 'ID Cliente')
```

Historia del balance

Por persona



theme

Hay varios temas que se pueden aplicar a los gráficos, para cambiar varios parámetros predefinidos. Este se agrega como una capa más. Están:

- ▶ `theme_bw()`

Y hay paquetes para incluir aún más temas...

theme

Hay varios temas que se pueden aplicar a los gráficos, para cambiar varios parámetros predefinidos. Este se agrega como una capa más. Están:

- ▶ `theme_bw()`
- ▶ `theme_classic()`

Y hay paquetes para incluir aún más temas...

theme

Hay varios temas que se pueden aplicar a los gráficos, para cambiar varios parámetros predefinidos. Este se agrega como una capa más. Están:

- ▶ `theme_bw()`
- ▶ `theme_classic()`
- ▶ `theme_dark()`

Y hay paquetes para incluir aún más temas...

theme

Hay varios temas que se pueden aplicar a los gráficos, para cambiar varios parámetros predefinidos. Este se agrega como una capa más. Están:

- ▶ `theme_bw()`
- ▶ `theme_classic()`
- ▶ `theme_dark()`
- ▶ `theme_get()`

Y hay paquetes para incluir aún más temas...

theme

Hay varios temas que se pueden aplicar a los gráficos, para cambiar varios parámetros predefinidos. Este se agrega como una capa más. Están:

- ▶ `theme_bw()`
- ▶ `theme_classic()`
- ▶ `theme_dark()`
- ▶ `theme_get()`
- ▶ `theme_gray()`

Y hay paquetes para incluir aún más temas...

theme

Hay varios temas que se pueden aplicar a los gráficos, para cambiar varios parámetros predefinidos. Este se agrega como una capa más. Están:

- ▶ `theme_bw()`
- ▶ `theme_classic()`
- ▶ `theme_dark()`
- ▶ `theme_get()`
- ▶ `theme_gray()`
- ▶ `theme_light()`

Y hay paquetes para incluir aún más temas...

theme

Hay varios temas que se pueden aplicar a los gráficos, para cambiar varios parámetros predefinidos. Este se agrega como una capa más. Están:

- ▶ `theme_bw()`
- ▶ `theme_classic()`
- ▶ `theme_dark()`
- ▶ `theme_get()`
- ▶ `theme_gray()`
- ▶ `theme_light()`
- ▶ `theme_linedraw()`

Y hay paquetes para incluir aún más temas...

theme

Hay varios temas que se pueden aplicar a los gráficos, para cambiar varios parámetros predefinidos. Este se agrega como una capa más. Están:

- ▶ `theme_bw()`
- ▶ `theme_classic()`
- ▶ `theme_dark()`
- ▶ `theme_get()`
- ▶ `theme_gray()`
- ▶ `theme_light()`
- ▶ `theme_linedraw()`
- ▶ `theme_minimal()`

Y hay paquetes para incluir aún más temas...

Escala

Escala



Figure 4: Pero no montañas

Escala

Escala

Muchas veces se quieren modificar los parámetros que se tienen predefinidos. Es por esto que se define la capa de escalas, para poder cambiar a gusto del usuario/programador/analista estos valores predefinidos.

Escala

Muchas veces se quieren modificar los parámetros que se tienen predefinidos. Es por esto que se define la capa de escalas, para poder cambiar a gusto del usuario/programador/analista estos valores predefinidos.

```
plot_2 <- ggplot(data = creditcardmarketing_bbm) +
  geom_bar(mapping = aes(x = Num_cuentas, fill=Acepta_oferta))
```

¿Cómo se ve el gráfico anterior?

Escala

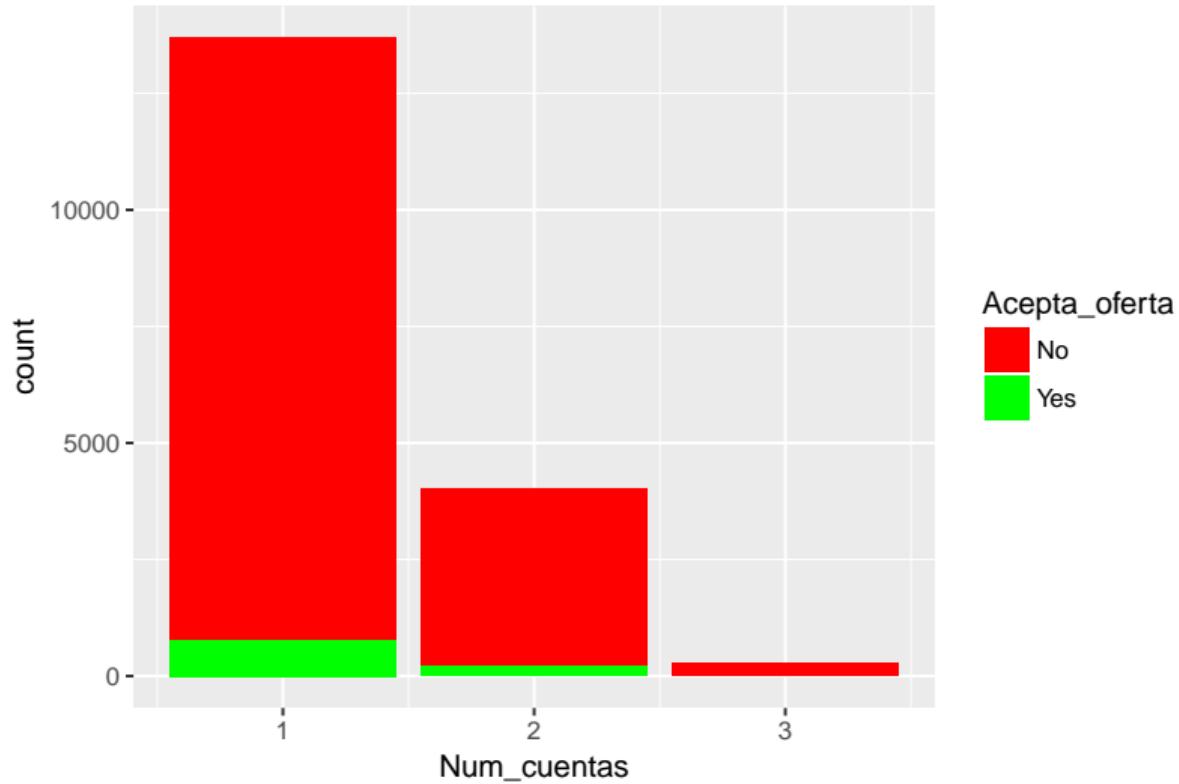
Muchas veces se quieren modificar los parámetros que se tienen predefinidos. Es por esto que se define la capa de escalas, para poder cambiar a gusto del usuario/programador/analista estos valores predefinidos.

