Bellabeat

Rafael Rodriguez

2024-09-05

Context

You are a junior data analyst working on the marketing analyst team at Bellabeat, a high-tech manufacturer of health-focused products for women. Bellabeat is a successful small company, but they have the potential to become a larger player in the global smart device market. Urška Sršen, cofounder and Chief Creative Officer of Bellabeat, believes that analyzing smart device fitness data could help unlock new growth opportunities for the company. You have been asked to focus on one of Bellabeat's products and analyze smart device data to gain insight into how consumers are using their smart devices. The insights you discover will then help guide marketing strategy for the company. You will present your analysis to the Bellabeat executive team along with your high-level recommendations for Bellabeat's marketing strategy. By 2016, Bellabeat had opened offices around the world and launched multiple products.

Bellabeat app: The Bellabeat app provides users with health data related to their activity, sleep, stress, menstrual cycle, and mindfulness habits. This data can help users better understand their current habits and make healthy decisions. The Bellabeat app connects to their line of smart wellness products. Leaf: Bellabeat's classic wellness tracker can be worn as a bracelet, necklace, or clip. The Leaf tracker connects to the Bellabeat app to track activity, sleep, and stress. Time: This wellness watch combines the timeless look of a classic timepiece with smart technology to track user activity, sleep, and stress. The Time watch connects to the Bellabeat app to provide you with insights into your daily wellness. Spring: This is a water bottle that tracks daily water intake using smart technology to ensure that you are appropriately hydrated throughout the day. The Spring bottle connects to the Bellabeat app to track your hydration levels. Bellabeat membership: Bellabeat also offers a subscription-based membership program for users. Membership gives users 24/7 access to fully personalized guidance on nutrition, activity, sleep, health and beauty, and mindfulness based on their lifestyle and goals.

Who is your audience?

- Urška Sršen: Bellabeat's cofounder and Chief Creative Officer
- Sando Mur: Mathematician and Bellabeat's cofounder; key member of the Bellabeat executive team.
- Bellabeat marketing analytics team: A team of data analysts responsible for collecting, analyzing, and reporting data

Problems

1. What are some trends in smart device usage?
2. How could these trends apply to Bellabeat customers?
3. How could these trends help influence Bellabeat marketing strategy?

What type of data should I collect?

There are 13 files csv - Daily_Activity_merged - Heart_Rate_seconds_merged - Hourly_Calories_merged - Hourly_Intensities_merged - Hourly_Steps_merged - Minute_Calories_Narrow_merged - Minute_Intensities_Narrow_mer - Minute_METs_Narrow_merged - Minute_Sleep_merged - Minute_Steps_Narrow_merged - Weight_Log_Info_merged

Hosted

https://www.kaggle.com/datasets/arashnic/fitbit

Licensed

https://www.kaggle.com/datasets/arashnic/fitbit dataset presents restrictions agree to license-agreement

