Course 2 Section 2.5 - Working with dates and times

Jiaying Wu

05/10/2020

```
library(tidyverse)
library(rwalkr)
library(lubridate)
```

```
ped <- melb_walk_fast(2018, "Melbourne Central")
ped
```

```
## # A tibble: 8,760 x 5
##    Sensor            Date_Time           Date        Time Count
##    <chr>             <dttm>              <date>     <int> <int>
##  1 Melbourne Central 2018-01-01 00:00:00 2018-01-01     0  2996
##  2 Melbourne Central 2018-01-01 01:00:00 2018-01-01     1  3481
##  3 Melbourne Central 2018-01-01 02:00:00 2018-01-01     2  1721
##  4 Melbourne Central 2018-01-01 03:00:00 2018-01-01     3  1056
##  5 Melbourne Central 2018-01-01 04:00:00 2018-01-01     4   417
##  6 Melbourne Central 2018-01-01 05:00:00 2018-01-01     5   222
##  7 Melbourne Central 2018-01-01 06:00:00 2018-01-01     6   110
##  8 Melbourne Central 2018-01-01 07:00:00 2018-01-01     7   180
##  9 Melbourne Central 2018-01-01 08:00:00 2018-01-01     8   205
## 10 Melbourne Central 2018-01-01 09:00:00 2018-01-01     9   326
## # ... with 8,750 more rows
```

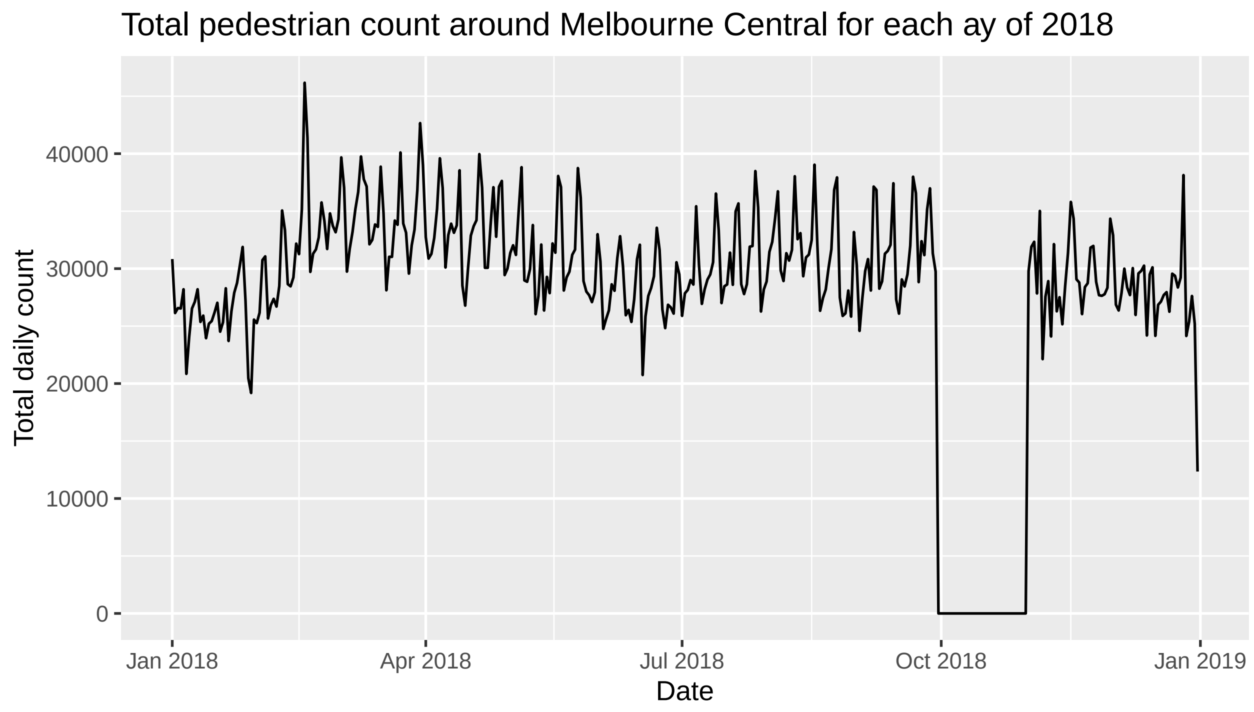
Using the pedestrian counts for Melbourne Central answer the following:

Q1.Is the pedestrian count different for each day of the year?

To answer this question, use the wrangling verbs, group_by() and summarise() to compute the total pedestrian counts for each day (set na.rm = TRUE). The following line plot represents the total pedestrian counts for each day. Note that ped in this exercise contains pedestrian count data for each hour of each day in 2018 based on the sensor located in Melbourne Central.

```
count_by_date <- ped %>%
  group_by(Date) %>%
  summarise(sum_count = sum(Count, na.rm = TRUE))

count_by_date %>%
  ggplot(aes(x = Date, y = sum_count)) +
  geom_line() +
  ggtitle("Total pedestrian count around Melbourne Central for each ay of 2018") +
  ylab("Total daily count")
```



Q2.Is the central tendency and variability of daily pedestrian counts by each day of the week different? To answer this question:

- You will need to use the data frame that you generated from the previous question, which contains the total daily counts.
- Create a day of the week variable with the wday() function.
- Use the wrangling verbs group_by() and summarise() to compute the mean and standard deviation daily counts over each day of the week (set na.rm = TRUE for both computations).

```r
count_by_date %>%
  mutate(wday = wday(Date)) %>%
  group_by(wday) %>%
  summarise(mean = sum(sum_count, na.rm = TRUE),
            std = sd(sum_count, na.rm = TRUE))
```

```
## # A tibble: 7 x 3
##    wday    mean    std
##   <dbl>   <int>  <dbl>
## 1     1 1315409  9024.
## 2     2 1356292  8995.
## 3     3 1364580  8989.
## 4     4 1436603  9535.
## 5     5 1450099  8744.
## 6     6 1640274 10239.
## 7     7 1584439  9823.
```