Cyclistic Bike-Share Analysis Capstone

Rachit Malik

December 11, 2025

1 Introduction

This R Markdown documents an end-to-end analysis for the **Cyclistic Bike-Share** capstone: reading and stacking monthly CSV files, cleaning and enriching the data, computing summary statistics, and producing visualizations that compare member and casual riders.

> Make sure your CSV files are in the folder: `C:/Users/a2z/Downloads/Cyclistic` (or update `file_paths` below to the correct location).

2 1. Import and merge data

```
# List CSV files in the folder
file_paths <- list.files("C:/Users/a2z/Downloads/Cyclistic", pattern = "\\.csv$", full.names = TRUE)

# Read all files and row-bind them into a single tibble
all_trips <- file_paths %>%
  map_dfr(read_csv)

# Quick peek
glimpse(all_trips)
```

```
## Rows: 5,667,717
## Columns: 13
## $ ride_id            <chr> "C2F7DD78E82EC875", "A6CF8980A652D272", "BD0F91DFF7…
## $ rideable_type      <chr> "electric_bike", "electric_bike", "classic_bike", "…
## $ started_at         <dttm> 2022-01-13 11:59:47, 2022-01-10 08:41:56, 2022-01-…
## $ ended_at           <dttm> 2022-01-13 12:02:44, 2022-01-10 08:46:17, 2022-01-…
## $ start_station_name <chr> "Glenwood Ave & Touhy Ave", "Glenwood Ave & Touhy A…
## $ start_station_id   <chr> "525", "525", "TA1306000016", "KA1504000151", "TA13…
## $ end_station_name   <chr> "Clark St & Touhy Ave", "Clark St & Touhy Ave", "Gr…
## $ end_station_id     <chr> "RP-007", "RP-007", "TA1307000001", "TA1309000021",…
## $ start_lat          <dbl> 42.01280, 42.01276, 41.92560, 41.98359, 41.87785, 4…
## $ start_lng          <dbl> -87.66591, -87.66597, -87.65371, -87.66915, -87.624…
## $ end_lat            <dbl> 42.01256, 42.01256, 41.92533, 41.96151, 41.88462, 4…
## $ end_lng            <dbl> -87.67437, -87.67437, -87.66580, -87.67139, -87.627…
## $ member_casual      <chr> "casual", "casual", "member", "casual", "member", "…
```

```
head(all_trips)
```

```
## # A tibble: 6 × 13
##   ride_id          rideable_type started_at          ended_at
##   <chr>            <chr>         <dttm>              <dttm>
## 1 C2F7DD78E82EC875 electric_bike 2022-01-13 11:59:47 2022-01-13 12:02:44
## 2 A6CF8980A652D272 electric_bike 2022-01-10 08:41:56 2022-01-10 08:46:17
## 3 BD0F91DFF741C66D classic_bike  2022-01-25 04:53:40 2022-01-25 04:58:01
## 4 CBB80ED419105406 classic_bike  2022-01-04 00:18:04 2022-01-04 00:33:00
## 5 DDC963BFDDA51EEA classic_bike  2022-01-20 01:31:10 2022-01-20 01:37:12
## 6 A39C6F6CC0586C0B classic_bike  2022-01-11 18:48:09 2022-01-11 18:51:31
## # ℹ 9 more variables: start_station_name <chr>, 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>
```

3 2. Initial checks

```
# If you have a large dataset, use skim to get a compact summary
skim_without_charts(all_trips)
```

Data summary

Name	all_trips
Number of rows	5667717
Number of columns	13
_______________________	
Column type frequency:	
character	7
numeric	4
POSIXct	2
________________________	
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
ride_id	0	1.00	16	16	0	5667717	0
rideable_type	0	1.00	11	13	0	3	0
start_station_name	833064	0.85	7	64	0	1674	0
start_station_id	833064	0.85	3	44	0	1313	0
end_station_name	892742	0.84	9	64	0	1692	0
end_station_id	892742	0.84	3	44	0	1317	0
member_casual	0	1.00	6	6	0	2	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100
start_lat	0	1	41.90	0.05	41.64	41.88	41.90	41.93	45.64
start_lng	0	1	-87.65	0.03	-87.84	-87.66	-87.64	-87.63	-73.80
end_lat	5858	1	41.90	0.07	0.00	41.88	41.90	41.93	42.37
end_lng	5858	1	-87.65	0.11	-88.14	-87.66	-87.64	-87.63	0.00