Review of dataset

We used the function skim reviewing each dataset

Modify the dataset features

We made the following changes to the dataset

```
A1 <- A %>%
  mutate(id = as.character(Id)) %>%
  mutate(date1 = mdy(ActivityDate)) %>%
  mutate(date = format(date1,"%d/%m/%y")) %>%
  mutate(year = year(date1)) %>%
  mutate(month = month(date1)) %>%
  mutate(day = day(date1)) %>%
  mutate(day_month = paste(month, day, sep = "-")) %>%
  select(id, date, year, day_month, TotalSteps, TotalDistance, TrackerDistance, Calories, date1) %>%
  mutate(day_month = factor(day_month, levels = unique(day_month[order(date1)])))  # Ordena los niveles

B1 <- B %>%
  rename(heart_rate = Value) %>%
  mutate(id = as.character(Id)) %>%
  mutate(date1 = mdy_hms(Time)) %>%
  mutate(date = format(date1,"%d/%m/%y %H:%M:%S")) %>%
  mutate(year = year(date1)) %>%
  mutate(month = month(date1)) %>%
  mutate(day = day(date1)) %>%
  mutate(time = format(date1, "%H:%M:%S")) %>%
  mutate(wday = weekdays(date1)) %>%
  mutate(day_month = paste(month, day, wday, sep = "-")) %>%
  select(day_month,heart_rate, date1) %>%
  mutate(day_month = factor(day_month, levels = unique(day_month[order(date1)])))  # Ordena los niveles


C1 <- C %>%
  mutate(id = as.character(Id)) %>%
  mutate(date1 = mdy_hms(ActivityHour)) %>%
```

```
  mutate(date = format(date1,"%d/%m/%y %H:%M:%S")) %>%
  mutate(year = year(date1)) %>%
  mutate(month = month(date1)) %>%
  mutate(day = day(date1)) %>%
  mutate(time = format(date1, "%H:%M:%S")) %>%
  mutate(day_month = paste(month, day, sep = "-")) %>%
  select(id,day_month, Calories, date1) %>%
  mutate(day_month = factor(day_month, levels = unique(day_month[order(date1)])))  # Ordena los niveles


D1 <- D %>%
  rename(Average_Intensity = AverageIntensity) %>%
  rename(Total_Intensity = TotalIntensity) %>%
  mutate(id = as.character(Id)) %>%
  mutate(date1 = mdy_hms(ActivityHour)) %>%
  mutate(date = format(date1,"%d/%m/%y %H:%M:%S")) %>%
  mutate(year = year(date1)) %>%
  mutate(month = month(date1)) %>%
  mutate(day = day(date1)) %>%
  mutate(time = format(date1, "%H:%M:%S")) %>%
  mutate(day_month = paste(month, day, sep = "-")) %>%
  select(id, day_month, Average_Intensity, Total_Intensity, date1) %>%
  mutate(day_month = factor(day_month, levels = unique(day_month[order(date1)])))  # Ordena los niveles


E1 <- E %>%
  mutate(id = as.character(Id)) %>%
  mutate(date1 = mdy_hms(ActivityHour)) %>%
  mutate(date = format(date1,"%d/%m/%y %H:%M:%S")) %>%
  mutate(year = year(date1)) %>%
  mutate(month = month(date1)) %>%
  mutate(day = day(date1)) %>%
  mutate(time = format(date1, "%H:%M:%S")) %>%
  mutate(day_month = paste(month, day, sep = "-")) %>%
  select(id, day_month, StepTotal, date1) %>%
  mutate(day_month = factor(day_month, levels = unique(day_month[order(date1)])))  # Ordena los niveles


F1 <- F %>%
  mutate(id = as.character(Id)) %>%
  mutate(date1 = mdy_hms(ActivityMinute)) %>%
  mutate(date = format(date1,"%d/%m/%y %H:%M:%S")) %>%
  mutate(year = year(date1)) %>%
  mutate(month = month(date1)) %>%
  mutate(day = day(date1)) %>%
  mutate(time = format(date1, "%H:%M:%S")) %>%
  mutate(day_month = paste(month, day, sep = "-")) %>%
  select(id, day_month, Calories, date1) %>%
  mutate(day_month = factor(day_month, levels = unique(day_month[order(date1)])))  # Ordena los niveles


G1 <- G %>%
  mutate(id = as.character(Id)) %>%
```

```
  mutate(date1 = mdy_hms(ActivityMinute)) %>%
  mutate(date = format(date1,"%d/%m/%y %H:%M:%S")) %>%
  mutate(year = year(date1)) %>%
  mutate(month = month(date1)) %>%
  mutate(day = day(date1)) %>%
  mutate(time = format(date1, "%H:%M:%S")) %>%
  mutate(day_month = paste(month, day, sep = "-")) %>%
  select(id, day_month, Intensity, date1) %>%
  mutate(day_month = factor(day_month, levels = unique(day_month[order(date1)])))  # Ordena los niveles


H1 <- H %>%
  mutate(id = as.character(Id)) %>%
  mutate(date1 = mdy_hms(ActivityMinute)) %>%
  mutate(date = format(date1,"%d/%m/%y %H:%M:%S")) %>%
  mutate(year = year(date1)) %>%
  mutate(month = month(date1)) %>%
  mutate(day = day(date1)) %>%
  mutate(time = format(date1, "%H:%M:%S")) %>%
  mutate(day_month = paste(month, day, sep = "-")) %>%
  select(id, day_month, METs, date1) %>%
  mutate(day_month = factor(day_month, levels = unique(day_month[order(date1)])))  # Ordena los niveles


I1 <- I %>%
  rename(value_sleep = value) %>%
  mutate(id = as.character(Id)) %>%
  mutate(log_id = as.character(logId)) %>%
  mutate(date1 = mdy_hms(date)) %>%
  mutate(date = format(date1,"%d/%m/%y %H:%M:%S")) %>%
  mutate(year = year(date1)) %>%
  mutate(month = month(date1)) %>%
  mutate(day = day(date1)) %>%
  mutate(time = format(date1, "%H:%M:%S")) %>%
  mutate(day_month = paste(month, day, sep = "-")) %>%
  select(id, day_month, value_sleep, date1) %>%
  mutate(day_month = factor(day_month, levels = unique(day_month[order(date1)])))  # Ordena los niveles


J1 <- J %>%
  mutate(id = as.character(Id)) %>%
  mutate(date1 = mdy_hms(ActivityMinute)) %>%
  mutate(date = format(date1,"%d/%m/%y %H:%M:%S")) %>%
  mutate(year = year(date1)) %>%
  mutate(month = month(date1)) %>%
  mutate(day = day(date1)) %>%
  mutate(time = format(date1, "%H:%M:%S")) %>%
  mutate(day_month = paste(month, day, sep = "-")) %>%
  select(id, day_month, Steps, date1) %>%
  mutate(day_month = factor(day_month, levels = unique(day_month[order(date1)])))  # Ordena los niveles


K1 <- K %>%
```

```
  rename(Weight_Kg = WeightKg) %>%
  rename(Weight_Pounds = WeightPounds) %>%
  mutate(id = as.character(Id)) %>%
  mutate(log_id = as.character(LogId)) %>%
  mutate(date1 = mdy_hms(Date)) %>%
  mutate(date = format(date1,"%d/%m/%y %H:%M:%S")) %>%
  mutate(year = year(date1)) %>%
  mutate(month = month(date1)) %>%
  mutate(day = day(date1)) %>%
  mutate(time = format(date1, "%H:%M:%S")) %>%
  mutate(day_month = paste(month, day, sep = "-")) %>%
  select(id, day_month, Weight_Kg, Weight_Pounds, Fat, BMI, date1) %>%
  mutate(day_month = factor(day_month, levels = unique(day_month[order(date1)])))  # Ordena los niveles
```