```
plot_2 <- ggplot(data = creditcardmarketing_bbm) +
  geom_bar(mapping = aes(x = Num_cuentas, fill=Acepta_oferta))
```

¿Cómo se ve el gráfico anterior? Para cambiar los colores que se usan, e incluso las viñetas asociadas a los colores, se usa `scale_fill_manual`

```
plot_2 +
  scale_fill_manual(values = c('red','green'))
```

Escala



Escala, etiquetas

Si además se le quieren cambiar las etiquetas que se usan, en lugar de 'No' y 'Yes'.
Se puede usar el parámetro

Escala, etiquetas

Si además se le quieren cambiar las etiquetas que se usan, en lugar de 'No' y 'Yes'. Se puede usar el parámetro `labels`, en el mismo orden que se usa para `values`.

```
plot_2 +  
  scale_fill_manual(values = c('red','green'),  
                    labels = c('Noup','Sip'))
```

Escala, etiquetas plot

```
plot_2 +  
  scale_fill_manual(values = c('red','green'),  
                    labels = c('Noup','Sip'))
```

Escalas, aún más!

Algunas veces es informativo cambiar la escala de alguno de los ejes. Por ejemplo, pasarlo a base logarítmica, raíz cuadrada, revertirlo, entre otras... Por ejemplo:

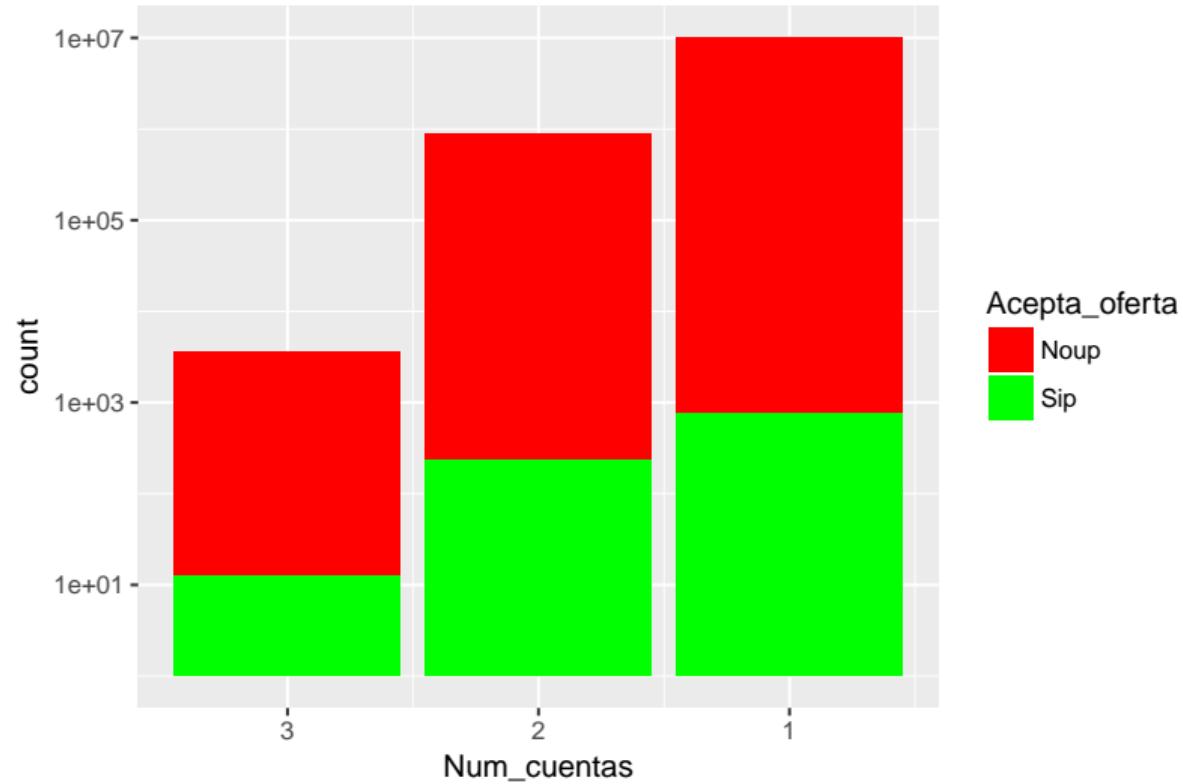
Escalas, aún más!

Algunas veces es informativo cambiar la escala de alguno de los ejes. Por ejemplo, pasarlo a base logarítmica, raíz cuadrada, revertirlo, entre otras... Por ejemplo:

```
plot_2 +  
  scale_fill_manual(values = c('red','green'),  
                    labels = c('Noup','Sip')) +  
  scale_y_log10() +  
  scale_x_reverse()
```

Escalas, aún más, gráfico:

Escalas, aún más, gráfico:



Ejercicio

Repita el gráfico anterior, quitando cada capa al menos una vez, note las diferencias.

Ejercicio

Repita el gráfico anterior, quitando cada capa al menos una vez, note las diferencias.
Repita uno de los gráficos anteriores y cambie el `scale_y_log10` por `scale_y_sqrt`.

Ejercicio

Repita el gráfico anterior, quitando cada capa al menos una vez, note las diferencias.

Repita uno de los gráficos anteriores y cambie el `scale_y_log10` por `scale_y_sqrt`.

Regrese al gráfico en que se hace el `geom_bin2d`, y agregue una capa usando:

`scale_fill_distiller`. Esta función recibe varios parámetros, la idea es que utilice el parámetro `palette` y use alguno de las siguientes strings como valor:

'Spectral', 'Set1', 'Set2', 'Set3', 'Greens'.

Ejercicio

Repita el gráfico anterior, quitando cada capa al menos una vez, note las diferencias.

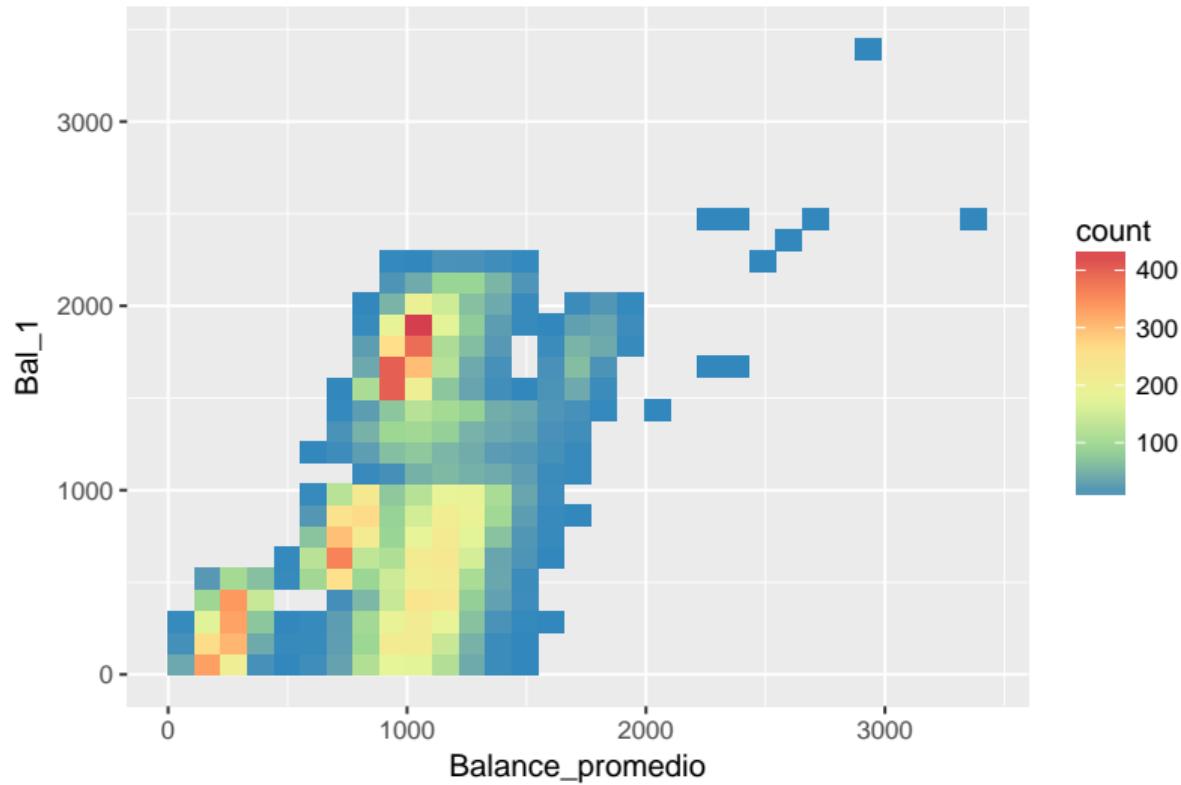
Repita uno de los gráficos anteriores y cambie el `scale_y_log10` por `scale_y_sqrt`.

Regrese al gráfico en que se hace el `geom_bin2d`, y agregue una capa usando:

`scale_fill_distiller`. Esta función recibe varios parámetros, la idea es que utilice el parámetro `palette` y use alguno de las siguientes strings como valor:

'Spectral', 'Set1', 'Set2', 'Set3', 'Greens'. ¿Qué pasa si además incluye el parámetro `direction = -1`? ¿Y si pone `direction = -7`?

Ejercicio fill



Más escalas...

Existen escalas para un montón de parámetros más. Por ejemplo, el tipo de linea (`scale_linetype`), la transparencia (`scale_alpha`), el tamaño `scale_size`, la forma (este es un parámetro de `geom_point`) usando `scale_shape`.

Y hay otro montón de formas de manipular las escalas...

Estadísticas

Estadísticas

Algunas veces se quieren agregar funciones para realizar una comparación entre los datos con una función particular.

Por ejemplo, si queremos comparar los balances históricos de la persona 1 con la función lineal $f(x) = a*x+b$, con $a = 570$ y $b = -1425$. Definimos la función anterior.

Estadísticas

Algunas veces se quieren agregar funciones para realizar una comparación entre los datos con una función particular.

Por ejemplo, si queremos comparar los balances históricos de la persona 1 con la función lineal $f(x) = a*x+b$, con $a = 570$ y $b = -1425$. Definimos la función anterior.

