Daniel Terra Gomes

Projeto Final - Fitbit Fitness Data

Campos dos Goytacazes, RJ

5 de fevereiro de 2023, v1.0.0

Daniel Terra Gomes

Projeto Final - Fitbit Fitness Data

Relatório Projeto Final apresentado ao Curso de Ciência da Computação da Universidade Estadual do Norte Fluminense Darcy Ribeiro, como requisito avaliativo da disciplina.

Campos dos Goytacazes, RJ
5 de fevereiro de 2023, v1.0.0

Sumário

0.1 Definição do problema.

Quais são algumas tendências no uso de dispositivos inteligentes?

Os dados que iremos usar para responder essa pergunta é o FitBit Fitness Tracker Data(License CC0: Public Domain, disponível através do Mobius distribuído através do Amazon Mechanical Turk entre 12 de março de 2016 e 12 de maio de 2016.)

O conjunto de dados contém rastreamento de atividade física pessoal para 33 usuários do Fitbit. Esses usuários qualificados do Fitbit concordaram com o envio de dados de rastreamento pessoal, incluindo minutos de desempenho de condicionamento físico, frequência cardíaca e monitoramento do sono.

O aplicativo fornece aos usuários dados de saúde relacionados à atividade, sono, estresse, ciclo menstrual e hábitos de foco. Esses dados podem ajudar os usuários a entender melhor seus hábitos atuais e tomar decisões saudáveis. O aplicativo se conecta à sua linha de produtos de bem-estar inteligentes da empresa.

Temos como **objetivo** nesse **projeto** analisar os dados de uso de dispositivos inteligentes para obter informações sobre como as pessoas já estão usando seus dispositivos e usar essas tendências para entendermos e identificar possiveis correlações e assim propor possiveis hypothesis para o frame de dados analisado.

0.1.1 Bibliotecas que iremos usar no nosso Projeto

[1]:
```
        library(janitor) # janitor tem pequenas ferramentas simples para examinar e␣
↪limpar dados sujos.
        library(arrow) # permite que os usuários leiam e gravem dados em vários␣
↪formatos: Parquet, csv, JSON
        library(tidyverse) # você disse ciência de dados usando R?
        library(naniar) # lidando com valores NA
        library(ggsci) # Revista Científica e Tema Sci-Fi Paletas de cores para␣
↪ggplot2
        library(skimr) # Skim estatísticas de resumo úteis
        library(lubridate) # Lubridate fornece ferramentas que facilitam a análise␣
↪e manipulação de datas. ymd_hms() week()
        library(ggpubr) # O pacote 'ggpubr' fornece algumas funções fáceis de usar␣
↪para criar e personalizar gráficos prontos para publicação baseados em 'ggplot2'.
↪stat_cor
```

```
Attaching package: ‘janitor’


The following objects are masked from ‘package:stats’:

chisq.test, fisher.test



Attaching package: ‘arrow’
```

```
The following object is masked from 'package:utils':

timestamp


 Attaching packages  tidyverse
1.3.2
 ggplot2 3.4.0       purrr   1.0.1
 tibble  3.1.8       dplyr   1.0.10
 tidyr   1.2.1       stringr 1.5.0
 readr   2.1.3       forcats 0.5.2
 Conflicts
tidyverse_conflicts()
 dplyr::filter() masks stats::filter()
 dplyr::lag()    masks stats::lag()

Attaching package: 'skimr'


The following object is masked from 'package:naniar':

n_complete


Loading required package: timechange


Attaching package: 'lubridate'


The following object is masked from 'package:arrow':

duration


The following objects are masked from 'package:base':

date, intersect, setdiff, union
```

0.2 Apresentação dos dados

Como o objetivo é identificar tendências no uso de dispositivos inteligentes, decidimos trabalhar com os cinco dataframes a seguir:

- sleepDay_merged.csv
- dailyActivity_merged.csv
- dailyIntensities_merged.csv
- hourlyIntensities_merged.csv
- hourlyCalories_merged.csv

0.2.1 Lendo os arquivos .csv

[2]:
```
sleep_day_file <- read_csv("/kaggle/input/fitbit/Fitabase Data 4.12.16-5.12.16/sleepDay_merged.csv")
daily_activity_file <- read_csv("/kaggle/input/fitbit/Fitabase Data 4.12.16-5.12.16/dailyActivity_merged.csv")
daily_intensities_file <- read_csv("/kaggle/input/fitbit/Fitabase Data 4.12.16-5.12.16/dailyIntensities_merged.csv")
hourly_intensities_file <- read_csv("/kaggle/input/fitbit/Fitabase Data 4.12.16-5.12.16/hourlyIntensities_merged.csv")
hourly_calories_file <- read_csv("/kaggle/input/fitbit/Fitabase Data 4.12.16-5.12.16/hourlyCalories_merged.csv")
```

```
Rows: 413 Columns: 5
 Column specification

Delimiter: ","
chr (1): SleepDay
dbl (4): Id, TotalSleepRecords, TotalMinutesAsleep, TotalTimeInBed

 Use `spec()` to retrieve the full column specification for this
data.
 Specify the column types or set `show_col_types = FALSE` to quiet
this message.
Rows: 940 Columns: 15
 Column specification

Delimiter: ","
chr  (1): ActivityDate
dbl (14): Id, TotalSteps, TotalDistance, TrackerDistance,
LoggedActivitiesDi...

 Use `spec()` to retrieve the full column specification for this
data.
 Specify the column types or set `show_col_types = FALSE` to quiet
this message.
Rows: 940 Columns: 10
 Column specification

Delimiter: ","
chr (1): ActivityDay
```

```
dbl (9): Id, SedentaryMinutes, LightlyActiveMinutes,
FairlyActiveMinutes, Ve...

 Use `spec()` to retrieve the full column specification for this
data.
 Specify the column types or set `show_col_types = FALSE` to quiet
this message.
Rows: 22099 Columns: 4
 Column specification

Delimiter: ","
chr (1): ActivityHour
dbl (3): Id, TotalIntensity, AverageIntensity

 Use `spec()` to retrieve the full column specification for this
data.
 Specify the column types or set `show_col_types = FALSE` to quiet
this message.
Rows: 22099 Columns: 3
 Column specification

Delimiter: ","
chr (1): ActivityHour
dbl (2): Id, Calories

 Use `spec()` to retrieve the full column specification for this
data.
 Specify the column types or set `show_col_types = FALSE` to quiet
this message.
```

0.2.2 Convertendo os arquivos .csv em .parquet

A partir dessa conversão iremos ganhar desempenho em nossas análises

O Apache Parquet é um formato de arquivo projetado para oferecer suporte ao processamento rápido de dados complexos, com várias características notáveis: * Compressão * Evolução do esquema * Código aberto e não proprietário Mesmo que o uso do Parquet não seja necessário para o nosso conjunto de dados, pois não é um banco muito grande. Ainda sim nos trara um pouco mais de rapidez.

Dataset	Size on Amazon S3	Query Run time	Data Scanned	Cost
Data stored as CSV files	1 TB	236 seconds	1.15 TB	$5.75
Data stored in Apache Parquet format*	130 GB	6.78 seconds	2.51 GB	$0.01
Savings / Speedup	87% less with Parquet	34x faster	99% less data scanned	99.7% savings

ref.

[3]:
```
write_parquet(sleep_day_file, "/kaggle/working/sleepDay.parquet")
write_parquet(daily_activity_file, "/kaggle/working/dailyActivity.parquet")
write_parquet(daily_intensities_file, "/kaggle/working/dailyIntensities.
↪parquet")
write_parquet(hourly_intensities_file, "/kaggle/working/hourly_intensities.
↪parquet")
write_parquet(hourly_calories_file, "/kaggle/working/hourly_calories.
↪parquet")
```

0.2.3 Lendo os arquivos .parquet

[4]:
```
sleep_day_file <- read_parquet("/kaggle/working/sleepDay.parquet")
daily_activity_file <- read_parquet("/kaggle/working/dailyActivity.parquet")
daily_intensities_file <- read_parquet("/kaggle/working/dailyIntensities.
↪parquet")
hourly_intensities_file <- read_parquet("/kaggle/working/hourly_intensities.
↪parquet")
hourly_calories_file <- read_parquet("/kaggle/working/hourly_calories.
↪parquet")
```

0.2.4 Entendendo um pouco dos dados

Aqui teremos o nome primeiro contato com o formato, conjunto dos dados que iremos trabalhar no nosso projeto

[5]:
```
#skim_without_charts(sleep_day_file)
#skim_without_charts(daily_activity_file)
#skim_without_charts(daily_intensities_file)
#skim_without_charts(hourly_intensities_file)
#skim_without_charts(hourly_calories_file)
```

skim output foi comentado pois não estava permitindo o push no kaggle. Provavelmente, devido ao tamanho da saída.