Evaluate the Calories behaviour with date

```
A1_group <- A1 %>%
  group_by(day_month) %>%
  summarise(calories = round(mean(Calories),1))
head(A1_group)
```

```
## # A tibble: 6 x 2
##   day_month calories
##   <fct>        <dbl>
## 1 3-12         2384.
## 2 3-13         2128.
## 3 3-14         2512.
## 4 3-15         2396
## 5 3-16         2882.
## 6 3-17         2741
```

Evaluate the heart rate behaviour with date

```
B1_group <- B1 %>%
  group_by(day_month) %>%
  summarise(heart_rate = round(mean(heart_rate),1))
head(B1_group)
```

```
## # A tibble: 6 x 2
##   day_month      heart_rate
##   <fct>               <dbl>
## 1 3-29-martes          71.3
## 2 3-30-miércoles       73.9
## 3 3-31-jueves          75.9
## 4 4-1-viernes          78.5
## 5 4-2-sábado           77.3
## 6 4-3-domingo          78.6
```

Join the datasets to include values by hour

```
combined_df <- C1 %>%
  full_join(D1, by = c("id", "day_month", "date1")) %>%
  full_join(E1, by = c("id", "day_month", "date1"))

hour_df <- combined_df %>%
  pivot_longer(cols = c(Calories, Total_Intensity, StepTotal),
               names_to = "Metric",
               values_to = "Valores")


hour_df$Metric <- factor(hour_df$Metric, levels = c("Total_Intensity", "Calories", "StepTotal"))

hour_df_group <- hour_df %>%
  group_by(day_month,Metric) %>%
  summarise(Valores = round(mean(Valores),1))
```

```
## `summarise()` has grouped output by 'day_month'. You can override using the
## `.groups` argument.
```

```
head(hour_df_group)
```

```
## # A tibble: 6 x 3
## # Groups:   day_month [2]
##   day_month Metric          Valores
##   <fct>     <fct>             <dbl>
## 1 3-12      Total_Intensity    11.6
## 2 3-12      Calories           97
## 3 3-12      StepTotal         315.
## 4 3-13      Total_Intensity    11.4
## 5 3-13      Calories           94.8
## 6 3-13      StepTotal         302.
```

Join the datasets to include values by hour, this change has with purpose can evaluate the behavior of the value in front of the value of the step

```
hour_df_1 <- combined_df %>%
  pivot_longer(cols = c(Calories, Total_Intensity, StepTotal),
               names_to = "Metric",
               values_to = "Valores") %>%
  arrange(date1)

average_df <- hour_df_1 %>%
  group_by(day_month, Metric) %>%
  summarise(Valores = round(mean(Valores, na.rm = TRUE), 1), .groups = 'drop')

average_steps <- hour_df_1 %>%
  filter(Metric == "StepTotal") %>%
  group_by(day_month) %>%
```

```
  summarise(StepTotal = round(mean(Valores, na.rm = TRUE), 1), .groups = 'drop')

average_df <- average_df %>%
  filter(Metric != "StepTotal") %>%
  left_join(average_steps, by = "day_month")
head((average_df))
```

```
## # A tibble: 6 x 4
##   day_month Metric          Valores StepTotal
##   <fct>     <chr>             <dbl>     <dbl>
## 1 3-12      Calories           97        315.
## 2 3-12      Total_Intensity    11.6      315.
## 3 3-13      Calories           94.8      302.
## 4 3-13      Total_Intensity    11.4      302.
## 5 3-14      Calories           92.7      261.
## 6 3-14      Total_Intensity    10.1      261.
```

Join the datasets to include values by minute

```
combined_df_1 <- F1 %>%
  full_join(G1, by = c("id", "day_month", "date1")) %>%
  full_join(H1, by = c("id", "day_month", "date1")) %>%
  full_join(J1, by = c("id", "day_month", "date1"))

minute_df <- combined_df_1 %>%
  pivot_longer(cols = c(Calories, Intensity, METs, Steps ),
               names_to = "Metric",
               values_to = "Valores")

minute_df$Metric <- factor(minute_df$Metric, levels = c("Intensity", "METs", "Calories","Steps"))

minute_df_group <- minute_df %>%
  group_by(day_month, Metric) %>%
  summarise(Valores = round(mean(Valores),1))
```

```
## `summarise()` has grouped output by 'day_month'. You can override using the
## `.groups` argument.
```

```
head(minute_df_group)
```

```
## # A tibble: 6 x 3
## # Groups:   day_month [2]
##   day_month Metric    Valores
##   <fct>     <fct>       <dbl>
## 1 3-12      Intensity     0.2
## 2 3-12      METs         14.5
## 3 3-12      Calories      1.6
## 4 3-12      Steps         5.3
## 5 3-13      Intensity     0.2
## 6 3-13      METs         14.3
```

Join the datasets to include values by minute, this change has with purpose can evaluate the behavior of the value in front of the value of the step