Variable type: POSIXct

skim_variable	n_missing	complete_rate	min	max	median	n_unique
started_at	0	1	2022-01-01 00:00:05	2022-12-31 23:59:26	2022-07-22 15:03:59	4745862
ended_at	0	1	2022-01-01 00:01:48	2023-01-02 04:56:45	2022-07-22 15:24:44	4758633

```
# Check for obvious issues: missing columns that we expect
expected_cols <- c("ride_id", "started_at", "ended_at", "start_station_name", "member_casual")
setdiff(expected_cols, colnames(all_trips))
```

```
## character(0)
```

4 3. Clean column names and drop empty / duplicate rows

```
all_trips_v2 <- all_trips %>%
  clean_names() %>%                  # standardise column names to snake_case
  drop_na() %>%                      # remove rows with any NA values (use with caution)
  distinct(ride_id, .keep_all = TRUE) # remove duplicate rides if present

# confirm
glimpse(all_trips_v2)
```

```
## Rows: 4,369,360
## Columns: 13
## $ ride_id            <chr> "C2F7DD78E82EC875", "A6CF8980A652D272", "BD0F91DFF7…
## $ rideable_type      <chr> "electric_bike", "electric_bike", "classic_bike", "…
## $ started_at         <dttm> 2022-01-13 11:59:47, 2022-01-10 08:41:56, 2022-01-…
## $ ended_at           <dttm> 2022-01-13 12:02:44, 2022-01-10 08:46:17, 2022-01-…
## $ start_station_name <chr> "Glenwood Ave & Touhy Ave", "Glenwood Ave & Touhy A…
## $ start_station_id   <chr> "525", "525", "TA1306000016", "KA1504000151", "TA13…
## $ end_station_name   <chr> "Clark St & Touhy Ave", "Clark St & Touhy Ave", "Gr…
## $ end_station_id     <chr> "RP-007", "RP-007", "TA1307000001", "TA1309000021",…
## $ start_lat          <dbl> 42.01280, 42.01276, 41.92560, 41.98359, 41.87785, 4…
## $ start_lng          <dbl> -87.66591, -87.66597, -87.65371, -87.66915, -87.624…
## $ end_lat            <dbl> 42.01256, 42.01256, 41.92533, 41.96151, 41.88462, 4…
## $ end_lng            <dbl> -87.67437, -87.67437, -87.66580, -87.67139, -87.627…
## $ member_casual      <chr> "casual", "casual", "member", "casual", "member", "…
```

> **Note:** `drop_na()` removes *any* row with an NA. If you'd prefer to only remove rows missing key columns (e.g., `started_at` or `ended_at`), replace `drop_na()` with `drop_na(started_at, ended_at, ride_id)`.

5 4. Add calculated columns

```
all_trips_v2 <- all_trips_v2 %>%
  mutate(
    # calculate ride length in minutes (ensure started_at / ended_at are POSIXct)
    started_at = as_datetime(started_at),
    ended_at   = as_datetime(ended_at),
    ride_length = as.numeric(difftime(ended_at, started_at, units = "mins")),

    # extract day, month, year for aggregation
    day_of_week = wday(started_at, label = TRUE, abbr = FALSE),
    month = month(started_at, label = TRUE, abbr = FALSE),
    year = year(started_at)
  )

# quick summary of ride_length
summary(all_trips_v2$ride_length)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##  -168.70     6.05    10.60    17.09    19.02 34354.07
```

6 5. Filter out invalid or administrative rides

```
all_trips_clean <- all_trips_v2 %>%
  filter(ride_length > 1,            # remove false starts
         ride_length < 1440,         # remove rides > 24 hours
         start_station_name != "HQ QR") # remove test/admin rides

# confirm cleaned row counts
nrow(all_trips_v2)
```

```
## [1] 4369360
```

```
nrow(all_trips_clean)
```

```
## [1] 4291805
```

7 6. Summary statistics by user type

```
summary_stats <- all_trips_clean %>%
  group_by(member_casual) %>%
  summarise(
    average_duration = mean(ride_length, na.rm = TRUE),
    median_duration = median(ride_length, na.rm = TRUE),
    max_duration = max(ride_length, na.rm = TRUE),
    total_rides = n(),
    .groups = "drop"
  )

print(summary_stats)
```

