BIKESHARE.PROJECT

JEAN YVENS ALBERUS

2023-02-23

# installing required packages

install.packages(“tidyverse”) install.packages(“lubridate”) install.packages(“ggplot”)

#loading the packages

library(tidyverse)

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──  
## ✔ ggplot2 3.3.6 ✔ purrr 0.3.4  
## ✔ tibble 3.1.7 ✔ dplyr 1.0.9  
## ✔ tidyr 1.2.0 ✔ stringr 1.4.0  
## ✔ readr 2.1.2 ✔ forcats 0.5.1  
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()

library(lubridate)

##   
## Attaching package: 'lubridate'  
##   
## The following objects are masked from 'package:base':  
##   
## date, intersect, setdiff, union

library(ggplot2)

#setting working environment

getwd()

## [1] "C:/Users/Jean Yvens Alberus/Desktop/Caspstone\_BikeshareProject/CSV\_Bikeshare\_data"

setwd("C:/Users/Jean Yvens Alberus/Desktop/Caspstone\_BikeshareProject/CSV\_Bikeshare\_data")

#loadind the data (first half of 2022)

jan\_data <- read\_csv("202201\_divvy\_tripdata.csv")

## Rows: 103770 Columns: 13  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (7): ride\_id, rideable\_type, start\_station\_name, start\_station\_id, end\_...  
## dbl (4): start\_lat, start\_lng, end\_lat, end\_lng  
## dttm (2): started\_at, ended\_at  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

feb\_data <-read\_csv("202202\_divvy\_tripdata.csv")

## Rows: 115609 Columns: 13  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (7): ride\_id, rideable\_type, start\_station\_name, start\_station\_id, end\_...  
## dbl (4): start\_lat, start\_lng, end\_lat, end\_lng  
## dttm (2): started\_at, ended\_at  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

mar\_data <- read\_csv("202203\_divvy\_tripdata.csv")

## Rows: 284042 Columns: 13  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (7): ride\_id, rideable\_type, start\_station\_name, start\_station\_id, end\_...  
## dbl (4): start\_lat, start\_lng, end\_lat, end\_lng  
## dttm (2): started\_at, ended\_at  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

apr\_data <- read\_csv("202204\_divvy\_tripdata.csv")

## Rows: 371249 Columns: 13  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (7): ride\_id, rideable\_type, start\_station\_name, start\_station\_id, end\_...  
## dbl (4): start\_lat, start\_lng, end\_lat, end\_lng  
## dttm (2): started\_at, ended\_at  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

may\_data <- read\_csv("202205\_divvy\_tripdata.csv")

## Rows: 634858 Columns: 13  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (7): ride\_id, rideable\_type, start\_station\_name, start\_station\_id, end\_...  
## dbl (4): start\_lat, start\_lng, end\_lat, end\_lng  
## dttm (2): started\_at, ended\_at  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

jun\_data <- read\_csv("202206\_divvy\_tripdata.csv")

## Rows: 769204 Columns: 13  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (7): ride\_id, rideable\_type, start\_station\_name, start\_station\_id, end\_...  
## dbl (4): start\_lat, start\_lng, end\_lat, end\_lng  
## dttm (2): started\_at, ended\_at  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

# DATA CLEANING AND TRANSFORMATION

#checking colnames before we merge the data

colnames(jan\_data)

## [1] "ride\_id" "rideable\_type" "started\_at"   
## [4] "ended\_at" "start\_station\_name" "start\_station\_id"   
## [7] "end\_station\_name" "end\_station\_id" "start\_lat"   
## [10] "start\_lng" "end\_lat" "end\_lng"   
## [13] "member\_casual"

colnames(feb\_data)

## [1] "ride\_id" "rideable\_type" "started\_at"   
## [4] "ended\_at" "start\_station\_name" "start\_station\_id"   
## [7] "end\_station\_name" "end\_station\_id" "start\_lat"   
## [10] "start\_lng" "end\_lat" "end\_lng"   
## [13] "member\_casual"

colnames(mar\_data)