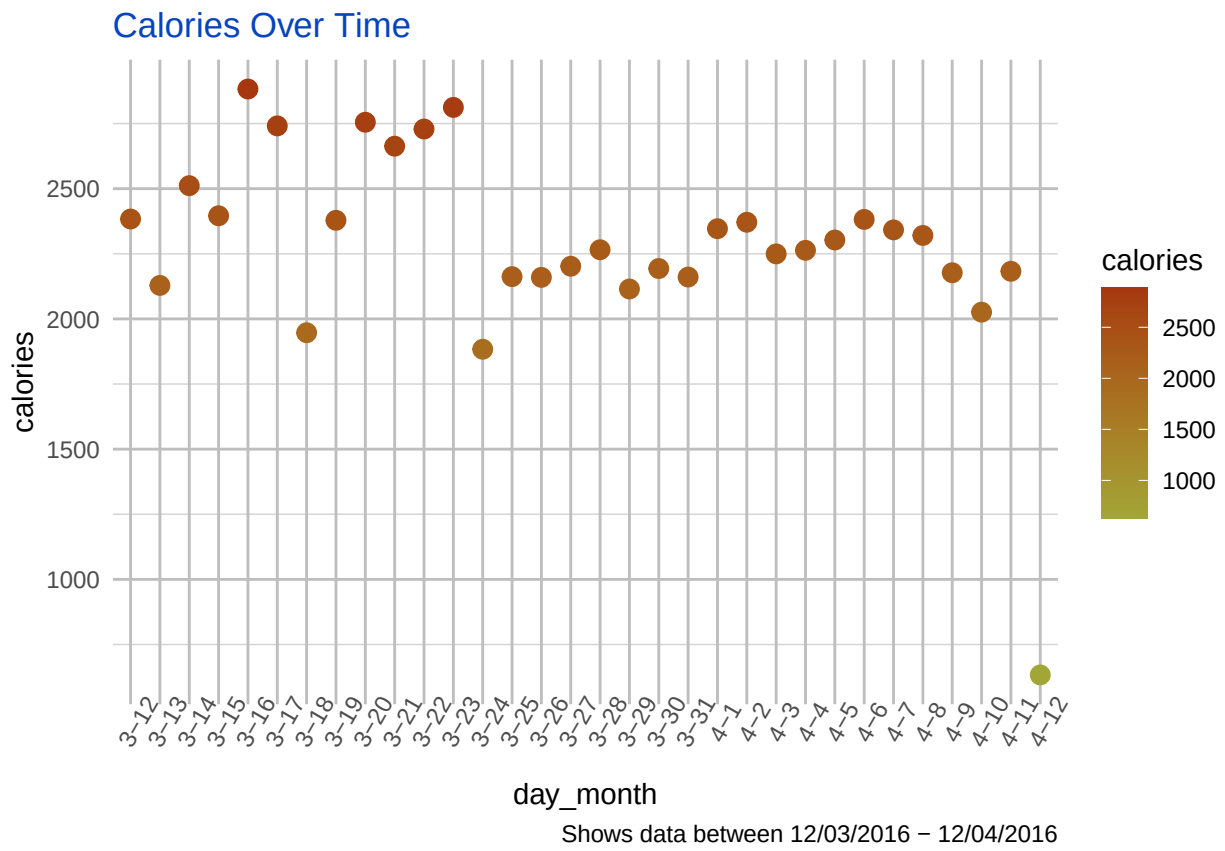
```
averageSteps_df <- minute_df %>%
  group_by(day_month, Metric) %>%
  summarise(Valores = round(mean(Valores, na.rm = TRUE), 1), .groups = 'drop')

average_steps <- minute_df %>%
  filter(Metric == "Steps") %>%
  group_by(day_month) %>%
  summarise(Steps = round(mean(Valores, na.rm = TRUE), 1), .groups = 'drop')

averageSteps_df <- averageSteps_df %>%
  filter(Metric != "Steps") %>%
  left_join(average_steps, by = "day_month")
head(averageSteps_df)
```

```
## # A tibble: 6 x 4
##   day_month Metric    Valores Steps
##   <fct>     <fct>       <dbl> <dbl>
## 1 3-12      Intensity     0.2   5.3
## 2 3-12      METs         14.5   5.3
## 3 3-12      Calories      1.6   5.3
## 4 3-13      Intensity     0.2   5
## 5 3-13      METs         14.3   5
## 6 3-13      Calories      1.6   5
```