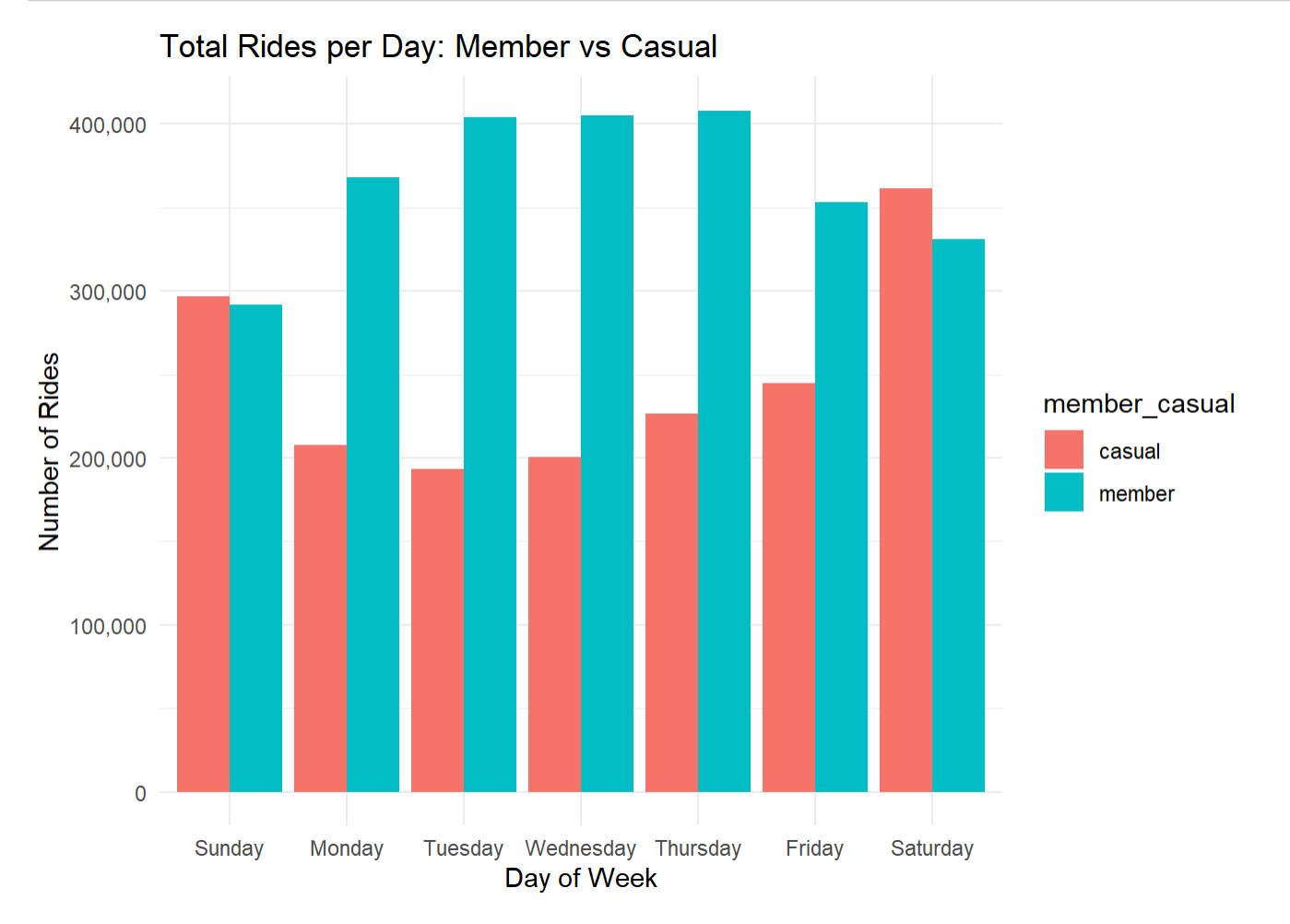
```
## # A tibble: 2 × 5
##   member_casual average_duration median_duration max_duration total_rides
##   <chr>                    <dbl>           <dbl>        <dbl>       <int>
## 1 casual                    24.1            14.1        1439.     1730819
## 2 member                    12.7             9.15       1436.     2560986
```

8 7. Daily usage patterns (member vs casual)

```
daily_usage <- all_trips_clean %>%
  group_by(member_casual, day_of_week) %>%
  summarise(
    number_of_rides = n(),
    average_duration = mean(ride_length, na.rm = TRUE),
    .groups = "drop"
  ) %>%
  arrange(member_casual, day_of_week)

print(daily_usage)
```