## [1] "ride\_id" "rideable\_type" "started\_at"   
## [4] "ended\_at" "start\_station\_name" "start\_station\_id"   
## [7] "end\_station\_name" "end\_station\_id" "start\_lat"   
## [10] "start\_lng" "end\_lat" "end\_lng"   
## [13] "member\_casual"

colnames(apr\_data)

## [1] "ride\_id" "rideable\_type" "started\_at"   
## [4] "ended\_at" "start\_station\_name" "start\_station\_id"   
## [7] "end\_station\_name" "end\_station\_id" "start\_lat"   
## [10] "start\_lng" "end\_lat" "end\_lng"   
## [13] "member\_casual"

colnames(may\_data)

## [1] "ride\_id" "rideable\_type" "started\_at"   
## [4] "ended\_at" "start\_station\_name" "start\_station\_id"   
## [7] "end\_station\_name" "end\_station\_id" "start\_lat"   
## [10] "start\_lng" "end\_lat" "end\_lng"   
## [13] "member\_casual"

colnames(jun\_data)

## [1] "ride\_id" "rideable\_type" "started\_at"   
## [4] "ended\_at" "start\_station\_name" "start\_station\_id"   
## [7] "end\_station\_name" "end\_station\_id" "start\_lat"   
## [10] "start\_lng" "end\_lat" "end\_lng"   
## [13] "member\_casual"

##renaming columns not necessary cause they are already identical

# Inspect the dataframes and look for incongruencies

str(jan\_data) str(feb\_data) str(mar\_data) str(apr\_data) str(may\_data) str(jun\_data)}

# Convert ride\_id and rideable\_type to character so that they can stack correctly

jan\_data <- mutate(jan\_data, ride\_id = as.character(ride\_id)  
 ,rideable\_type = as.character(rideable\_type))   
feb\_data <- mutate(feb\_data, ride\_id = as.character(ride\_id)  
 ,rideable\_type = as.character(rideable\_type))   
mar\_data <- mutate(mar\_data, ride\_id = as.character(ride\_id)  
 ,rideable\_type = as.character(rideable\_type))   
apr\_data<- mutate(apr\_data, ride\_id = as.character(ride\_id)  
 ,rideable\_type = as.character(rideable\_type))   
may\_data <- mutate(may\_data, ride\_id = as.character(ride\_id)  
 ,rideable\_type = as.character(rideable\_type))   
jun\_data <- mutate(may\_data, ride\_id = as.character(ride\_id)  
 ,rideable\_type = as.character(rideable\_type))

# Stacking individual month data frames into one big data frame

all\_trips <- bind\_rows(jan\_data, feb\_data, mar\_data, apr\_data, may\_data, jun\_data)  
  
head(all\_trips)

## # A tibble: 6 × 13  
## ride\_id rideable\_type started\_at ended\_at start\_station\_n…  
## <chr> <chr> <dttm> <dttm> <chr>   
## 1 C2F7DD… electric\_bike 2022-01-13 11:59:47 2022-01-13 12:02:44 Glenwood Ave & …  
## 2 A6CF89… electric\_bike 2022-01-10 08:41:56 2022-01-10 08:46:17 Glenwood Ave & …  
## 3 BD0F91… classic\_bike 2022-01-25 04:53:40 2022-01-25 04:58:01 Sheffield Ave &…  
## 4 CBB80E… classic\_bike 2022-01-04 00:18:04 2022-01-04 00:33:00 Clark St & Bryn…  
## 5 DDC963… classic\_bike 2022-01-20 01:31:10 2022-01-20 01:37:12 Michigan Ave & …  
## 6 A39C6F… classic\_bike 2022-01-11 18:48:09 2022-01-11 18:51:31 Wood St & Chica…  
## # … with 8 more variables: start\_station\_id <chr>, end\_station\_name <chr>,  
## # end\_station\_id <chr>, start\_lat <dbl>, start\_lng <dbl>, end\_lat <dbl>,  
## # end\_lng <dbl>, member\_casual <chr>

directory <- getwd()  
  
colnames(all\_trips) #List of column names

## [1] "ride\_id" "rideable\_type" "started\_at"   
## [4] "ended\_at" "start\_station\_name" "start\_station\_id"   
## [7] "end\_station\_name" "end\_station\_id" "start\_lat"   
## [10] "start\_lng" "end\_lat" "end\_lng"   
## [13] "member\_casual"

dim(all\_trips) #Dimensions of the data frame?