Plots

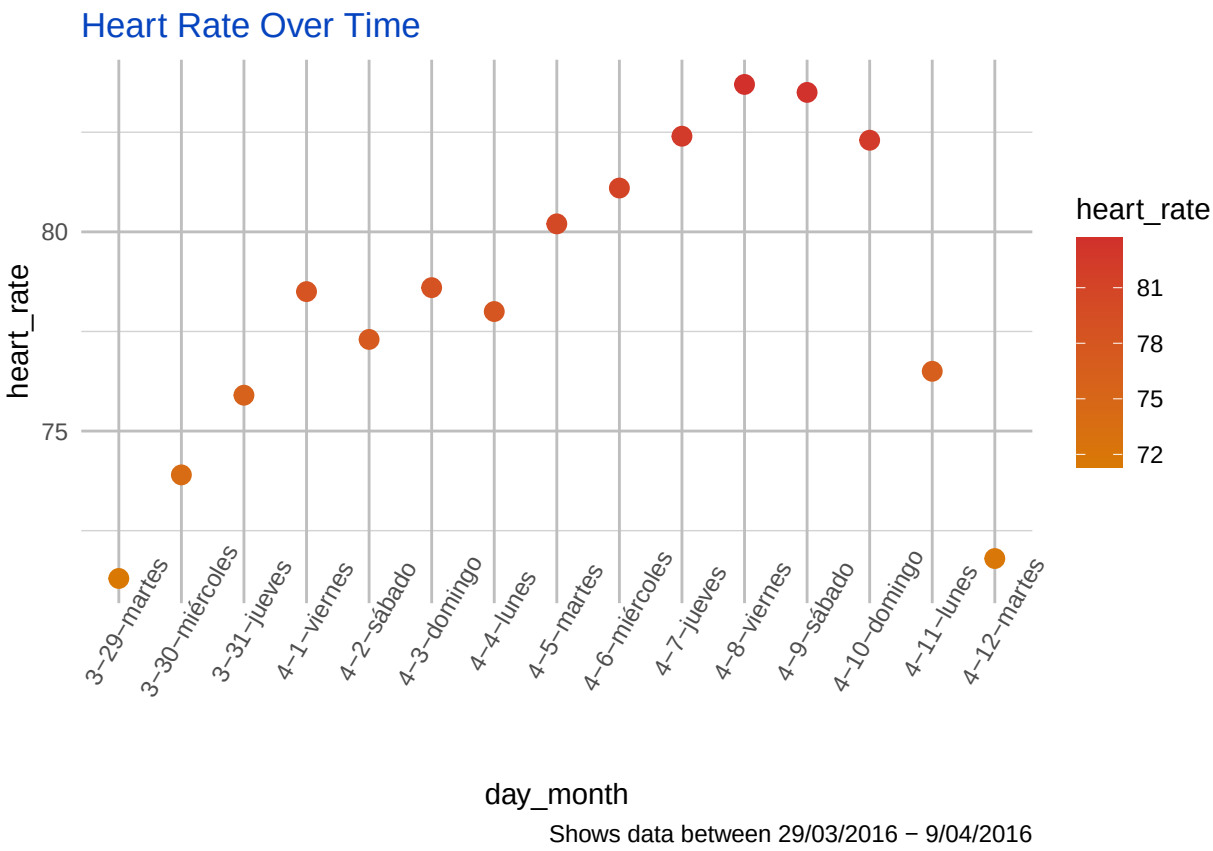
Calories vs Date

```
ggplot(data = A1_group) +
  geom_point(mapping = aes(x = day_month, y = calories, colour = calories), size = 3) +
  scale_color_gradient(low = "#A2A637", high = "#A6370F") +
  theme_minimal() +  # Cambia el fondo a blanco
  theme(
    panel.grid.major = element_line(color = "grey"),  # Líneas principales en gris
    panel.grid.minor = element_line(color = "lightgrey"),  # Líneas secundarias en gris claro
    axis.text.x = element_text(angle = 60),
    plot.title = element_text(color = "#0644BF"))+
  labs(title = "Calories Over Time",
       subtitle = waiver(),
       caption = "Shows data between 12/03/2016 - 12/04/2016",
       tag = waiver(),
       alt = waiver(),
       alt_insight = waiver)
```