[6]:
```
glimpse(sleep_day_file)
glimpse(daily_activity_file)
glimpse(daily_intensities_file)
glimpse(hourly_intensities_file)
glimpse(hourly_calories_file)
```

```
Rows: 413
Columns: 5
$ Id                <dbl> 1503960366, 1503960366,
1503960366, 1503960366, 150...
$ SleepDay          <chr> "4/12/2016 12:00:00 AM",
"4/13/2016 12:00:00 AM", "...
$ TotalSleepRecords <dbl> 1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, ...
```

```
$ TotalMinutesAsleep <dbl> 327, 384, 412, 340, 700, 304, 360,
325, 361, 430, 2...
$ TotalTimeInBed     <dbl> 346, 407, 442, 367, 712, 320, 377,
364, 384, 449, 3...
Rows: 940
Columns: 15
$ Id                       <dbl> 1503960366, 1503960366,
1503960366, 150396036...
$ ActivityDate             <chr> "4/12/2016", "4/13/2016",
"4/14/2016", "4/15/...
$ TotalSteps               <dbl> 13162, 10735, 10460, 9762,
12669, 9705, 13019...
$ TotalDistance            <dbl> 8.50, 6.97, 6.74, 6.28,
8.16, 6.48, 8.59, 9.8...
$ TrackerDistance          <dbl> 8.50, 6.97, 6.74, 6.28,
8.16, 6.48, 8.59, 9.8...
$ LoggedActivitiesDistance <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, ...
$ VeryActiveDistance       <dbl> 1.88, 1.57, 2.44, 2.14,
2.71, 3.19, 3.25, 3.5...
$ ModeratelyActiveDistance <dbl> 0.55, 0.69, 0.40, 1.26,
0.41, 0.78, 0.64, 1.3...
$ LightActiveDistance      <dbl> 6.06, 4.71, 3.91, 2.83,
5.04, 2.51, 4.71, 5.0...
$ SedentaryActiveDistance  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, ...
$ VeryActiveMinutes        <dbl> 25, 21, 30, 29, 36, 38, 42,
50, 28, 19, 66, 4...
$ FairlyActiveMinutes      <dbl> 13, 19, 11, 34, 10, 20, 16,
31, 12, 8, 27, 21...
$ LightlyActiveMinutes     <dbl> 328, 217, 181, 209, 221,
164, 233, 264, 205, ...
$ SedentaryMinutes         <dbl> 728, 776, 1218, 726, 773,
539, 1149, 775, 818...
$ Calories                 <dbl> 1985, 1797, 1776, 1745,
1863, 1728, 1921, 203...
Rows: 940
Columns: 10
$ Id                       <dbl> 1503960366, 1503960366,
1503960366, 150396036...
$ ActivityDay              <chr> "4/12/2016", "4/13/2016",
"4/14/2016", "4/15/...
$ SedentaryMinutes         <dbl> 728, 776, 1218, 726, 773,
539, 1149, 775, 818...
$ LightlyActiveMinutes     <dbl> 328, 217, 181, 209, 221,
164, 233, 264, 205, ...
$ FairlyActiveMinutes      <dbl> 13, 19, 11, 34, 10, 20, 16,
```

```
31, 12, 8, 27, 21...
$ VeryActiveMinutes        <dbl> 25, 21, 30, 29, 36, 38, 42,
50, 28, 19, 66, 4...
$ SedentaryActiveDistance  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, ...
$ LightActiveDistance      <dbl> 6.06, 4.71, 3.91, 2.83,
5.04, 2.51, 4.71, 5.0...
$ ModeratelyActiveDistance <dbl> 0.55, 0.69, 0.40, 1.26,
0.41, 0.78, 0.64, 1.3...
$ VeryActiveDistance       <dbl> 1.88, 1.57, 2.44, 2.14,
2.71, 3.19, 3.25, 3.5...
Rows: 22,099
Columns: 4
$ Id               <dbl> 1503960366, 1503960366, 1503960366,
1503960366, 15039...
$ ActivityHour     <chr> "4/12/2016 12:00:00 AM", "4/12/2016
1:00:00 AM", "4/1...
$ TotalIntensity   <dbl> 20, 8, 7, 0, 0, 0, 0, 0, 13, 30, 29,
12, 11, 6, 36, 5...
$ AverageIntensity <dbl> 0.333333, 0.133333, 0.116667,
0.000000, 0.000000, 0.0...
Rows: 22,099
Columns: 3
$ Id           <dbl> 1503960366, 1503960366, 1503960366,
1503960366, 150396036...
$ ActivityHour <chr> "4/12/2016 12:00:00 AM", "4/12/2016
1:00:00 AM", "4/12/20...
$ Calories     <dbl> 81, 61, 59, 47, 48, 48, 48, 47, 68, 141,
99, 76, 73, 66, ...
```

A partir desses resultados podemos identificar que não temos dados NA, dados faltantes em nosso conjuntos de dados. Assim como podemos ter uma média desses valores, e ver como as colunas estão organizadas.

0.2.4.1 Número de usuários únicos

[7]:
```
        count(distinct(sleep_day_file, Id))
        count(distinct(daily_activity_file, Id))
        count(distinct(daily_intensities_file, Id))
        count(distinct(hourly_intensities_file, Id))
        count(distinct(hourly_calories_file, Id))
```

A tibble: 1 × 1

n
<int>
24

A tibble: 1 × 1

n
<int>
33

A tibble: 1 × 1

n
<int>
33

A tibble: 1 × 1

n
<int>
33

A tibble: 1 × 1

n
<int>
33

0.3 Preparação dos dados

0.3.1 Limpeza dos Dados

[8]:
```
anyDuplicated(sleep_day_file)
anyDuplicated(daily_activity_file)
anyDuplicated(daily_intensities_file)
anyDuplicated(hourly_intensities_file)
anyDuplicated(hourly_calories_file)
```

162

0

0

0

0

aqui podemos identificar que temos 162 duplicatas no sleep day file

0.3.1.1 Descartando NA e duplicatas

[9]:
```
sleep_day_file <- sleep_day_file %>%
  distinct() %>%
  drop_na()

anyDuplicated(sleep_day_file)
```

0

Tratamentos dos dados feitos, e agora temos 0 dados duplicados. Esse tratamento é feito para evitar dados iguais esses dados duplicados podem trazer vieses para as nossas analises futuras.

0.3.1.2 Limpando os nomes para o formato usado nas aulas

[10]:
```
sleep_day_file <- clean_names(sleep_day_file)
daily_activity_file <- clean_names(daily_activity_file)
daily_intensities_file <- clean_names(daily_intensities_file)
hourly_intensities_file <- clean_names(hourly_intensities_file)
hourly_calories_file <- clean_names(hourly_calories_file)
```

agora temos os nossos nomes de colunas em um formato padronizado; name1_name2 (low-case,separado por _)