## [1] 2144386 13

head(all\_trips) #See the first 6 rows of data frame. Also

## # A tibble: 6 × 13  
## ride\_id rideable\_type started\_at ended\_at start\_station\_n…  
## <chr> <chr> <dttm> <dttm> <chr>   
## 1 C2F7DD… electric\_bike 2022-01-13 11:59:47 2022-01-13 12:02:44 Glenwood Ave & …  
## 2 A6CF89… electric\_bike 2022-01-10 08:41:56 2022-01-10 08:46:17 Glenwood Ave & …  
## 3 BD0F91… classic\_bike 2022-01-25 04:53:40 2022-01-25 04:58:01 Sheffield Ave &…  
## 4 CBB80E… classic\_bike 2022-01-04 00:18:04 2022-01-04 00:33:00 Clark St & Bryn…  
## 5 DDC963… classic\_bike 2022-01-20 01:31:10 2022-01-20 01:37:12 Michigan Ave & …  
## 6 A39C6F… classic\_bike 2022-01-11 18:48:09 2022-01-11 18:51:31 Wood St & Chica…  
## # … with 8 more variables: start\_station\_id <chr>, end\_station\_name <chr>,  
## # end\_station\_id <chr>, start\_lat <dbl>, start\_lng <dbl>, end\_lat <dbl>,  
## # end\_lng <dbl>, member\_casual <chr>

tail(all\_trips)

## # A tibble: 6 × 13  
## ride\_id rideable\_type started\_at ended\_at start\_station\_n…  
## <chr> <chr> <dttm> <dttm> <chr>   
## 1 284843… classic\_bike 2022-05-30 18:34:44 2022-05-31 19:34:35 Ashland Ave & L…  
## 2 8891BA… electric\_bike 2022-05-27 22:00:02 2022-05-27 22:07:01 Clark St & Newp…  
## 3 47D8B5… electric\_bike 2022-05-15 16:05:39 2022-05-15 16:44:12 Clark St & Newp…  
## 4 AA8D16… electric\_bike 2022-05-21 10:10:13 2022-05-21 10:26:09 Francisco Ave &…  
## 5 897EBF… electric\_bike 2022-05-12 07:53:58 2022-05-12 08:01:18 Francisco Ave &…  
## 6 AAC23A… electric\_bike 2022-05-11 21:14:28 2022-05-11 21:18:16 Clark St & Newp…  
## # … with 8 more variables: start\_station\_id <chr>, end\_station\_name <chr>,  
## # end\_station\_id <chr>, start\_lat <dbl>, start\_lng <dbl>, end\_lat <dbl>,  
## # end\_lng <dbl>, member\_casual <chr>

str(all\_trips) #See list of columns and data types (numeric, character, etc)

## tibble [2,144,386 × 13] (S3: tbl\_df/tbl/data.frame)  
## $ ride\_id : chr [1:2144386] "C2F7DD78E82EC875" "A6CF8980A652D272" "BD0F91DFF741C66D" "CBB80ED419105406" ...  
## $ rideable\_type : chr [1:2144386] "electric\_bike" "electric\_bike" "classic\_bike" "classic\_bike" ...  
## $ started\_at : POSIXct[1:2144386], format: "2022-01-13 11:59:47" "2022-01-10 08:41:56" ...  
## $ ended\_at : POSIXct[1:2144386], format: "2022-01-13 12:02:44" "2022-01-10 08:46:17" ...  
## $ start\_station\_name: chr [1:2144386] "Glenwood Ave & Touhy Ave" "Glenwood Ave & Touhy Ave" "Sheffield Ave & Fullerton Ave" "Clark St & Bryn Mawr Ave" ...  
## $ start\_station\_id : chr [1:2144386] "525" "525" "TA1306000016" "KA1504000151" ...  
## $ end\_station\_name : chr [1:2144386] "Clark St & Touhy Ave" "Clark St & Touhy Ave" "Greenview Ave & Fullerton Ave" "Paulina St & Montrose Ave" ...  
## $ end\_station\_id : chr [1:2144386] "RP-007" "RP-007" "TA1307000001" "TA1309000021" ...  
## $ start\_lat : num [1:2144386] 42 42 41.9 42 41.9 ...  
## $ start\_lng : num [1:2144386] -87.7 -87.7 -87.7 -87.7 -87.6 ...  
## $ end\_lat : num [1:2144386] 42 42 41.9 42 41.9 ...  
## $ end\_lng : num [1:2144386] -87.7 -87.7 -87.7 -87.7 -87.6 ...  
## $ member\_casual : chr [1:2144386] "casual" "casual" "member" "casual" ...