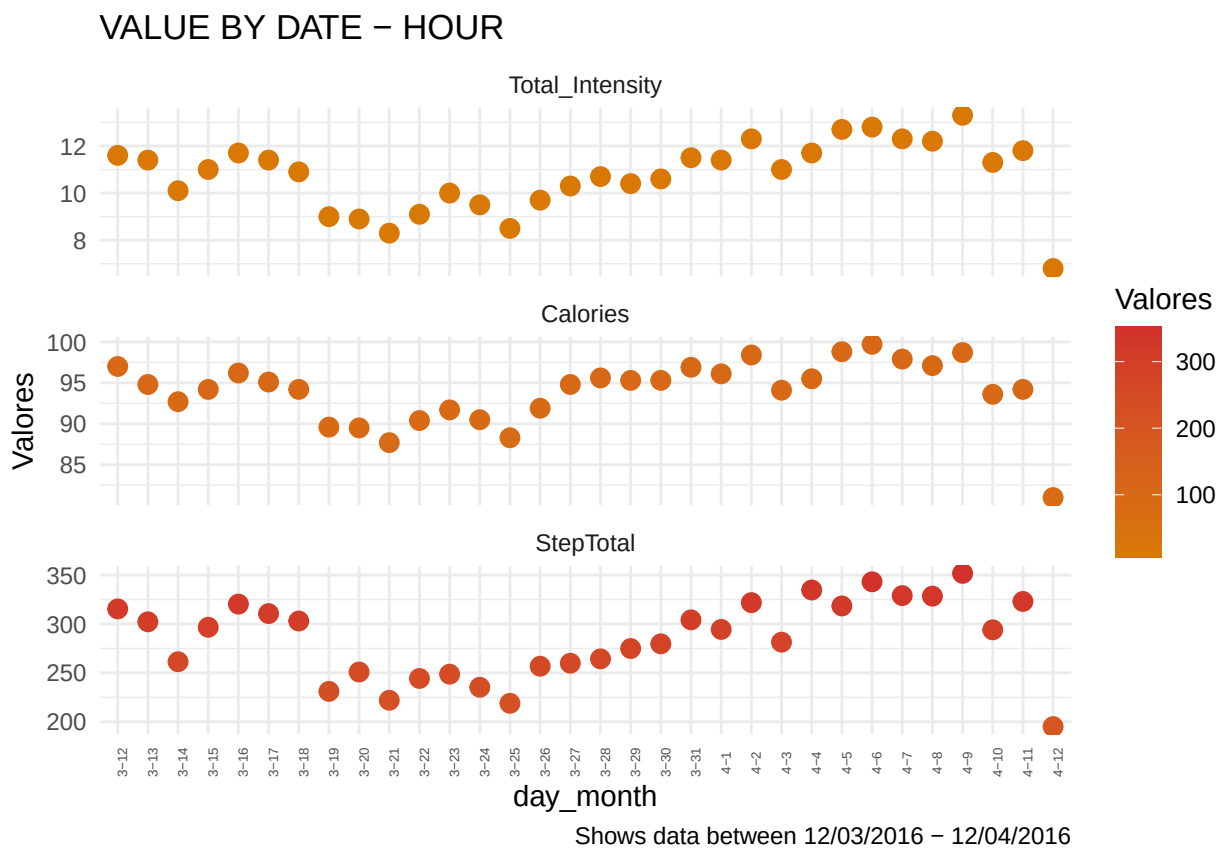
Heart rate vs Date

```
ggplot(data = B1_group) +
  geom_point(mapping = aes(x = day_month, y = heart_rate, color = heart_rate), size = 3) + # Aumenta el
  scale_color_gradient(low = "#D97904", high = "#D1302B") +
  theme_minimal() +  # Cambia el fondo a blanco
  theme(
    panel.grid.major = element_line(color = "grey"),  # Líneas principales en gris
    panel.grid.minor = element_line(color = "lightgrey"),  # Líneas secundarias en gris claro
    axis.text.x = element_text(angle = 60),
    plot.title = element_text(color = "#0644BF"))+
  labs(title = "Heart Rate Over Time",
       subtitle = waiver(),
       caption = "Shows data between 29/03/2016 - 9/04/2016",
       tag = waiver(),
       alt = waiver(),
       alt_insight = waiver)
```



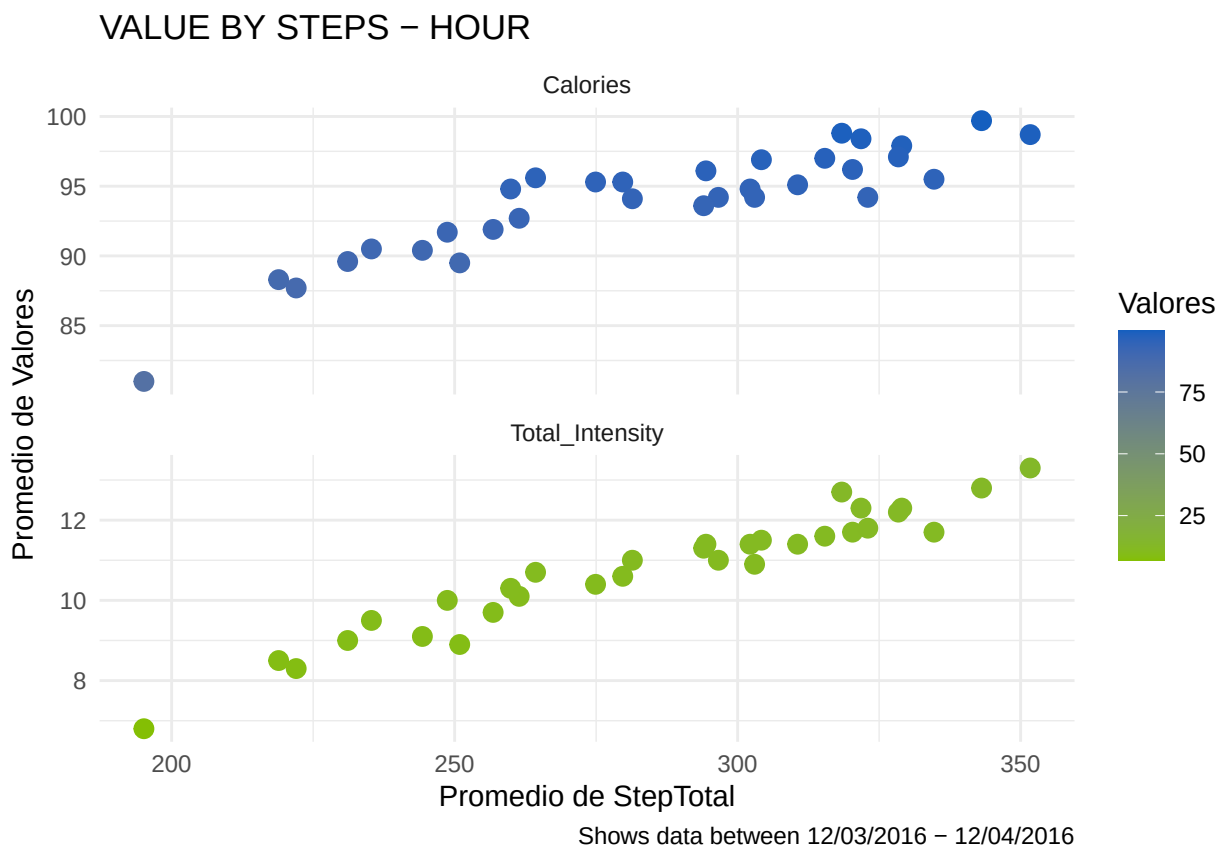
Values averageby hour Vs Date

```
ggplot(data = hour_df_group) +
  geom_point(mapping = aes(x = day_month, y = Valores, colour = Valores), size = 3) + # Aumenta el tama
  facet_wrap(~Metric, ncol = 1, scales = "free_y") + # Disposición 2x2 y ejes y libres
  scale_color_gradient(low = "#D97904", high = "#D1302B") +
  theme_minimal()+
  theme(axis.text.x = element_text(angle = 90, size = 5))+
  labs(title = "VALUE BY DATE - HOUR",
       subtitle = waiver(),
       caption = "Shows data between 12/03/2016 - 12/04/2016",
       tag = waiver(),
       alt = waiver(),
       alt_insight = waiver)
```