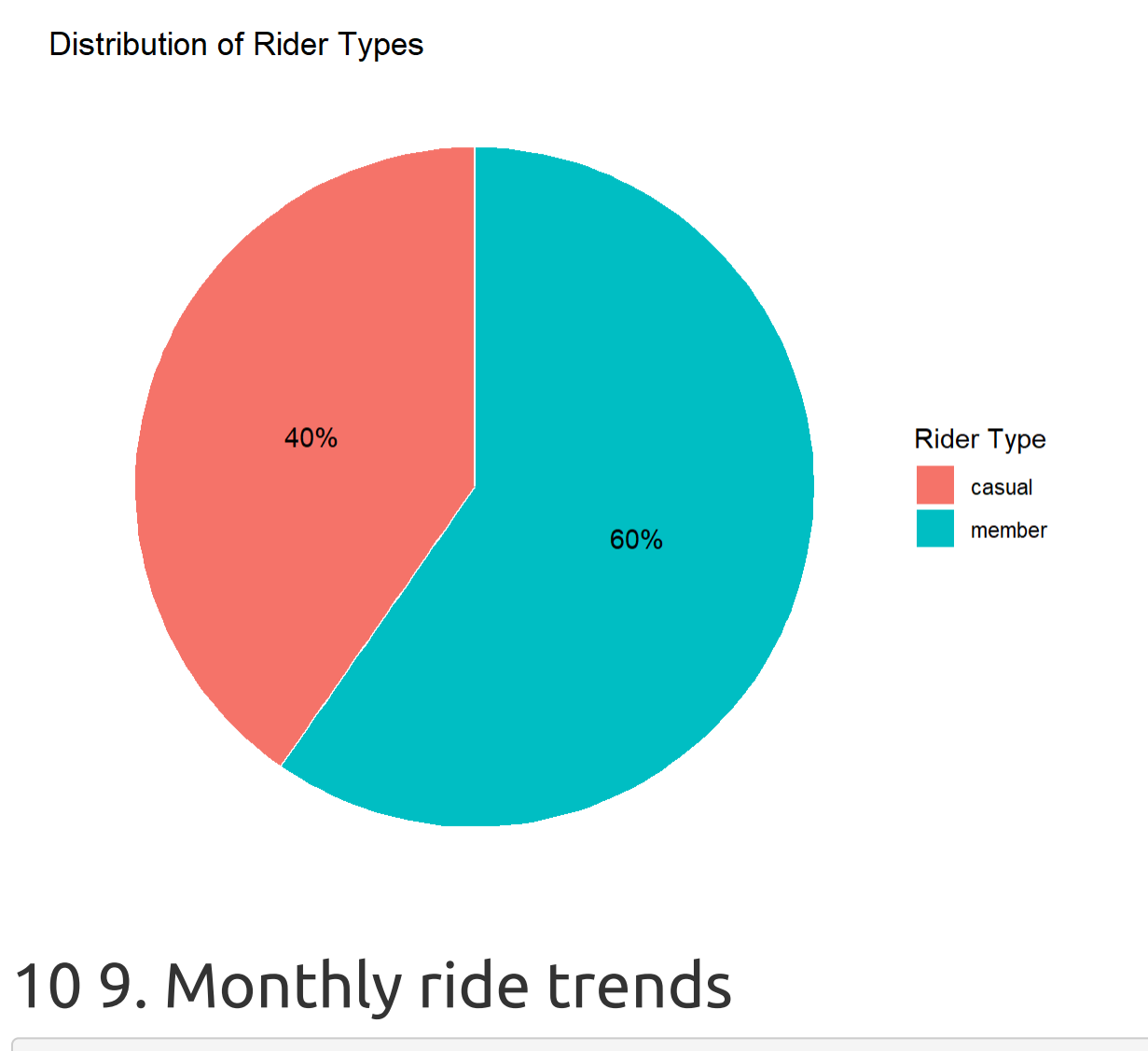
```
## # A tibble: 14 × 4
##    member_casual day_of_week number_of_rides average_duration
##    <chr>         <ord>                 <int>            <dbl>
##  1 casual        Sunday               296510             27.4
##  2 casual        Monday               207483             24.7
##  3 casual        Tuesday              193370             21.6
##  4 casual        Wednesday            200483             20.8
##  5 casual        Thursday             226515             21.4
##  6 casual        Friday               244931             22.5
##  7 casual        Saturday             361527             26.9
##  8 member        Sunday               291582             14.1
##  9 member        Monday               368169             12.2
## 10 member        Tuesday              403772             12.0
## 11 member        Wednesday            405056             12.1
## 12 member        Thursday             408121             12.2
## 13 member        Friday               353028             12.5
## 14 member        Saturday             331258             14.3
```

```
# bar chart comparing days
ggplot(data = daily_usage) +
  aes(x = day_of_week, y = number_of_rides, fill = member_casual) +
  geom_col(position = "dodge") +
  labs(title = "Total Rides per Day: Member vs Casual",
       y = "Number of Rides", x = "Day of Week") +
  scale_y_continuous(labels = scales::comma) +
  theme_minimal()
```