summary(all\_trips) #Statistical summary of data. Mainly for numerics

## ride\_id rideable\_type started\_at   
## Length:2144386 Length:2144386 Min. :2022-01-01 00:00:05.0   
## Class :character Class :character 1st Qu.:2022-04-04 17:09:44.0   
## Mode :character Mode :character Median :2022-05-08 16:46:29.0   
## Mean :2022-04-23 16:48:50.7   
## 3rd Qu.:2022-05-20 06:26:12.0   
## Max. :2022-05-31 23:59:56.0   
##   
## ended\_at start\_station\_name start\_station\_id   
## Min. :2022-01-01 00:01:48.00 Length:2144386 Length:2144386   
## 1st Qu.:2022-04-04 17:22:19.25 Class :character Class :character   
## Median :2022-05-08 17:06:49.00 Mode :character Mode :character   
## Mean :2022-04-23 17:08:20.91   
## 3rd Qu.:2022-05-20 06:47:17.00   
## Max. :2022-06-02 11:35:01.00   
##   
## end\_station\_name end\_station\_id start\_lat start\_lng   
## Length:2144386 Length:2144386 Min. :41.65 Min. :-87.84   
## Class :character Class :character 1st Qu.:41.88 1st Qu.:-87.66   
## Mode :character Mode :character Median :41.90 Median :-87.64   
## Mean :41.90 Mean :-87.65   
## 3rd Qu.:41.93 3rd Qu.:-87.63   
## Max. :45.64 Max. :-73.80   
##   
## end\_lat end\_lng member\_casual   
## Min. :41.63 Min. :-88.14 Length:2144386   
## 1st Qu.:41.88 1st Qu.:-87.66 Class :character   
## Median :41.90 Median :-87.64 Mode :character   
## Mean :41.90 Mean :-87.65   
## 3rd Qu.:41.93 3rd Qu.:-87.63   
## Max. :42.11 Max. :-87.51   
## NA's :2190 NA's :2190

# We will want to add a calculated field for length of ride since the raw data did not have the “tripduration” column. We will add “ride\_length” to the entire dataframe for consistency.

## if There are some rides where tripduration shows up as negative, including several hundred rides where Divvy took bikes out of circulation for Quality Control reasons. We will want to delete these rides.

table(all\_trips$member\_casual)

##   
## casual member   
## 817065 1327321

# Reassign to the desired values (we will go with the current 2020 labels)

all\_trips <- all\_trips %>%   
 mutate(member\_casual = recode(member\_casual  
 ,"Subscriber" = "member"  
 ,"Customer" = "casual"))  
  
table(all\_trips$member\_casual)

##   
## casual member   
## 817065 1327321

all\_trips$date <- as.Date(all\_trips$started\_at) #The default format is yyyy-mm-dd  
  
all\_trips$month <- format(as.Date(all\_trips$date), "%m")  
all\_trips$day <- format(as.Date(all\_trips$date), "%d")  
all\_trips$year <- format(as.Date(all\_trips$date), "%Y")  
all\_trips$day\_of\_week <- format(as.Date(all\_trips$date), "%A")

# Adding a “ride\_length” calculation to all\_trips

all\_trips$ride\_length <- difftime(all\_trips$ended\_at,all\_trips$started\_at)