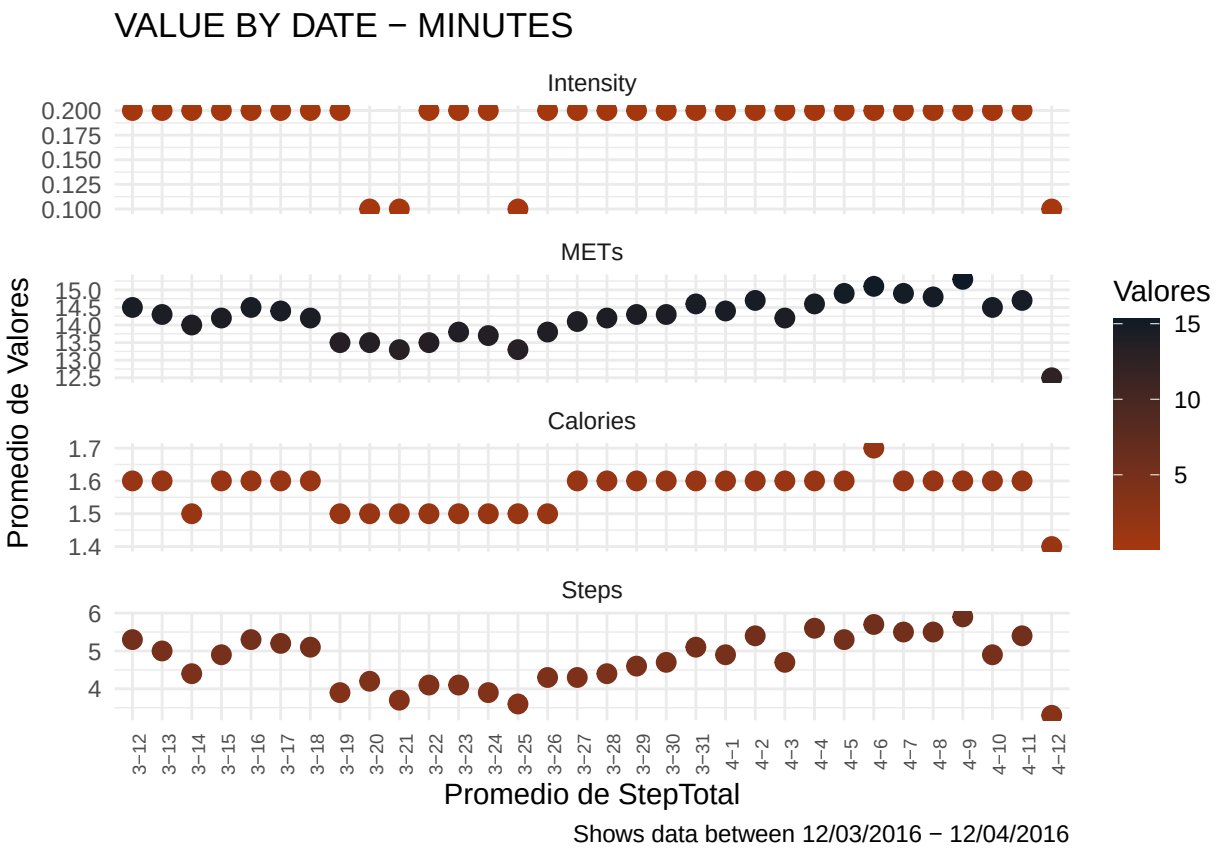
Values average by hour Vs Steps

```
ggplot(data = average_df) +
  geom_point(mapping = aes(x = StepTotal, y = Valores, color = Valores), size = 3) + # Aumenta el tamañ
  facet_wrap(~Metric, ncol = 1, scales = "free_y") + # Disposición 2x2 y ejes y libres
  scale_color_gradient(low = "#84BF04", high = "#155FBF") +
  theme_minimal() +
  labs(x = "Promedio de StepTotal", y = "Promedio de Valores",
       title = "VALUE BY STEPS - HOUR",
       subtitle = waiver(),
       caption = "Shows data between 12/03/2016 - 12/04/2016",
       tag = waiver(),
       alt = waiver(),
       alt_insight = waiver())
```