0.3.1.3 Transformando as Datas

[11]:
```
glimpse(sleep_day_file)
glimpse(daily_activity_file)
glimpse(daily_intensities_file)
glimpse(hourly_intensities_file)
glimpse(hourly_calories_file)
```

```
Rows: 410
Columns: 5
$ id                   <dbl> 1503960366, 1503960366,
1503960366, 1503960366, 1...
$ sleep_day            <chr> "4/12/2016 12:00:00 AM",
"4/13/2016 12:00:00 AM",...
$ total_sleep_records  <dbl> 1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1...
$ total_minutes_asleep <dbl> 327, 384, 412, 340, 700, 304,
360, 325, 361, 430,...
$ total_time_in_bed    <dbl> 346, 407, 442, 367, 712, 320,
377, 364, 384, 449,...
Rows: 940
Columns: 15
$ id                          <dbl> 1503960366, 1503960366,
1503960366, 1503960...
$ activity_date               <chr> "4/12/2016", "4/13/2016",
"4/14/2016", "4/1...
$ total_steps                 <dbl> 13162, 10735, 10460, 9762,
12669, 9705, 130...
$ total_distance              <dbl> 8.50, 6.97, 6.74, 6.28,
8.16, 6.48, 8.59, 9...
$ tracker_distance            <dbl> 8.50, 6.97, 6.74, 6.28,
8.16, 6.48, 8.59, 9...
$ logged_activities_distance <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0...
$ very_active_distance        <dbl> 1.88, 1.57, 2.44, 2.14,
2.71, 3.19, 3.25, 3...
$ moderately_active_distance <dbl> 0.55, 0.69, 0.40, 1.26,
```

```
0.41, 0.78, 0.64, 1...
$ light_active_distance      <dbl> 6.06, 4.71, 3.91, 2.83,
5.04, 2.51, 4.71, 5...
$ sedentary_active_distance  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0...
$ very_active_minutes        <dbl> 25, 21, 30, 29, 36, 38,
42, 50, 28, 19, 66,...
$ fairly_active_minutes      <dbl> 13, 19, 11, 34, 10, 20,
16, 31, 12, 8, 27, ...
$ lightly_active_minutes     <dbl> 328, 217, 181, 209, 221,
164, 233, 264, 205...
$ sedentary_minutes          <dbl> 728, 776, 1218, 726, 773,
539, 1149, 775, 8...
$ calories                   <dbl> 1985, 1797, 1776, 1745,
1863, 1728, 1921, 2...
Rows: 940
Columns: 10
$ id                         <dbl> 1503960366, 1503960366,
1503960366, 1503960...
$ activity_day               <chr> "4/12/2016", "4/13/2016",
"4/14/2016", "4/1...
$ sedentary_minutes          <dbl> 728, 776, 1218, 726, 773,
539, 1149, 775, 8...
$ lightly_active_minutes     <dbl> 328, 217, 181, 209, 221,
164, 233, 264, 205...
$ fairly_active_minutes      <dbl> 13, 19, 11, 34, 10, 20,
16, 31, 12, 8, 27, ...
$ very_active_minutes        <dbl> 25, 21, 30, 29, 36, 38,
42, 50, 28, 19, 66,...
$ sedentary_active_distance  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0...
$ light_active_distance      <dbl> 6.06, 4.71, 3.91, 2.83,
5.04, 2.51, 4.71, 5...
$ moderately_active_distance <dbl> 0.55, 0.69, 0.40, 1.26,
0.41, 0.78, 0.64, 1...
$ very_active_distance       <dbl> 1.88, 1.57, 2.44, 2.14,
2.71, 3.19, 3.25, 3...
Rows: 22,099
Columns: 4
$ id                <dbl> 1503960366, 1503960366, 1503960366,
1503960366, 1503...
$ activity_hour     <chr> "4/12/2016 12:00:00 AM", "4/12/2016
1:00:00 AM", "4/...
$ total_intensity   <dbl> 20, 8, 7, 0, 0, 0, 0, 0, 13, 30,
29, 12, 11, 6, 36, ...
$ average_intensity <dbl> 0.333333, 0.133333, 0.116667,
0.000000, 0.000000, 0....
```

```
Rows: 22,099
Columns: 3
$ id           <dbl> 1503960366, 1503960366, 1503960366,
1503960366, 15039603...
$ activity_hour <chr> "4/12/2016 12:00:00 AM", "4/12/2016
1:00:00 AM", "4/12/2...
$ calories     <dbl> 81, 61, 59, 47, 48, 48, 48, 47, 68,
141, 99, 76, 73, 66,...
```

iremos fazer a transformação das colunas que tratam de **horas e datas** para facilicar no momento das nossas analises.

[12]:
```
        sleep_day_file$sleep_day <- mdy_hms(sleep_day_file$sleep_day)
        daily_activity_file$activity_date <- mdy(daily_activity_file$activity_date)
        daily_intensities_file$activity_day <-␣
↪mdy(daily_intensities_file$activity_day)
        hourly_intensities_file$activity_hour <-␣
↪mdy_hms(hourly_intensities_file$activity_hour)
        hourly_calories_file$activity_hour <-␣
↪mdy_hms(hourly_calories_file$activity_hour)
```

A partir do **glimpse()** podemos identificar que as colunas referentes a hora e datas agora estão nos seguintes formatos de dados: **dttm,date**. Esse processamento é possível a partir da biblioteca ***lubridate*** que nos da as funções e argumentos para tratar esse tipo de situações e muito mais.

0.3.1.3.1 Transformações extras

tratando dados para fazer o grafico de total de Total de Passos diários Vs. Total de Minutos de Sono

[13]:
```
        glimpse(sleep_day_file)
        glimpse(daily_activity_file)
```

```
Rows: 410
Columns: 5
$ id                   <dbl> 1503960366, 1503960366,
1503960366, 1503960366, 1...
$ sleep_day            <dttm> 2016-04-12, 2016-04-13,
2016-04-15, 2016-04-16, ...
$ total_sleep_records  <dbl> 1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1...
$ total_minutes_asleep <dbl> 327, 384, 412, 340, 700, 304,
360, 325, 361, 430,...
$ total_time_in_bed    <dbl> 346, 407, 442, 367, 712, 320,
377, 364, 384, 449,...
Rows: 940
Columns: 15
$ id                         <dbl> 1503960366, 1503960366,
```

```
1503960366, 1503960...
$ activity_date              <date> 2016-04-12, 2016-04-13,
2016-04-14, 2016-0...
$ total_steps                <dbl> 13162, 10735, 10460, 9762,
12669, 9705, 130...
$ total_distance             <dbl> 8.50, 6.97, 6.74, 6.28,
8.16, 6.48, 8.59, 9...
$ tracker_distance           <dbl> 8.50, 6.97, 6.74, 6.28,
8.16, 6.48, 8.59, 9...
$ logged_activities_distance <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0...
$ very_active_distance       <dbl> 1.88, 1.57, 2.44, 2.14,
2.71, 3.19, 3.25, 3...
$ moderately_active_distance <dbl> 0.55, 0.69, 0.40, 1.26,
0.41, 0.78, 0.64, 1...
$ light_active_distance      <dbl> 6.06, 4.71, 3.91, 2.83,
5.04, 2.51, 4.71, 5...
$ sedentary_active_distance  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0...
$ very_active_minutes        <dbl> 25, 21, 30, 29, 36, 38,
42, 50, 28, 19, 66,...
$ fairly_active_minutes      <dbl> 13, 19, 11, 34, 10, 20,
16, 31, 12, 8, 27, ...
$ lightly_active_minutes     <dbl> 328, 217, 181, 209, 221,
164, 233, 264, 205...
$ sedentary_minutes          <dbl> 728, 776, 1218, 726, 773,
539, 1149, 775, 8...
$ calories                   <dbl> 1985, 1797, 1776, 1745,
1863, 1728, 1921, 2...
```

Para dar merge() os dados diários de sono e atividade com base em ID e data, precisaremos padronizar os nomes das colunas de ID e Data.

Renomeando a coluna que exibe as datas para que possa ser usada como uma das chaves durante a merge() de dados:

[14]:
```
daily_activity_file <- daily_activity_file %>% rename(date = activity_date)
sleep_day_file <- sleep_day_file %>% rename(date = sleep_day)
```

Merge() os subconjuntos 'sleep' e 'activity' para criar o conjunto de dados 'activity_sleep', usando "id" e "date" como chaves:

[15]:
```
activity_sleep <- merge(daily_activity_file, sleep_day_file, by=c("id",␣
↪"date"))



glimpse(activity_sleep)
```

```
Rows: 410
Columns: 18
$ id                          <dbl> 1503960366, 1503960366,
1503960366, 1503960...
$ date                        <date> 2016-04-12, 2016-04-13,
2016-04-15, 2016-0...
$ total_steps                 <dbl> 13162, 10735, 9762, 12669,
9705, 15506, 105...
$ total_distance              <dbl> 8.50, 6.97, 6.28, 8.16,
6.48, 9.88, 6.68, 6...
$ tracker_distance            <dbl> 8.50, 6.97, 6.28, 8.16,
6.48, 9.88, 6.68, 6...
$ logged_activities_distance <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0...
$ very_active_distance        <dbl> 1.88, 1.57, 2.14, 2.71,
3.19, 3.53, 1.96, 1...
$ moderately_active_distance <dbl> 0.55, 0.69, 1.26, 0.41,
0.78, 1.32, 0.48, 0...
$ light_active_distance       <dbl> 6.06, 4.71, 2.83, 5.04,
2.51, 5.03, 4.24, 4...
$ sedentary_active_distance  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0...
$ very_active_minutes         <dbl> 25, 21, 29, 36, 38, 50,
28, 19, 41, 39, 73,...
$ fairly_active_minutes       <dbl> 13, 19, 34, 10, 20, 31,
12, 8, 21, 5, 14, 2...
$ lightly_active_minutes      <dbl> 328, 217, 209, 221, 164,
264, 205, 211, 262...
$ sedentary_minutes           <dbl> 728, 776, 726, 773, 539,
775, 818, 838, 732...
$ calories                    <dbl> 1985, 1797, 1745, 1863,
1728, 2035, 1786, 1...
$ total_sleep_records         <dbl> 1, 2, 1, 2, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1...
$ total_minutes_asleep        <dbl> 327, 384, 412, 340, 700,
304, 360, 325, 361...
$ total_time_in_bed           <dbl> 346, 407, 442, 367, 712,
320, 377, 364, 384...
```

Abaixo trataremos os dados para o Grafico - Calorias x intensidade. Vamos colocar as datas em dia para conseguiremos agrupar por esse conjunto de dados, e vamos dar um **cbind()** nos valores de calorias essa função pega uma sequência de argumentos de vetor, matriz ou quadro de dados e combine por colunas ou linhas, respectivamente.