```
linea_5 <- function(x){-1425 + 570*x}
```

Estadísticas

Algunas veces se quieren agregar funciones para realizar una comparación entre los datos con una función particular.

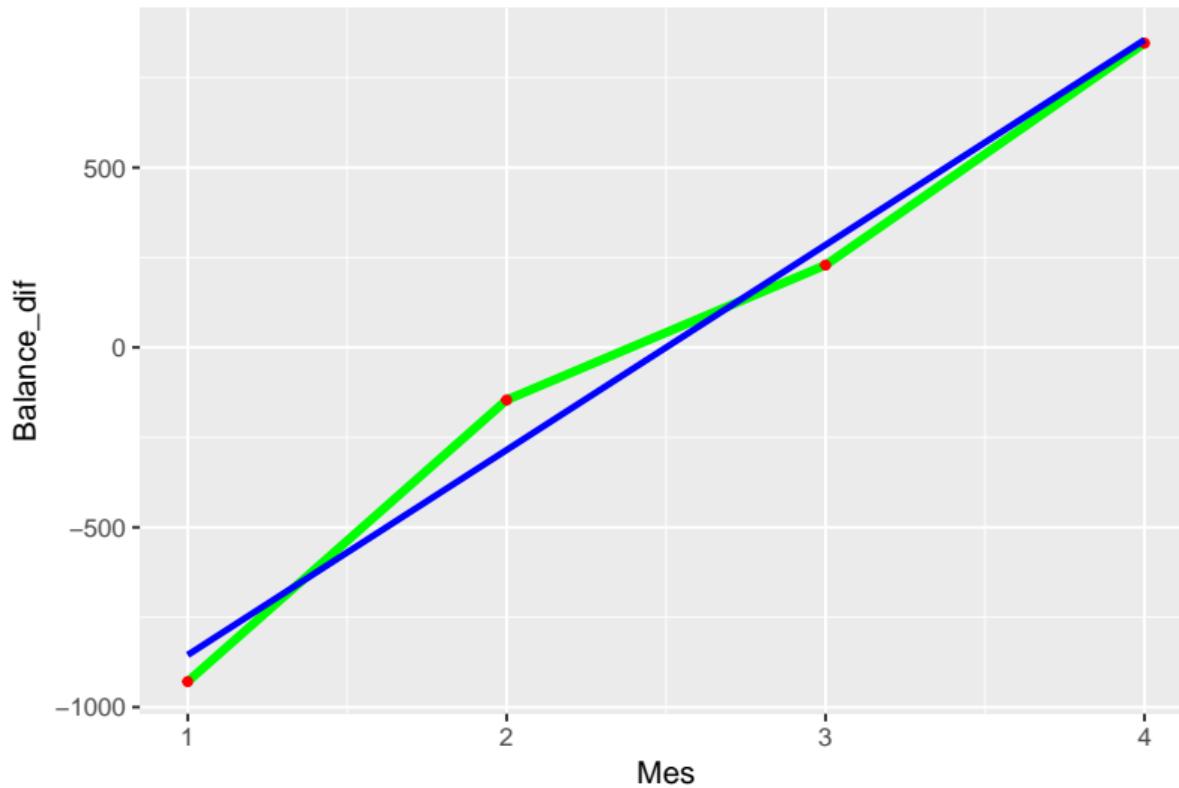
Por ejemplo, si queremos comparar los balances históricos de la persona 1 con la función lineal $f(x) = a*x+b$, con $a = 570$ y $b = -1425$. Definimos la función anterior.

```
linea_5 <- function(x){-1425 + 570*x}
```

Para evaluarlo, usamos:

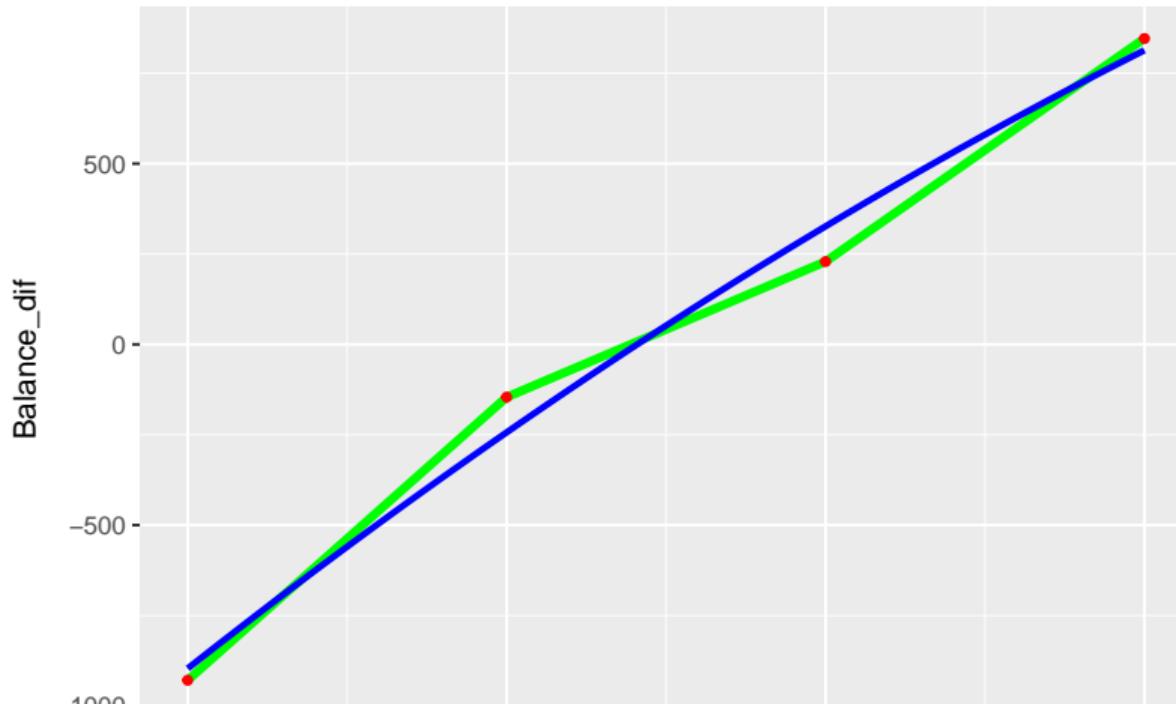
```
plot_3 <- creditos_Mes %>%
  filter(Num_cliente == 5) %>%
  mutate(Balance_dif = Balance_promedio - Balance) %>%
  ggplot(mapping = aes(x = Mes,y = Balance_dif)) +
  geom_line(size = 1.5,color = 'green') +
  geom_point(size = 1.2,color= 'red') +
  stat_function(fun = linea_5,color = 'blue',size = 1.1)
```

Estadísticas



Ejercicio

Repita el gráfico anterior, pero usando una cuadrática $g(x) = a*x^2 + b*x + c$, con $a = -41.5$, $b = 777.5$, $c = -1632.5$. ¿cuál gráfico se ve como una mejor aproximación?

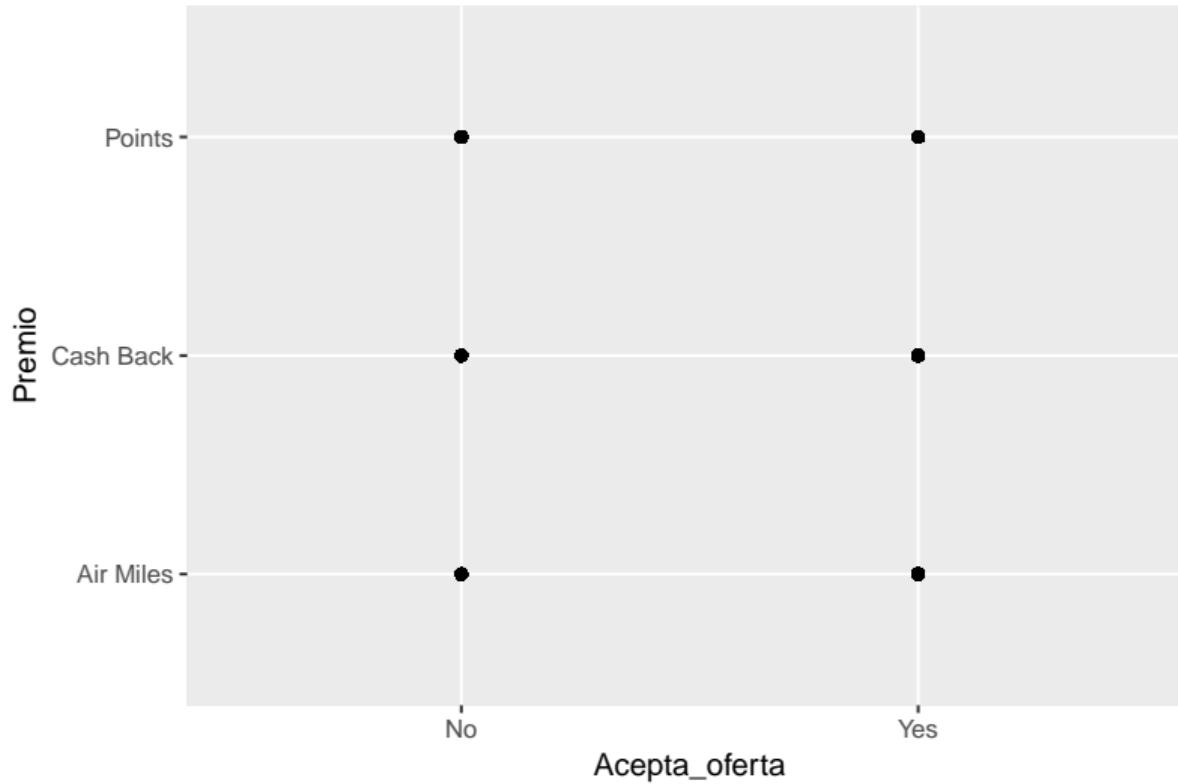


stat_unique

Esta función lo que hace es que elimina los valores repetidos del gráfico:

```
creditcardmarketing_bbm %>%  
  ggplot(aes(x=Acepta_oferta,y = Premio)) +  
  geom_point() +  
  stat_unique()
```

stat_unique, gráfico



Coordenadas

Como en un mapa.

Coordenadas

Como en un mapa.

La mayoría de gráficos que se hacen (como los de puntos), tienen coordenadas en sus ejes (X, Y). Esta capa se encarga de modificar el comportamiento predefinido. Funciona para hacer *zoom* a una parte del gráfico, o alejarse de una sección, modificar perspectivas, o realizar transformaciones.

zoom

ZOOM

Esto se hace usando la capa `coord_cartesian`, la cual tiene 2 parámetros: `xlim` y `ylim`. Estos toman un vector con dos entradas, las cuales indican en qué sección del gráfico original se van a concentrar.

```
plot_5 <- creditcardmarketing_bbm %>%
  ggplot(aes(x = Balance_promedio)) +
  geom_histogram(fill = 'white', color = 'black',
                 na.rm = TRUE)
```

¿Cómo se ve este gráfico?

ZOOM

Esto se hace usando la capa `coord_cartesian`, la cual tiene 2 parámetros: `xlim` y `ylim`. Estos toman un vector con dos entradas, las cuales indican en qué sección del gráfico original se van a concentrar.

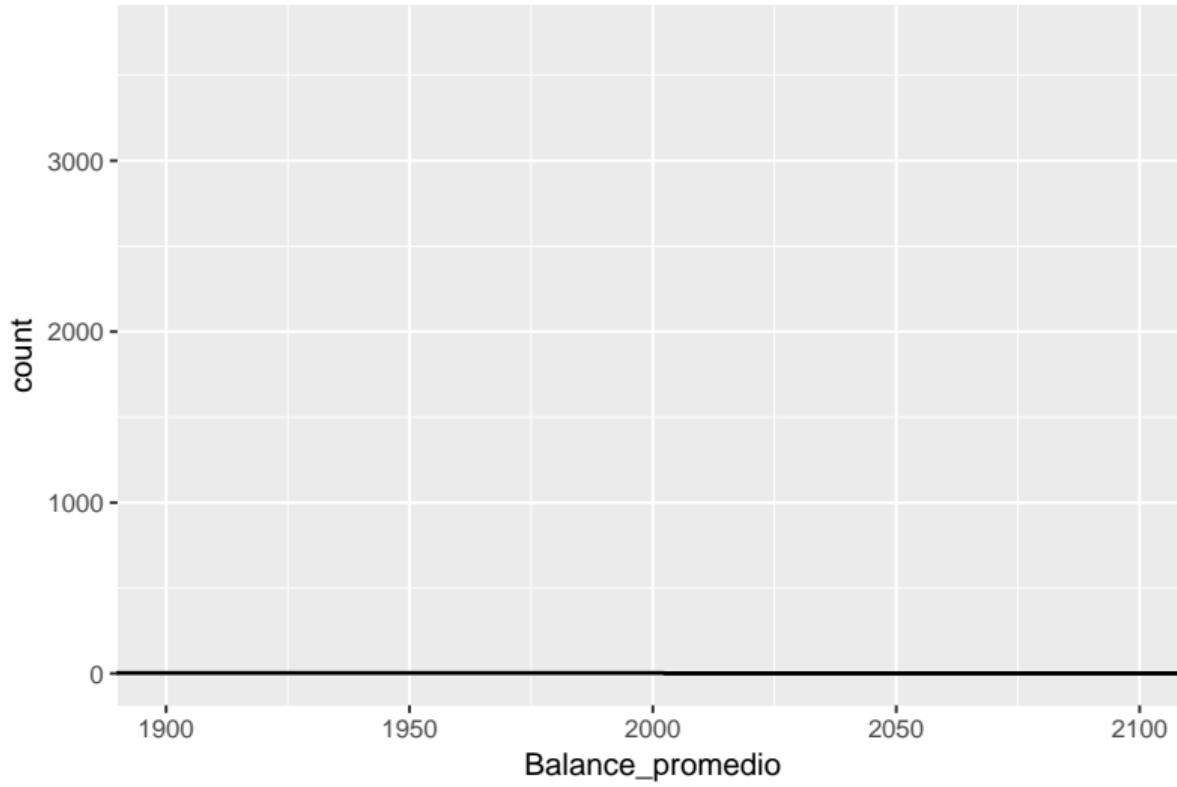
```
plot_5 <- creditcardmarketing_bbm %>%
  ggplot(aes(x = Balance_promedio)) +
  geom_histogram(fill = 'white', color = 'black',
                 na.rm = TRUE)
```

¿Cómo se ve este gráfico? Para hacerle *zoom* al área cerca del 2000, hacemos:

```
plot_5 +
  coord_cartesian(xlim = c(1900, 2100))
```

zoom

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



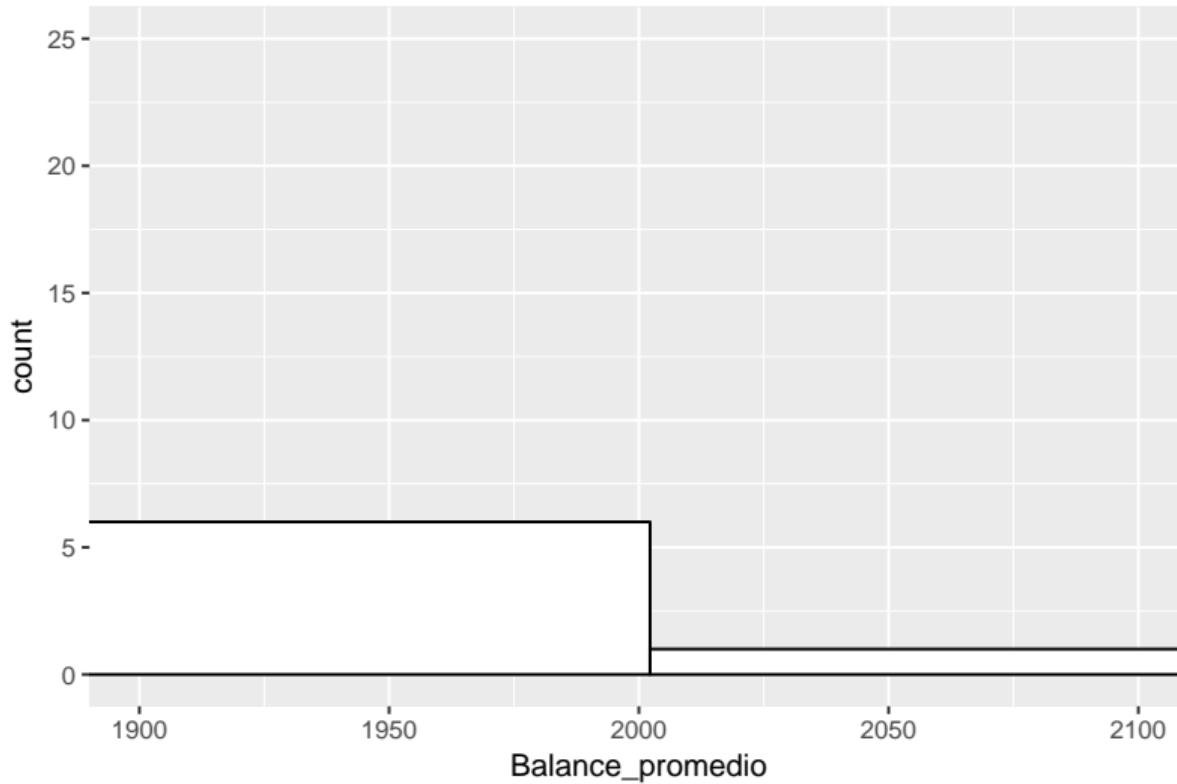
zoom

El gráfico anterior, aún no se ve bien. Pues falta hacerle zoom también al eje y, por lo que hacemos:

```
plot_5 +  
  coord_cartesian(xlim = c(1900,2100), ylim = c(0,50))
```

zoom

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



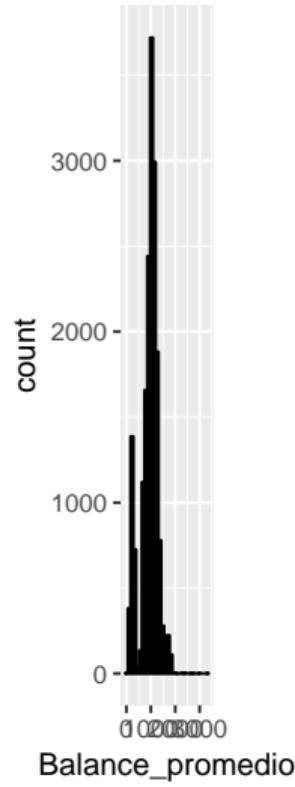
Proporcional

Esta capa recibe el parámetro `ratio`, el cual se encarga de indicar cuanto vale el tamaño a mostrar del eje y sobre el del eje x.