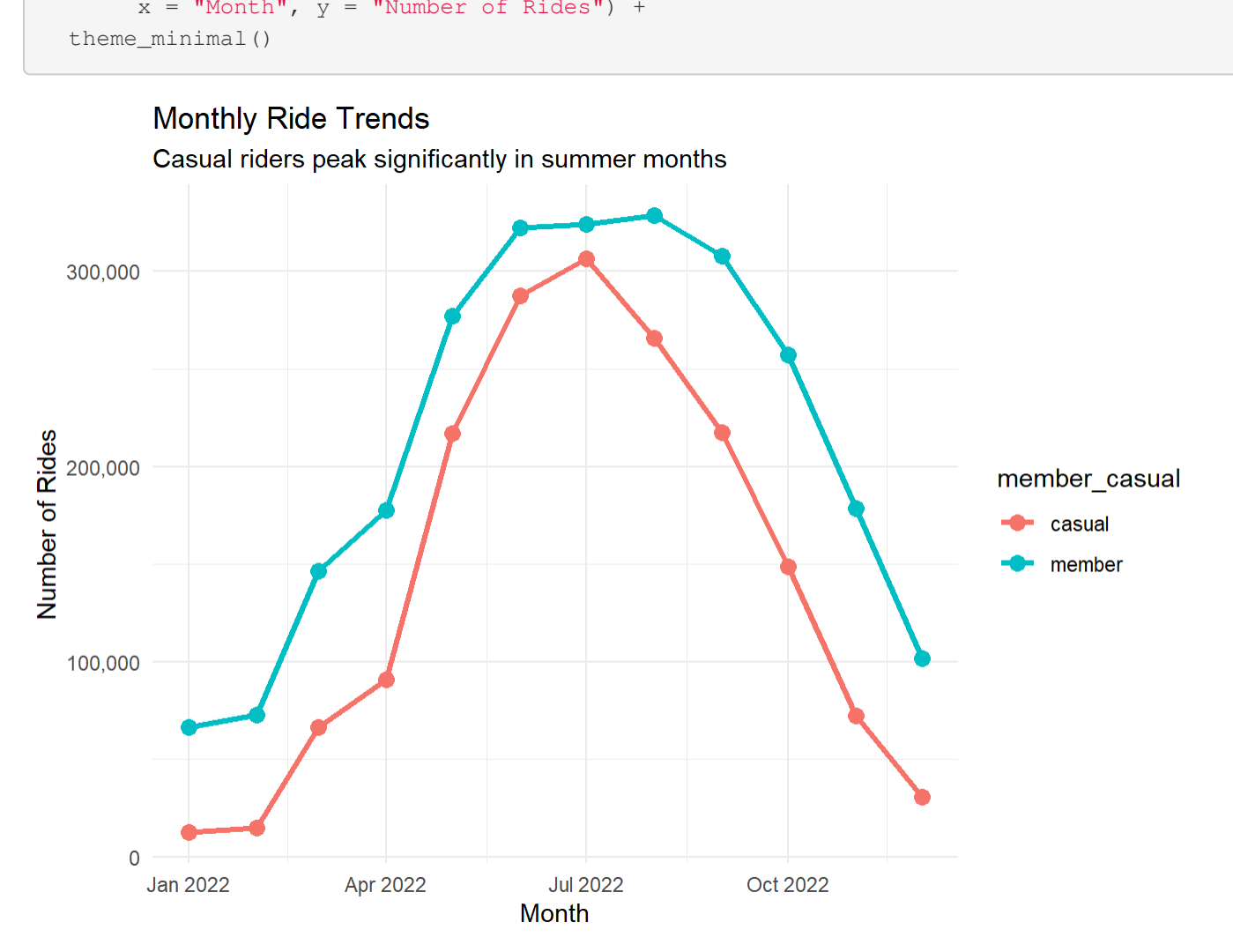
9 8. Rider type distribution (pie chart)

```
user_dist <- all_trips_clean %>%
  count(member_casual) %>%
  mutate(perc = n / sum(n), labels = scales::percent(perc))

# pie chart
ggplot(user_dist, aes(x = "", y = perc, fill = member_casual)) +
  geom_col(color = "white") +
  coord_polar(theta = "y") +
  geom_text(aes(label = labels), position = position_stack(vjust = 0.5)) +
  theme_void() +
  labs(title = "Distribution of Rider Types", fill = "Rider Type")
```