[16]:
```
      hourly_intensities_file$day <-␣
↪format(hourly_intensities_file$activity_hour, format = "%Y %m %d")
      hourly_intensities_file$calories <- cbind(hourly_calories_file$calories)
```

```
glimpse(hourly_intensities_file)
```

```
Rows: 22,099
Columns: 6
$ id                <dbl> 1503960366, 1503960366, 1503960366,
1503960366, 1503...
$ activity_hour     <dttm> 2016-04-12 00:00:00, 2016-04-12
01:00:00, 2016-04-1...
$ total_intensity   <dbl> 20, 8, 7, 0, 0, 0, 0, 0, 13, 30,
29, 12, 11, 6, 36, ...
$ average_intensity <dbl> 0.333333, 0.133333, 0.116667,
0.000000, 0.000000, 0....
$ day               <chr> "2016 04 12", "2016 04 12", "2016
04 12", "2016 04 1...
$ calories          <dbl[,1]> <matrix[26 x 1]>
```

Abaixo adicionaremos dias da semana aos conjuntos de dados para auxiliar nas análises refentes ao gráfico Tempo de sono

[17]:
```
sleep_day_file <- sleep_day_file %>% mutate(day_of_week = weekdays(date))
# sleep_day_file$weekday <- weekdays(sleep_day_file$sleep_day)
sleep_day_file$day_of_week <- factor(sleep_day_file$day_of_week,levels =␣
↪c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"))
```

Aqui fizemos um mutate para adicionar a coluna referente a dias da semana. Passamos *date* para a *função* weekdays para assim termos os dias da semana refentes as data do nosso dataframe. Em seguida, fazemos um *factor* para que os dias da semana sejam organizadas da ordem passada no vetor *c* em *levels*.

[18]:
```
unique(sleep_day_file$day_of_week)
```

Tuesday

Wednesday

Friday

Saturday

Sunday

Thursday

Monday

Levels:

'Monday'

'Tuesday'

'Wednesday'

'Thursday'

'Friday'

'Saturday'

'Sunday'

testando para ver se temos os dias referentes a todos os dias da semana

Abaixo vamos fazer o somatório para o gráfico Uso diário dos dispositivos que nós apresenta a quantidade de horas que os nossos usuários passam usando os aparelhos inteligentes.

[19]:
```
      daily_activity_file$total_time =␣
↪rowSums(daily_activity_file[c("very_active_minutes",
                                                    ␣
↪"fairly_active_minutes",
                                                    ␣
↪"lightly_active_minutes",
                                                    ␣
↪"sedentary_minutes")])
      glimpse(daily_activity_file)
```

```
Rows: 940
Columns: 16
$ id                       <dbl> 1503960366, 1503960366,
1503960366, 1503960...
$ date                     <date> 2016-04-12, 2016-04-13,
2016-04-14, 2016-0...
$ total_steps              <dbl> 13162, 10735, 10460, 9762,
12669, 9705, 130...
$ total_distance           <dbl> 8.50, 6.97, 6.74, 6.28,
8.16, 6.48, 8.59, 9...
$ tracker_distance         <dbl> 8.50, 6.97, 6.74, 6.28,
8.16, 6.48, 8.59, 9...
$ logged_activities_distance <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0...
$ very_active_distance     <dbl> 1.88, 1.57, 2.44, 2.14,
2.71, 3.19, 3.25, 3...
$ moderately_active_distance <dbl> 0.55, 0.69, 0.40, 1.26,
0.41, 0.78, 0.64, 1...
$ light_active_distance    <dbl> 6.06, 4.71, 3.91, 2.83,
5.04, 2.51, 4.71, 5...
$ sedentary_active_distance  <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0...
$ very_active_minutes      <dbl> 25, 21, 30, 29, 36, 38,
42, 50, 28, 19, 66,...
$ fairly_active_minutes    <dbl> 13, 19, 11, 34, 10, 20,
16, 31, 12, 8, 27, ...
```

```
$ lightly_active_minutes     <dbl> 328, 217, 181, 209, 221,
164, 233, 264, 205...
$ sedentary_minutes          <dbl> 728, 776, 1218, 726, 773,
539, 1149, 775, 8...
$ calories                   <dbl> 1985, 1797, 1776, 1745,
1863, 1728, 1921, 2...
$ total_time                 <dbl> 1094, 1033, 1440, 998,
1040, 761, 1440, 112...
```

Fizemos o somatório das colunas *very_active_minutes*, *fairly_active_minutes*, *lightly_active_minutes*, *sedentary_minutes* usando a função **rowSums()**

0.4 Análise dos dados

0.4.1 Analisando

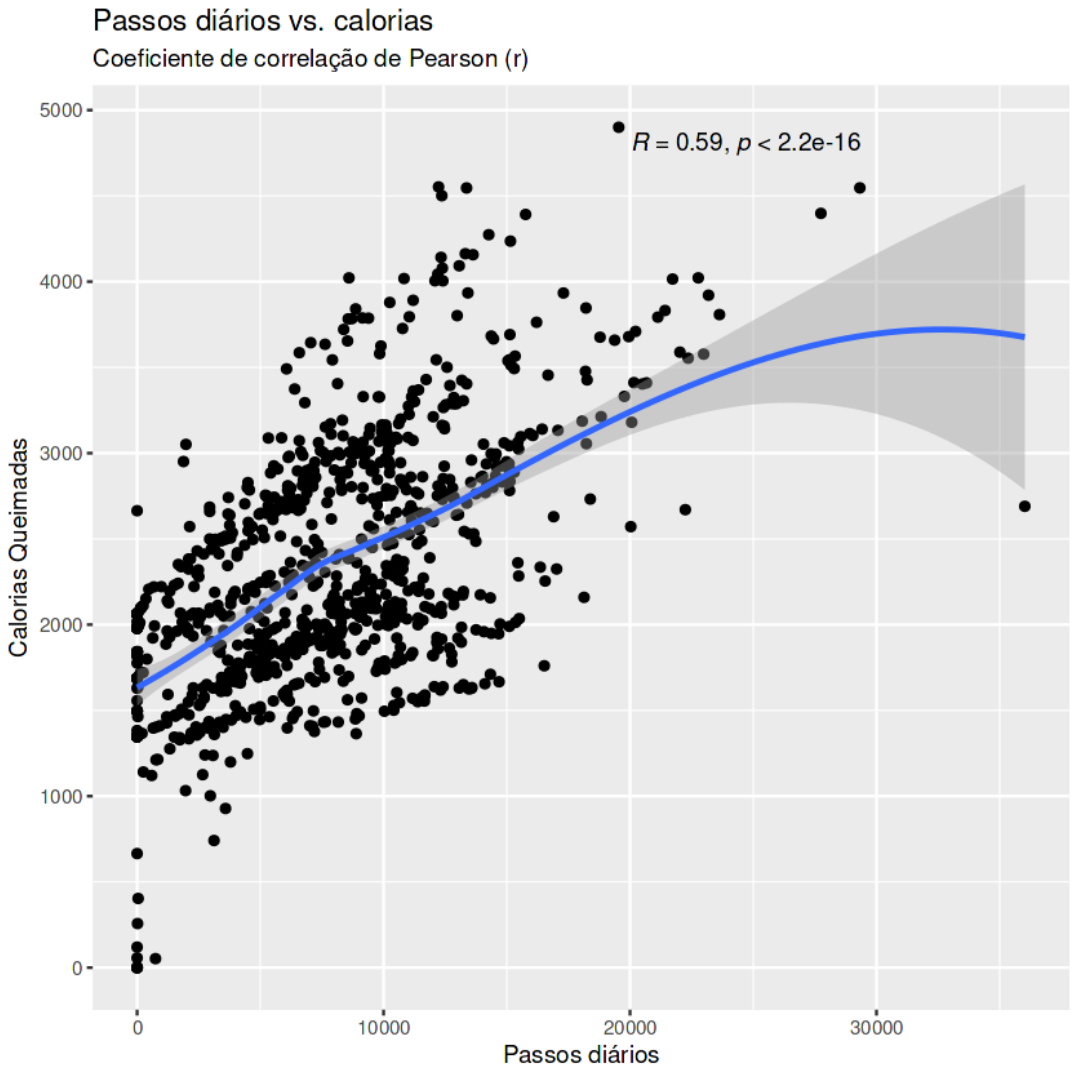
Nessa parte do projeto iremos dar nossas primeira olhada nos gráficos possiveis para o nosso BD.

0.4.1.1 Calorias x Passos

Agora vamos dar uma olhada nas calorias gastas pelos usuários e fazer uns graficos na tentativa de buscar alguma compreensão do que está acontecendo.