```
plot_5 +  
  coord_fixed(ratio = 7)
```

Proporcional

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Ejercicios proporcional

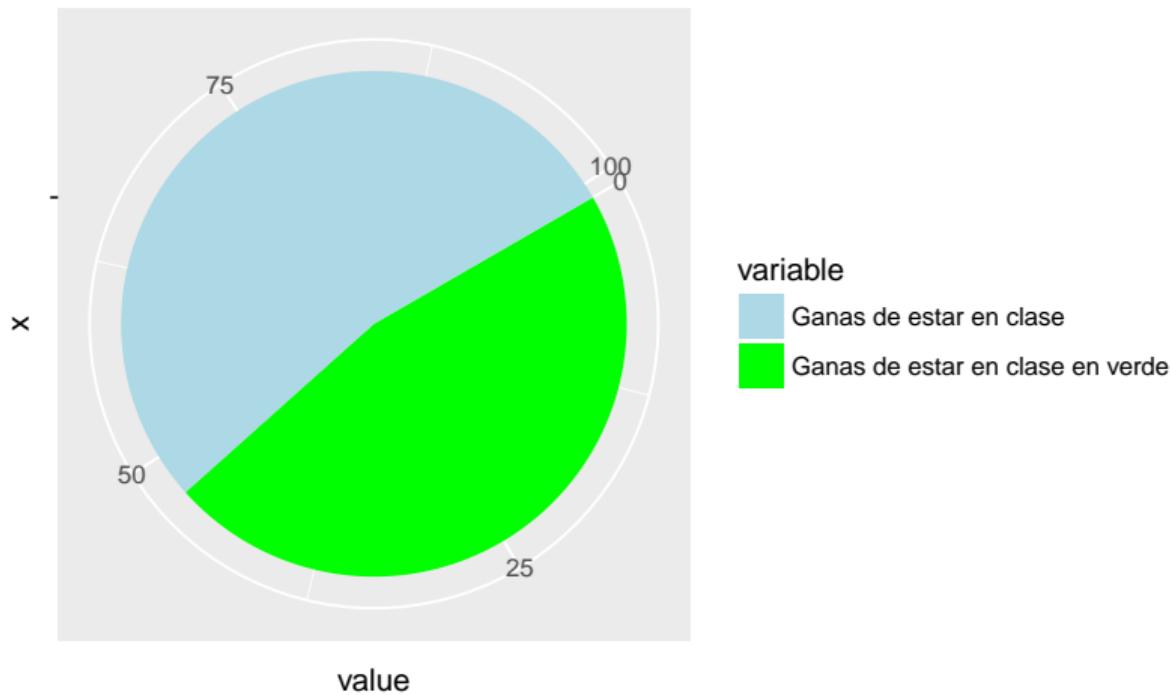
Repita el gráfico anterior usando `ratio = 1/7`, luego `ratio = 1`, y `ratio = 1/3`.

coord_polar

Sirve para hacer gráficos de porcentajes. Los cuales pocas veces son recomendados, pues los humanos somos malos distinguiendo ángulos. Muchas veces es mejor usar un gráfico de barras sencillo:

```
df <- data.frame(  
  variable = c("Ganas de estar en clase en verde",  
              "Ganas de estar en clase"),  
  value = c(47.19, 53.81)  
)  
  
ggplot(df, aes(x = "", y = value, fill = variable)) +  
  geom_col(width = 1) +  
  scale_fill_manual(values = c("lightblue", "green")) +  
  coord_polar("y", start = pi / 3)
```

coord_polar, gráfico



Ejercicio de coord_polar

Repita el gráfico anterior, pero quite la capa de coord_polar.

Ejercicio de coord_polar

Repita el gráfico anterior, pero quite la capa de coord_polar. ¿cuál gráfico entiende mejor?

Ejercicio de coord_polar

Repita el gráfico anterior, pero quite la capa de coord_polar. ¿cuál gráfico entiende mejor?

Haga un gráfico similar al anterior, pero usando 3 grupos distintos.

Ejercicio de coord_polar

Repita el gráfico anterior, pero quite la capa de coord_polar. ¿cuál gráfico entiende mejor?

Haga un gráfico similar al anterior, pero usando 3 grupos distintos.

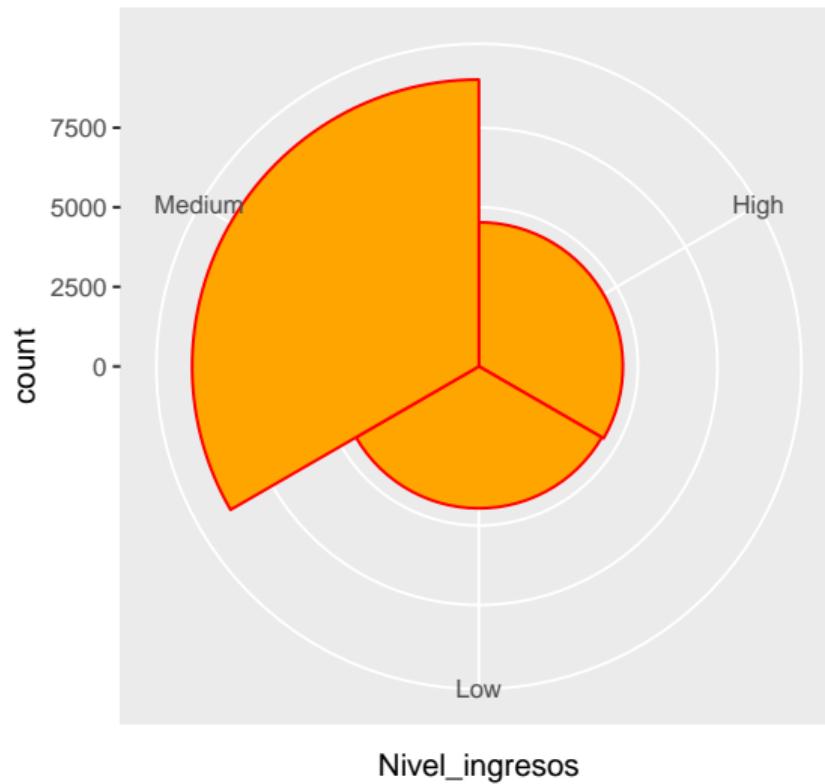
Haga un pac-man

coord_polar

Si se quieren “comparar” tamaños de grupos, usando este tipo de gráficos, se puede hacer:

```
creditcardmarketing_bbm %>%
  ggplot(aes(x = Nivel_ingresos)) +
  geom_bar(width = 1, colour = 'red', fill = 'orange') +
  coord_polar()
```

Gráfico



Tipos de capas:

En general hay 5 capas generales para generar un gráfico usando ggplot2

Tipos de capas:

En general hay 5 capas generales para generar un gráfico usando ggplot2

- ▶ `geom_*`, que indica la forma en que se van a graficar los datos
 - *¿cuales hemos visto?*

Tipos de capas:

En general hay 5 capas generales para generar un gráfico usando ggplot2

- ▶ `geom_*`, que indica la forma en que se van a graficar los datos
 - *¿cuales hemos visto?*
- ▶ `stat_*`, que realiza una transformación (resumen) en los datos y lo grafica

Tipos de capas:

En general hay 5 capas generales para generar un gráfico usando ggplot2

- ▶ `geom_*`, que indica la forma en que se van a graficar los datos
 - *¿cuales hemos visto?*
- ▶ `stat_*`, que realiza una transformación (resumen) en los datos y lo grafica
- ▶ `scale_*_**`, sirva para cambiar algún parámetro predefinido

Tipos de capas:

En general hay 5 capas generales para generar un gráfico usando ggplot2

- ▶ `geom_*`, que indica la forma en que se van a graficar los datos
 - *¿cuales hemos visto?*
- ▶ `stat_*`, que realiza una transformación (resumen) en los datos y lo grafica
- ▶ `scale_*_**`, sirva para cambiar algún parámetro predefinido
- ▶ `coord_*`, cambia el tipo/tamaño de las coordenadas que se usan

Tipos de capas:

En general hay 5 capas generales para generar un gráfico usando ggplot2

- ▶ `geom_*`, que indica la forma en que se van a graficar los datos
 - *¿cuales hemos visto?*
- ▶ `stat_*`, que realiza una transformación (resumen) en los datos y lo grafica
- ▶ `scale_*_**`, sirva para cambiar algún parámetro predefinido
- ▶ `coord_*`, cambia el tipo/tamaño de las coordenadas que se usan
- ▶ `facet_*`, puede ser `grid` o `wrap`

Ejercicios

Ejercicios

En los gráficos que usan `geom_line`, explore usar los parámetros `linetype` y `size`.

Ejercicios

En los gráficos que usan `geom_line`, explore usar los parámetros `linetype` y `size`.

En los gráficos de `geom_bar`, explore usando el parámetro `fill` dentro de `aes`. ¿Qué pasa si pone `position='dodge'`, como parámetro dentro de un `geom_bar`?

Ejercicios

En los gráficos que usan `geom_line`, explore usar los parámetros `linetype` y `size`.

En los gráficos de `geom_bar`, explore usando el parámetro `fill` dentro de `aes`. ¿Qué pasa si pone `position='dodge'`, como parámetro dentro de un `geom_bar`?

Juegue con los temas y los gráficos que ha hecho. Use al menos una vez cada tema.
¿Cuál le gusta más?

Ejercicios

En los gráficos que usan `geom_line`, explore usar los parámetros `linetype` y `size`.

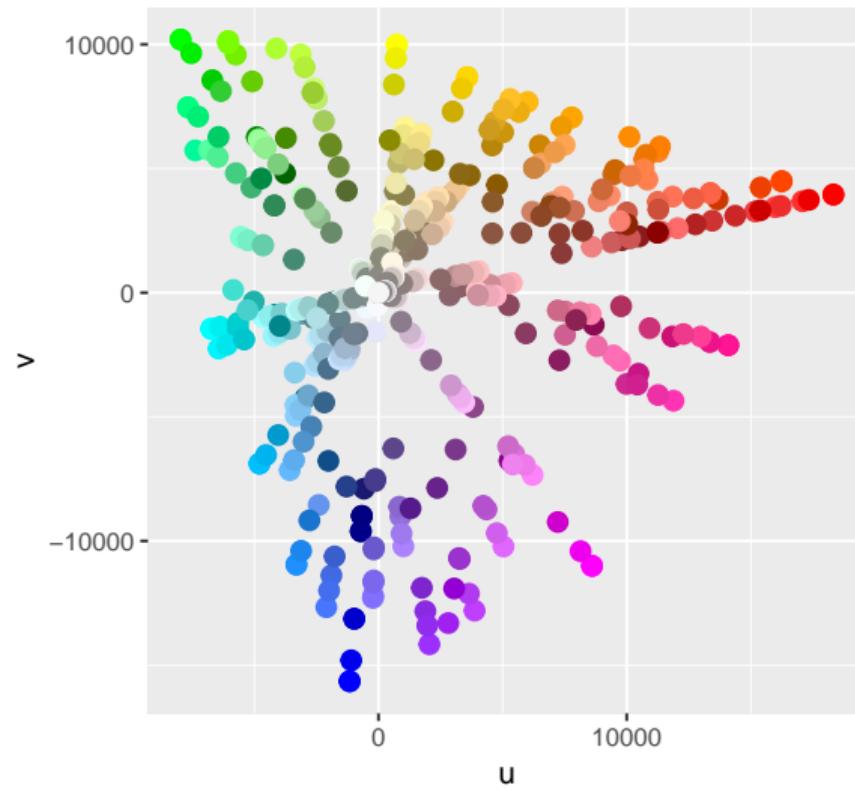
En los gráficos de `geom_bar`, explore usando el parámetro `fill` dentro de `aes`. ¿Qué pasa si pone `position='dodge'`, como parámetro dentro de un `geom_bar`?

Juegue con los temas y los gráficos que ha hecho. Use al menos una vez cada tema.
¿Cuál le gusta más?

Lea sobre `ggsave`, usando la ayuda que se provee en R

Más colores?

Ver dataframe luv_colours



Más información sobre ggplot2

Sitio de referencia de ggplot2

Próximamente:

optim