10 9. Monthly ride trends

```
monthly_trend <- all_trips_clean %>%
  mutate(month = floor_date(started_at, "month")) %>%
  group_by(month, member_casual) %>%
  summarise(number_of_rides = n(), .groups = "drop")

# line plot
ggplot(monthly_trend, aes(x = month, y = number_of_rides, color = member_casual)) +
  geom_line(size = 1.2) +
  geom_point(size = 3) +
  scale_y_continuous(labels = scales::comma) +
  labs(title = "Monthly Ride Trends",
       subtitle = "Casual riders peak significantly in summer months",
       x = "Month", y = "Number of Rides") +
  theme_minimal()
```



11 10. Export cleaned data

```
write_csv(all_trips_clean, "C:/Users/a2z/Downloads/cyclistic_final_cleaned_data.csv")
```

12 11. Suggested next steps

- Add checks for timezone inconsistencies and ensure `started_at` / `ended_at` parse correctly.
- Consider creating `ride_length_minutes` categories (e.g., short: <10, medium: 10-60, long: >60) for segmentation.
- Compare trip distances (if available) vs ride duration to detect e-bikes or other anomalies.
- Add interactive visualizations using `plotly` or build a small Shiny app to explore segments.

13 Session info

```
## R version 4.5.2 (2025-10-31 ucrt)
## Platform: x86_64-w64-mingw32/x64
## Running under: Windows 11 x64 (build 26200)
##
## Matrix products: default
##   LAPACK version 3.12.1
##
## locale:
## [1] LC_COLLATE=English_United States.utf8
## [2] LC_CTYPE=English_United States.utf8
## [3] LC_MONETARY=English_United States.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
## time zone: Asia/Calcutta
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] scales_1.4.0    janitor_2.2.1   skimr_2.2.1     lubridate_1.9.4
##  [5] forcats_1.0.1   stringr_1.6.0   dplyr_1.1.4     purrr_1.2.0
##  [9] readr_2.1.6     tidyr_1.3.1     tibble_3.3.0    ggplot2_4.0.1
## [13] tidyverse_2.0.0
##
## loaded via a namespace (and not attached):
##  [1] utf8_1.2.6         sass_0.4.10        generics_0.1.4     stringi_1.8.7
##  [5] hms_1.1.4          digest_0.6.38      magrittr_2.0.4     evaluate_1.0.5
##  [9] grid_4.5.2         timechange_0.3.0   RColorBrewer_1.1-3 fastmap_1.2.0
## [13] jsonlite_2.0.0     jquerylib_0.1.4    cli_3.6.5          rlang_1.1.6
## [17] crayon_1.5.3       bit64_4.6.0-1      base64enc_0.1-3    withr_3.0.2
## [21] repr_1.1.7         cachem_1.1.0       yaml_2.3.10        tools_4.5.2
## [25] parallel_4.5.2     tzdb_0.5.0         vctrs_0.6.5        R6_2.6.1
## [29] lifecycle_1.0.4    snakecase_0.11.1   bit_4.6.0          vroom_1.6.6
## [33] pkgconfig_2.0.3    pillar_1.11.1      bslib_0.9.0        gtable_0.3.6
## [37] glue_1.8.0         haven_2.5.5        xfun_0.54          tidyselect_1.2.1
## [41] rstudioapi_0.17.1  knitr_1.50         farver_2.1.2       htmltools_0.5.8.1
## [45] labeling_0.4.3     rmarkdown_2.30     compiler_4.5.2     S7_0.2.1
```