[20]:
```
daily_activity_file %>%
  ggplot() +
  (mapping = aes(x = total_steps, y = calories)) +
  geom_jitter() +
  geom_smooth() +
  stat_cor(method = "pearson", label.x = 20000, label.y = 4800) +
  scale_color_igv() +
  scale_fill_igv() +
  theme_grey() +
  labs(
    title = "Passos diários vs. calorias",
    subtitle = "Coeficiente de correlação de Pearson (r)",
    x = "Passos diários",
    y = "Calorias Queimadas"
  )
```

```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



A escala do coeficiente de Correlação de Pearson (r = 0,59) (correlação moderada), entre 0 e 1 é uma correlação positiva. Podemos identificar que temos uma correlação moderada isso nos diz que outros fatores podem estar afetando/colaborando para tal projeção do nosso conjunto de dados, fatores como: passos mais rápidos dos usuários...

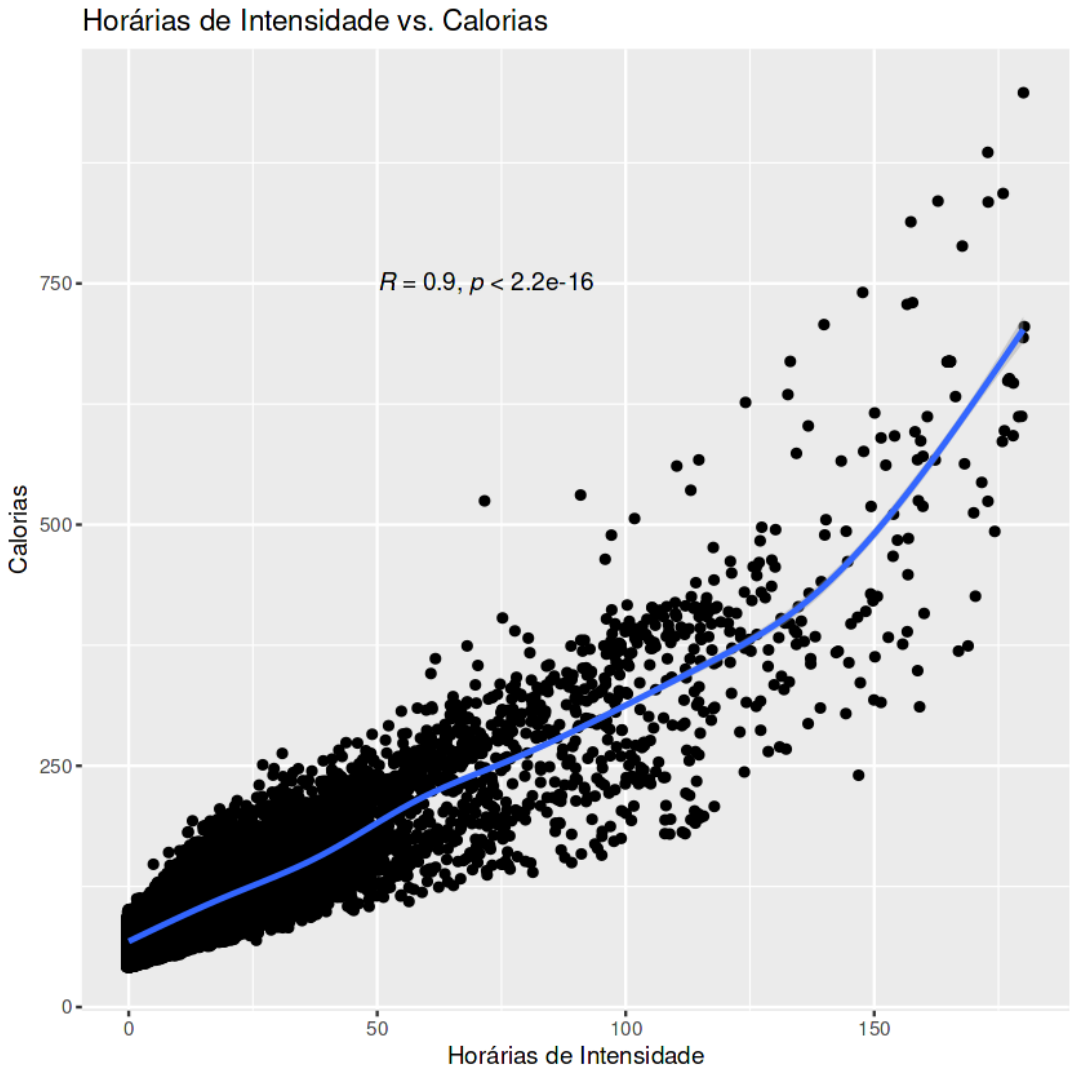
Vamos dar uma olhada na intensidade x calorias para identificamos se haverá uma correlação entre esses dados.

0.4.2 Calorias x Intensidade

[21]:
```
graph_intensidade_calorias <- hourly_intensities_file %>%
  group_by(day) %>%
  summarise(total_int = total_intensity, total_cal = calories) %>%
  ggplot() +
  (mapping = aes(x = total_int, y = total_cal)) +
  geom_jitter() +
  geom_smooth() +
```

```
        stat_cor(method = "pearson", label.x = 50, label.y = 750) +
        labs(
          title = "Horárias de Intensidade vs. Calorias",
          x = "Horárias de Intensidade",
          y = "Calorias"
        )
      graph_intensidade_calorias
```

```
`summarise()` has grouped output by 'day'. You can override using the
`.groups`
argument.
`geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs =
"cs")'
```

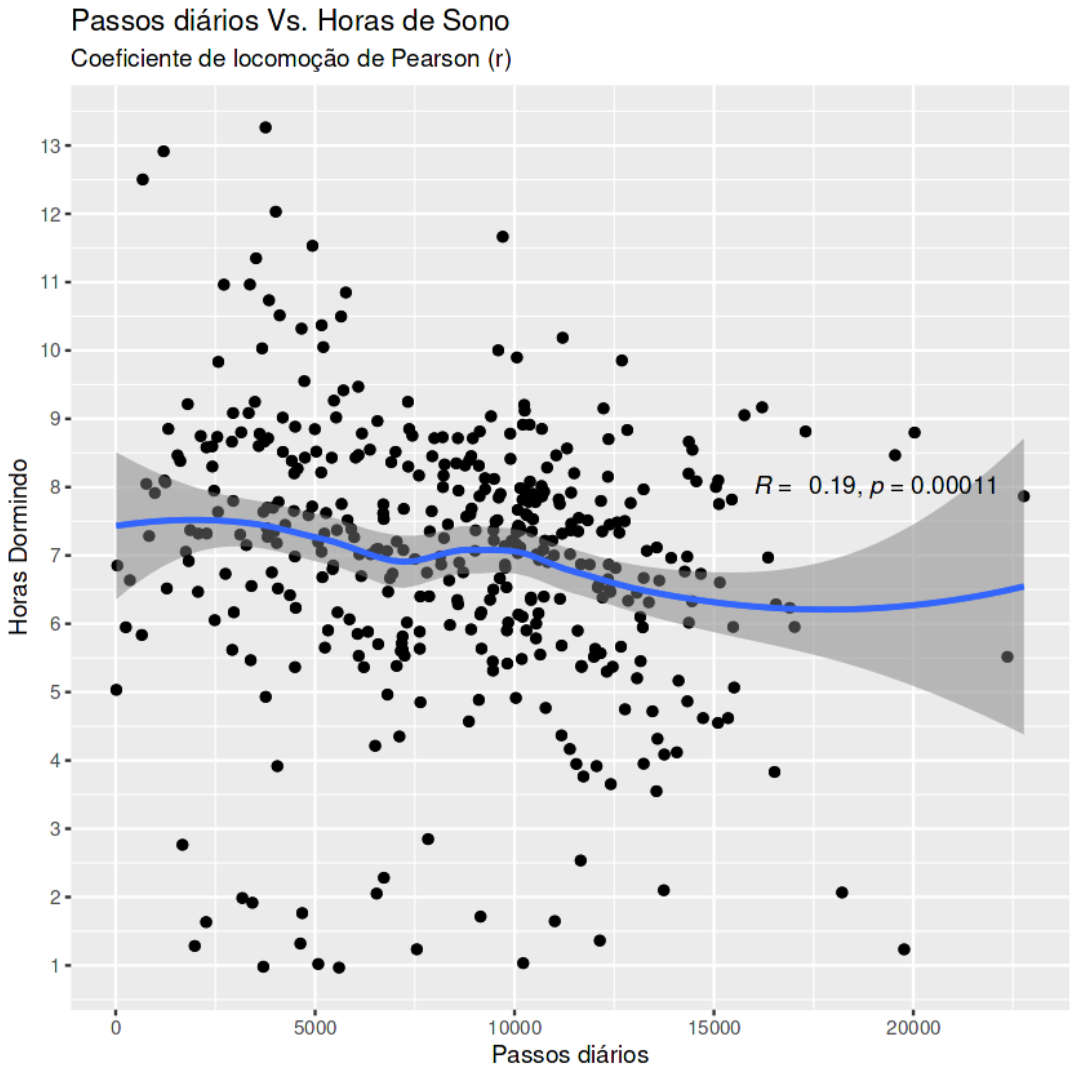


Agora podemos notar uma correlação muito alta; (R = 0,9). Quando uma variável muda, a outra variável muda na mesma direção.

0.4.3 Total de Passos diários Vs. Total de Minutos de Sono

[22]:
```
graph_passos_sono <- activity_sleep %>%
    ggplot(
     mapping = aes(x = total_steps, y = total_minutes_asleep/60)) +
      scale_y_continuous(breaks = c(0:24)) +
      geom_point() +
      geom_smooth() +
      geom_jitter() +
      geom_smooth() +
      stat_cor(method = "pearson", label.x = 16000, label.y = 8) +
      scale_color_igv() +
      scale_fill_igv() +
      theme_grey() +
      labs(
        title = "Passos diários Vs. Horas de Sono",
        subtitle = "Coeficiente de locomoção de Pearson (r)",
        x = "Passos diários",
        y = "Horas Dormindo"
      )
graph_passos_sono
```