# Remove “bad” data

## The dataframe includes a few hundred entries when bikes were taken out of docks and checked for quality by Divvy or ride\_length was negative

all\_trips\_v2 <- all\_trips[!(all\_trips$start\_station\_name == "HQ QR" | all\_trips$ride\_length<0),]

# CONDUCT DESCRIPTIVE ANALYSIS

##=====================================

# Descriptive analysis on ride\_length (all figures in seconds)

mean(all\_trips\_v2$ride\_length) #straight average (total ride length / rides)

## Time difference of NA secs

median(all\_trips\_v2$ride\_length)

## Time difference of NA secs

max(all\_trips\_v2$ride\_length) #longest ride

## Time difference of NA secs

min(all\_trips\_v2$ride\_length) #shortest ride

## Time difference of NA secs

summary(all\_trips\_v2$ride\_length)

## Length Class Mode   
## 2144382 difftime numeric

# Compare members and casual users

aggregate(all\_trips\_v2$ride\_length ~ all\_trips\_v2$member\_casual, FUN = mean)

## all\_trips\_v2$member\_casual all\_trips\_v2$ride\_length  
## 1 casual 1994.5223 secs  
## 2 member 769.7888 secs

aggregate(all\_trips\_v2$ride\_length ~ all\_trips\_v2$member\_casual, FUN = median)

## all\_trips\_v2$member\_casual all\_trips\_v2$ride\_length  
## 1 casual 921 secs  
## 2 member 529 secs

aggregate(all\_trips\_v2$ride\_length ~ all\_trips\_v2$member\_casual, FUN = max)

## all\_trips\_v2$member\_casual all\_trips\_v2$ride\_length  
## 1 casual 2175468 secs  
## 2 member 93594 secs

aggregate(all\_trips\_v2$ride\_length ~ all\_trips\_v2$member\_casual, FUN = min)

## all\_trips\_v2$member\_casual all\_trips\_v2$ride\_length  
## 1 casual 0 secs  
## 2 member 0 secs

# See the average ride time by each day for members vs casual users

aggregate(all\_trips\_v2$ride\_length ~ all\_trips\_v2$member\_casual + all\_trips\_v2$day\_of\_week, FUN = mean)

## all\_trips\_v2$member\_casual all\_trips\_v2$day\_of\_week all\_trips\_v2$ride\_length  
## 1 casual domingo 2184.5990 secs  
## 2 member domingo 851.0604 secs  
## 3 casual jueves 1890.0201 secs  
## 4 member jueves 744.8440 secs  
## 5 casual lunes 2083.5216 secs  
## 6 member lunes 770.5501 secs  
## 7 casual martes 1711.0121 secs  
## 8 member martes 731.4727 secs  
## 9 casual miércoles 1722.8181 secs  
## 10 member miércoles 718.5795 secs  
## 11 casual sábado 2177.7096 secs  
## 12 member sábado 859.5084 secs  
## 13 casual viernes 1850.0039 secs  
## 14 member viernes 737.3686 secs

# Notice that the days of the week are out of order. Let's fix that.  
#all\_trips\_v2$day\_of\_week <- ordered(all\_trips\_v2$day\_of\_week,   
# levels=c("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"))

# Now, let’s run the average ride time by each day for members vs casual users

##aggregate(all\_trips\_v2member\_casual + all\_trips\_v2$day\_of\_week, FUN = mean)

# Analyze ridership data by type and weekday

all\_trips\_v2 %>%   
 mutate(weekday = wday(started\_at, label = TRUE)) %>% #creates weekday field using wday()  
 group\_by(member\_casual, weekday) %>% #groups by usertype and weekday  
 summarise(number\_of\_rides = n() #calculates the number of rides and average duration   
 ,average\_duration = mean(ride\_length)) %>% # calculates the average duration  
 arrange(member\_casual, weekday) # sorts