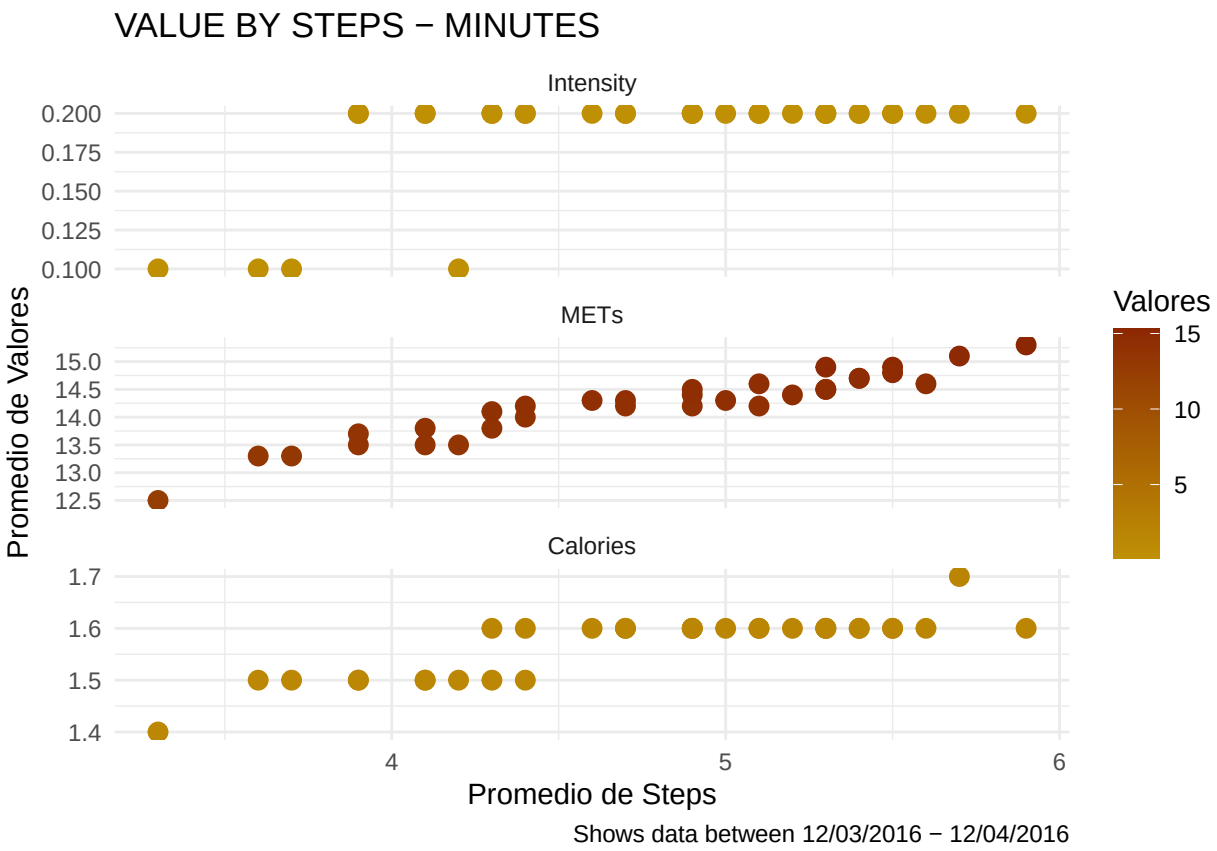
Values average by minute Vs Date

```
ggplot(data = minute_df_group) +
  geom_point(mapping = aes(x = day_month, y = Valores, colour = Valores), size = 3) + # Aumenta el tama
  facet_wrap(~Metric, ncol = 1, scales = "free_y") + # Disposición 2x2 y ejes y libres
  scale_color_gradient(low = "#A6370F", high = "#0E1B26") +
  theme_minimal()+
  theme(axis.text.x = element_text(angle = 90, size = 7)) +
  labs(x = "Promedio de StepTotal", y = "Promedio de Valores",
       title = "VALUE BY DATE - MINUTES",
       subtitle = waiver(),
       caption = "Shows data between 12/03/2016 - 12/04/2016",
       tag = waiver(),
       alt = waiver(),
       alt_insight = waiver())
```



Values average by minute Vs Steps

```
ggplot(data = averageSteps_df) +
  geom_point(mapping = aes(x = Steps, y = Valores, color = Valores), size = 3) + # Aumenta el tamaño de
  facet_wrap(~Metric, ncol = 1, scales = "free_y") + # Disposición 2x2 y ejes y libres
  scale_color_gradient(low = "#BF9004", high = "#8C2703") +
  theme_minimal() +
  labs(x = "Promedio de Steps", y = "Promedio de Valores",
      title = "VALUE BY STEPS - MINUTES",
      subtitle = waiver(),
      caption = "Shows data between 12/03/2016 - 12/04/2016",
      tag = waiver(),
      alt = waiver(),
      alt_insight = waiver())
```



Recommendations

- Run marketing campaigns at the beginning of the month about buying health device
- Promote more walking for the end of the month
- Display the health benefits when you walk more