```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



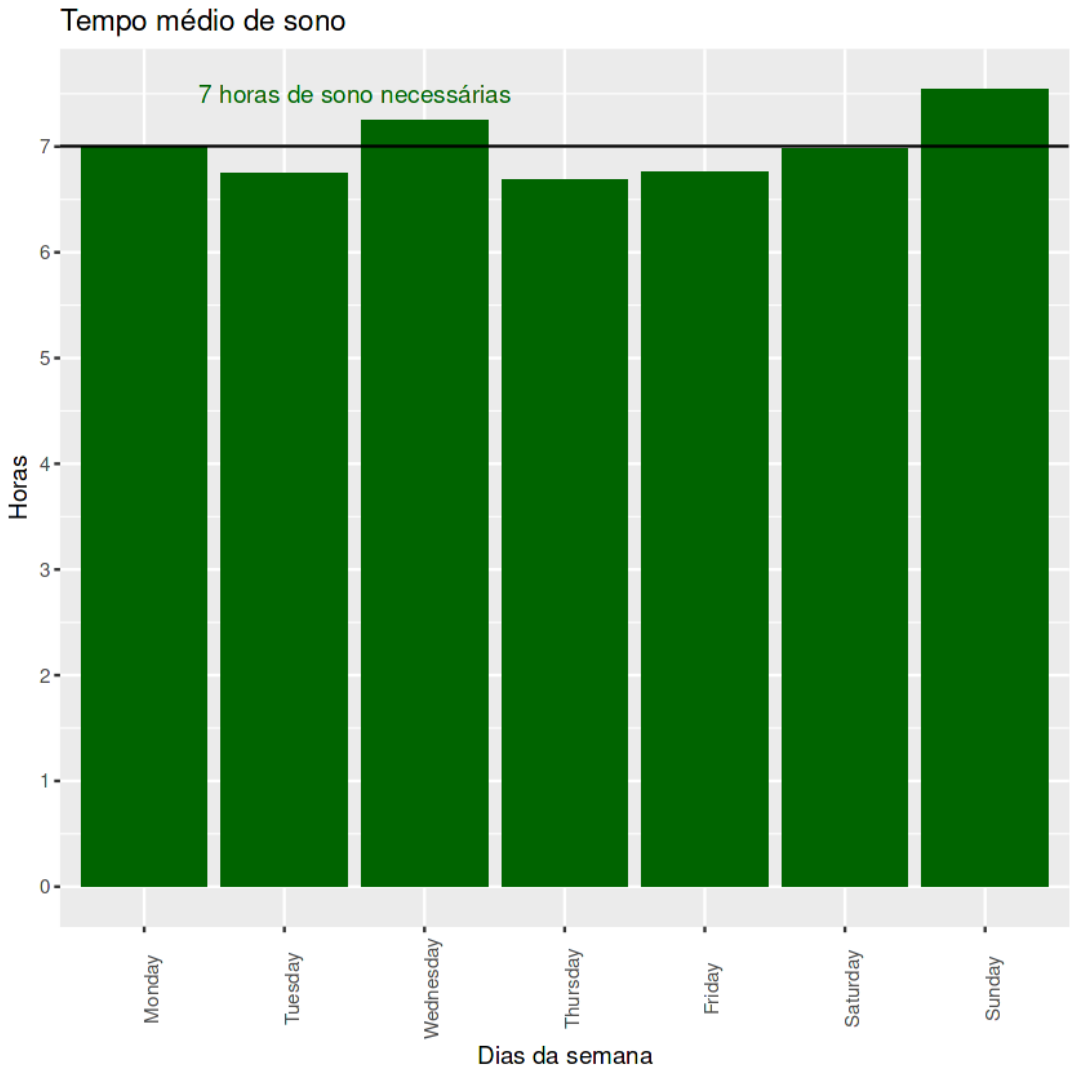
Diferentemente do gráfico de passos e calorias queimadas, em que quanto mais passos forem dados mais calorias podem ser queimadas.

Neste gráfico de passos e a quantidade de minutos que os usuários dormem por dia não há correlação entre as variáveis.

0.4.4 Tempo de sono

[23]:
```
graph_sono_medio <- sleep_day_file %>%
  group_by(day_of_week) %>%
  summarise(avg_sleep = mean(total_minutes_asleep )/60) %>%
  ggplot() +
  scale_y_continuous(breaks = c(0:24)) +
  geom_col(mapping = aes(x= day_of_week, y = avg_sleep), fill =␣
↪"darkgreen") +
  geom_hline(aes(yintercept = 7)) +
```

```
    annotate(geom="text", x=2.5, y= 7.5, label="7 horas de sono necessárias",
             color="darkgreen") +
    theme(axis.text.x = element_text(angle = 90)) +
    labs(title="Tempo médio de sono",
         y="Horas",
        x = "Dias da semana")
graph_sono_medio
```



Parece que os usuários nem sempre dormem pelo menos 7 horas por dia. Sabendo que nossos usuários são adultos devido a necessidade de consentimento para a cessão dos dados. Podemos induzir que esses usuários precisariam de 7 ou mais horas por noite, por estarem no parâmetro 18–60 anos (adultos) como sugerido pela CDC.

Última revisão: 14 de setembro de 2022 Fonte: National Center for Chronic Disease Prevention and Health Promotion, Division of Population Health