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

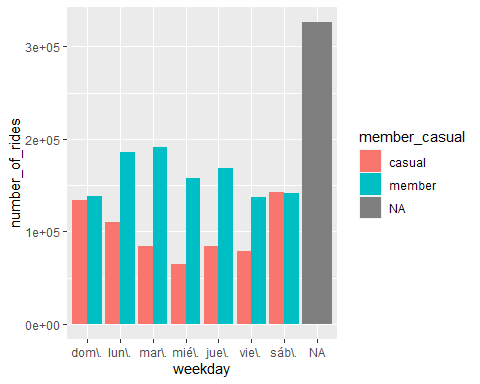
## # A tibble: 15 × 4  
## # Groups: member\_casual [3]  
## member\_casual weekday number\_of\_rides average\_duration  
## <chr> <ord> <int> <drtn>   
## 1 casual "dom\\." 133833 2184.5990 secs   
## 2 casual "lun\\." 109820 2083.5216 secs   
## 3 casual "mar\\." 83770 1711.0121 secs   
## 4 casual "mié\\." 64576 1722.8181 secs   
## 5 casual "jue\\." 83903 1890.0201 secs   
## 6 casual "vie\\." 78318 1850.0039 secs   
## 7 casual "sáb\\." 142855 2177.7096 secs   
## 8 member "dom\\." 138353 851.0604 secs   
## 9 member "lun\\." 186129 770.5501 secs   
## 10 member "mar\\." 191130 731.4727 secs   
## 11 member "mié\\." 157611 718.5795 secs   
## 12 member "jue\\." 169011 744.8440 secs   
## 13 member "vie\\." 137380 737.3686 secs   
## 14 member "sáb\\." 141312 859.5084 secs   
## 15 <NA> <NA> 326381 NA secs

#DATA VIZ

# Let’s visualize the number of rides by rider type

all\_trips\_v2 %>%   
 mutate(weekday = wday(started\_at, label = TRUE)) %>%   
 group\_by(member\_casual, weekday) %>%   
 summarise(number\_of\_rides = n()  
 ,average\_duration = mean(ride\_length)) %>%   
 arrange(member\_casual, weekday) %>%   
 ggplot(aes(x = weekday, y = number\_of\_rides, fill = member\_casual)) +  
 geom\_col(position = "dodge")

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



##getting the work environment

getwd()

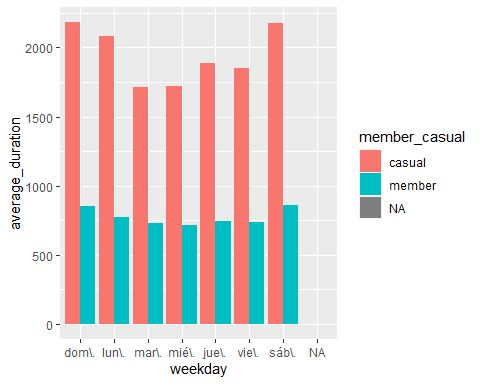
## [1] "C:/Users/Jean Yvens Alberus/Desktop/Caspstone\_BikeshareProject/CSV\_Bikeshare\_data"

# Let’s create a visualization for average duration

all\_trips\_v2 %>%   
 mutate(weekday = wday(started\_at, label = TRUE)) %>%   
 group\_by(member\_casual, weekday) %>%   
 summarise(number\_of\_rides = n()  
 ,average\_duration = mean(ride\_length)) %>%   
 arrange(member\_casual, weekday) %>%   
 ggplot(aes(x = weekday, y = average\_duration, fill = member\_casual)) +  
 geom\_col(position = "dodge")

## `summarise()` has grouped output by 'member\_casual'. You can override using the  
## `.groups` argument.  
## Don't know how to automatically pick scale for object of type difftime.  
## Defaulting to continuous.

## Warning: Removed 1 rows containing missing values (geom\_col).



# EXPORTING SUMMARY FILE FOR FURTHER ANALYSIS

#counts <- aggregate(all\_trips\_v2$ride\_length ~ all\_trips\_v2$member\_casual + all\_trips\_v2$day\_of\_week, FUN = mean)

#code to export the file avg\_ride\_length.csv write.csv(counts, file = “C:/Users/Jean Yvens Alberus/Desktop/Caspstone\_BikeshareProject/CSV\_Bikeshare\_data\avg\_ride\_length.csv”, row.names= TRUE)

write.csv(all\_trips\_v2, file= "alltrip\_bikeshare.csv")