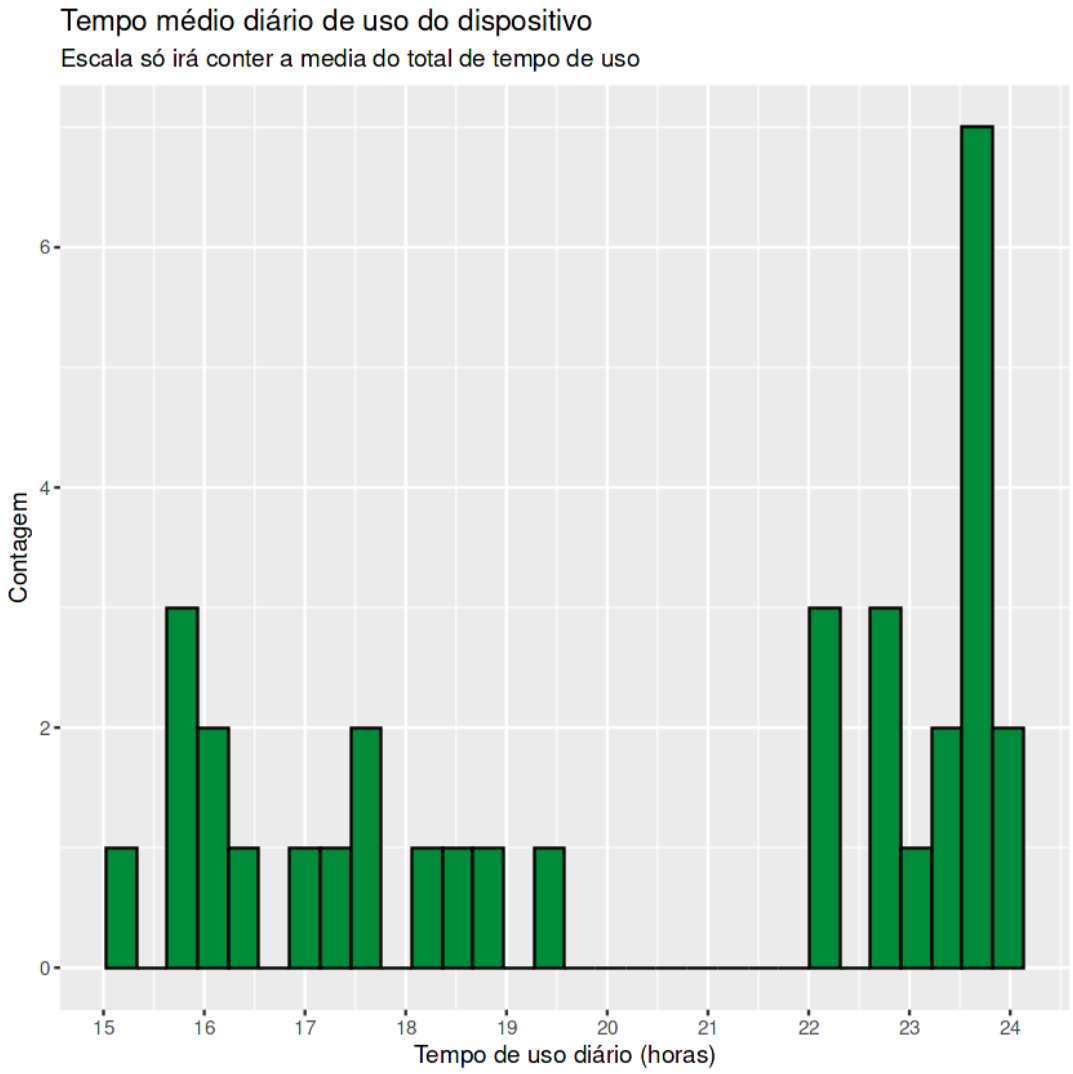
0.4.5 Apenas brincando com os gráficos

0.4.6 Uso diário dos dispositivos

Vamos dar uma espiada de como esses dados se comportam em um gráfico.

[24]:
```
daily_activity_file %>%
  group_by(id) %>%
  summarise(daily_usage_hour = mean(total_time / 60)) %>%
  ggplot(aes(x = daily_usage_hour)) +
  geom_histogram(
    color = "black", fill = "#008b3a"
  ) +
  scale_color_igv() +
  scale_fill_igv() +
  theme_grey() +
  scale_x_continuous(breaks = c(1:24)) +
  labs(
    title = "Tempo médio diário de uso do dispositivo",
    subtitle = "Escala só irá conter a media do total de tempo de uso",
    x = "Tempo de uso diário (horas)",
    y = "Contagem"
  )
```

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



podemos identificar no gráfico a media de tempo que os usuarios passam usando o dispositivo inteligente.

foi usado so pipes antifos %>% pois o kaggle nao estava identificando os novos |>

0.4.7 Multi gráficos

[25]:
```
curr_locale <- Sys.getlocale("LC_TIME")
Sys.setlocale("LC_TIME","pt_BR")

#<specific code for graph generation>

#Sys.setlocale("LC_TIME",curr_locale)
```

```
Warning message in Sys.setlocale("LC_TIME", "pt_BR"):
"OS reports request to set locale to "pt_BR" cannot be honored"
```

”

tentativa de mudar a localização para fazer uso da função weekdays em português. Entretanto, não foi possível.

[26]:
```
        weekday_steps_sleep <- activity_sleep %>%
          mutate(day_of_week = weekdays(date))



        weekday_steps_sleep$day_of_week <-ordered(weekday_steps_sleep$day_of_week,␣
↪levels=c("Monday", "Tuesday", "Wednesday", "Thursday",
        "Friday", "Saturday", "Sunday"))

         weekday_steps_sleep <- weekday_steps_sleep %>%
          group_by(day_of_week) %>%
          summarize (daily_steps = mean(total_steps), daily_sleep_hour =␣
↪mean(total_minutes_asleep)/60)

        head(weekday_steps_sleep)

        weekday_steps_sleep$day_of_week <- c("Segunda", "Terca", "Quarta", "Quinta",
        "Sexta", "Sabado", "Domingo")
        weekday_steps_sleep$day_of_week <-ordered(weekday_steps_sleep$day_of_week,␣
↪levels=c("Segunda", "Terca", "Quarta", "Quinta",
        "Sexta", "Sabado", "Domingo"))



        head(weekday_steps_sleep)
```

A tibble: 6 × 3

day_of_week <ord>	daily_steps <dbl>	daily_sleep_hour <dbl>
Monday	9273.217	6.991667
Tuesday	9182.692	6.742308
Wednesday	8022.864	7.244697
Thursday	8183.516	6.688281
Friday	7901.404	6.757018
Saturday	9871.123	6.984503

A tibble: 6 × 3

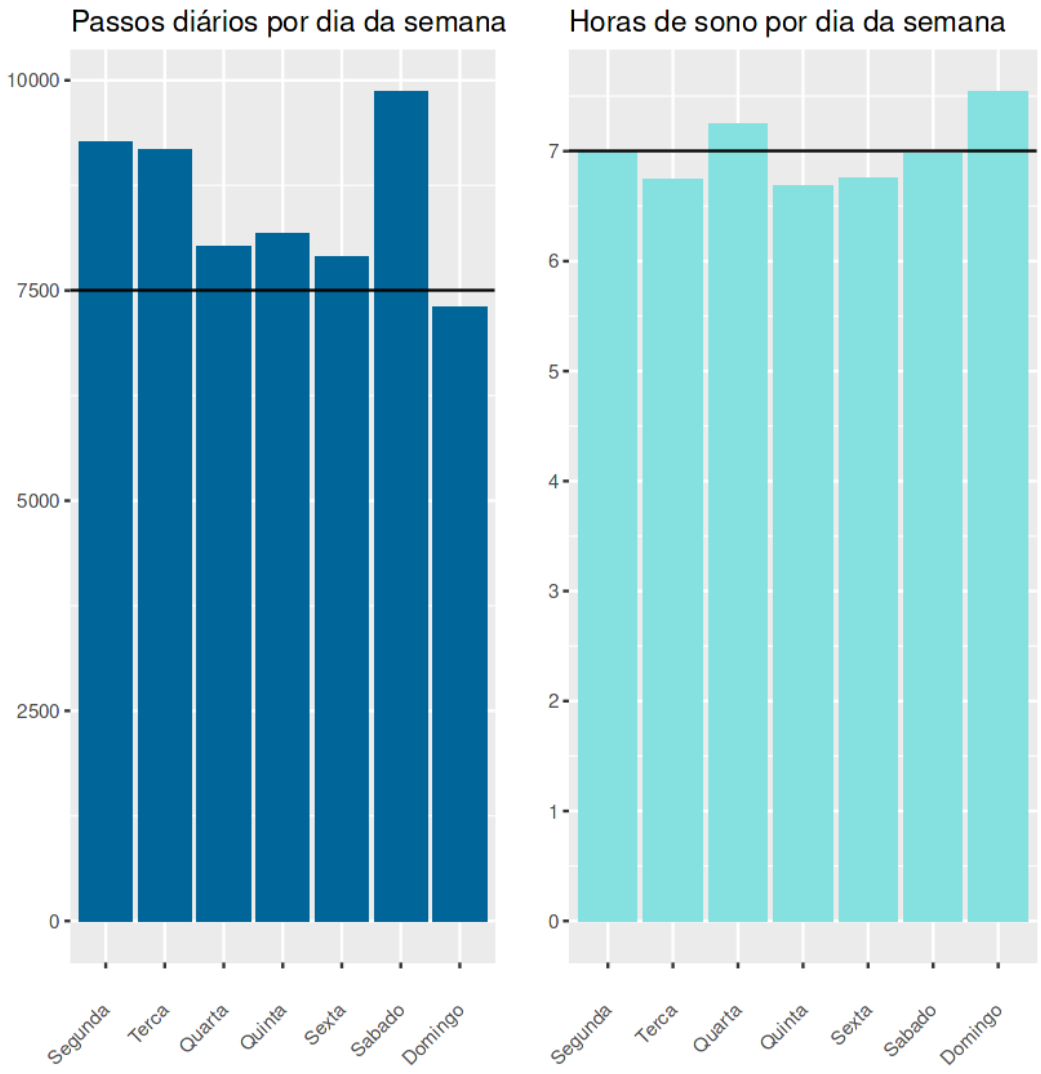
day_of_week <ord>	daily_steps <dbl>	daily_sleep_hour <dbl>
Segunda	9273.217	6.991667
Terca	9182.692	6.742308
Quarta	8022.864	7.244697
Quinta	8183.516	6.688281
Sexta	7901.404	6.757018
Sabado	9871.123	6.984503

[27]:
```
ggarrange(

    ggplot(weekday_steps_sleep) +
      geom_col(aes(day_of_week, daily_steps), fill = "#006699") +
      geom_hline(yintercept = 7500) +
      labs(title = "Passos diários por dia da semana", x= "", y = "") +
      theme(axis.text.x = element_text(angle = 45,vjust = 0.5, hjust = 1)),

    ggplot(weekday_steps_sleep, aes(day_of_week, daily_sleep_hour)) +
      geom_col(fill = "#85e0e0") +
      scale_y_continuous(breaks = c(0:24)) +
      geom_hline(yintercept = 7) +
      labs(title = "Horas de sono por dia da semana", x= "", y = "") +
      theme(axis.text.x = element_text(angle = 45,vjust = 0.5, hjust = 1))
  ) -> bar_weekday_steps_sleep

bar_weekday_steps_sleep
```

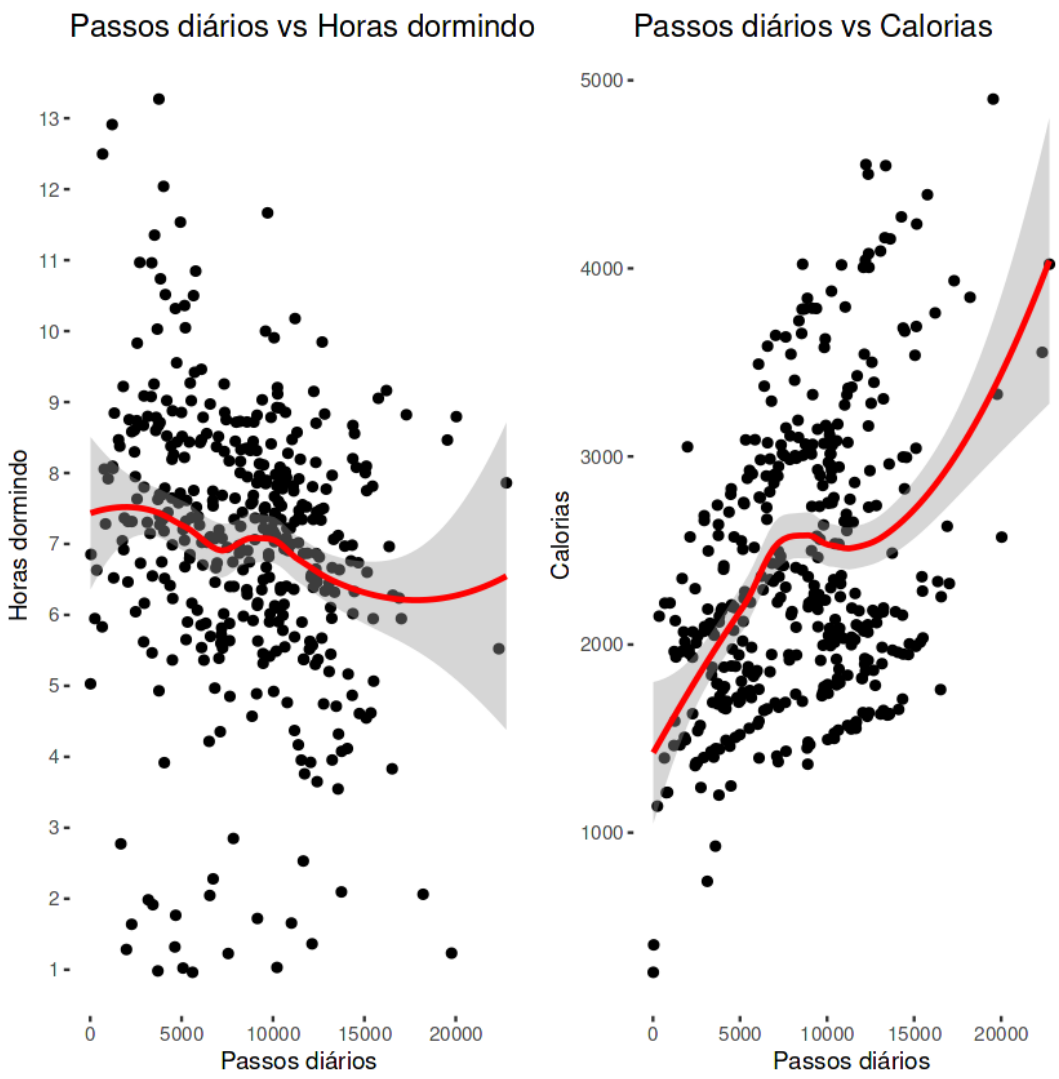


[28]:
```
max(activity_sleep$total_minutes_asleep/60)
```

13.2666666666667

[29]:
```
ggarrange(
ggplot(activity_sleep, aes(x=total_steps, y=total_minutes_asleep/60))+
  scale_y_continuous(breaks = c(0:24)) +
  geom_jitter() +
  geom_smooth(color = "red") +
  labs(title = "Passos diários vs Horas dormindo",
       x = "Passos diários",
       y= "Horas dormindo") +
   theme(panel.background = element_blank(),
        plot.title = element_text( size=14)),
ggplot(activity_sleep, aes(x=total_steps, y=calories))+
  geom_jitter() +
  geom_smooth(color = "red") +
  labs(title = "Passos diários vs Calorias",
       x = "Passos diários",
       y= "Calorias") +
   theme(panel.background = element_blank(),
        plot.title = element_text( size=14))
) -> jitter_steps_sleep_calories
jitter_steps_sleep_calories
```

```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



0.5 Conclusão

Quais são algumas tendências no uso de dispositivos inteligentes?

0.5.1 Achados

0.5.1.1 Qualidade de sono

Com base em nossos resultados, podemos ver que os usuários dormem menos de 8 horas por dia. Esse grupo de pessoas poderiam definir em nosso aplicativo um horário desejado para dormir e receber uma notificação minutos antes para se prepararem para dormir. O aplicativo poderia fazer sugestões de atitudes a serem tomadas para que os usuários melhorem a qualidade de seu sono. Por exemplo: Não comer muito (2-3 horas) antes de antes de ir dormir (ref. 1), evitar luzes fortes no período da noite especialemte luzes do teto e celulares entre 22:00 e 04:00 (ref. Sleep Toolkit: Tools for Optimizing Sleep & Sleep-Wake Timing); exercícios leves (caminhada, ciclismo, corrida) ao ar livre no final da tarde pode ajudar a mitigar os problema com o sono oriundos do uso de aparelhos eletrônicos na parte da noite

(ref. Master Your Sleep & Be More Alert When Awake, Sleep Toolkit: Tools for Optimizing Sleep & Sleep-Wake Timing), evitar o consumo de cafeina na parte da tarde e noite (de 8 a 10 horas antes de dormir) (ref. Sleep Toolkit: Tools for Optimizing Sleep & Sleep-Wake Timing), recomendar a pratica de meditações que ajudem a relaxar e músicas para ajudar a cair no sono (ref. Sleep Toolkit: Tools for Optimizing Sleep & Sleep-Wake Timing, Dr. Matthew Walker: The Science & Practice of Perfecting Your Sleep) e ver a luz do sol saindo de casa 30 a 60 minutos depois de acordar (ref. Toolkit for Sleep).

0.5.1.2 Calorias e Intensidade

Apesar de trivial foi possível identificar, através de nossas análises, que a *Intensidade* da atividade do usuário refletiu na quantidade de *Calorias* queimadas. Existindo uma forte correlação entre as duas variáveis (r = 0.9).

0.5.1.3 Tempo de sono e passos no dia

Quebrando as espectativas a quantidade de sono não afetou a quantidade de passos no dia desse grupo de usuários. É Necessário mais dados para que possamos elaborar uma explicação para esse fenfenômeno. Pois esse grupo de usuários podem fazer parte de uma categoria de pessoas que andar é parte crucial de sua rotina. Portanto, teriamos dados enviesados por estarmos "olhando" para apenas uma parcela de nossos clientes.

0.5.2 Recomendações

1. Coletar mais dados

O conjunto de dados baseado nos rastreadores pessoais Fitbit fornece informações úteis sobre as preferências do usuário com as quais a empresa pode aprender. Entretanto, o conjunto de dados fornece uma baixa variação de usuários para que possamos entender o perfil dos nossos clientes, contendo, apenas um conjunto de 33 usuários. Portanto, os conjuntos de dados usados têm uma amostra pequena de dados e podem ser tendenciosos, pois não tínhamos nenhum detalhe demográfico dos usuários, e/ou dados categóricos. O nosso time de Dados recomenda a obtenção de mais dados para que possamos fundar nossas hypothesis com mais robustez.

Como fazer: Seguindo as recomendações da Lei Geral de Proteção de Dados (LGPD).

Podemos fazer esse processo de obtenção de mais dados pela inclusão de descontos para a compra adicional ou recompensa ao usuário por meio do aplicativo Fitbit por um tempo de uso mais longo para aqueles que aceitarem preencher os formulários e/ou conceder acesso aos seus dados de uso. Entre essas formas podemos, também, fazer a coleta dos cookies com o estilo de navegação dos nossos usuários. É de extremamente importância estarmos seguindo as recomendações de regras sobre o uso e concessão de dados pessoais presentes na LGPD.