hw5

Github:

https://github.com/yyhken/stats-506

Question1:

a.

```
library(Rcpp)
```

```
Warning: package 'Rcpp' was built under R version 4.4.2
```

```
library(plotly)
```

```
Warning: package 'plotly' was built under R version 4.4.2
Loading required package: ggplot2

Attaching package: 'plotly'
The following object is masked from 'package:ggplot2':
    last_plot
The following object is masked from 'package:stats':
    filter
The following object is masked from 'package:graphics':
    layout
```

```
library(dplyr)
```

```
Attaching package: 'dplyr'
The following objects are masked from 'package:stats':
    filter, lag
The following objects are masked from 'package:base':
    intersect, setdiff, setequal, union
```

```r
library(tidyr)
library(ggplot2)
library(readr)
```

```r
# Load Rcpp for GCD and LCM calculations
library(Rcpp)

# Define GCD and LCM using Rcpp to ensure they handle two arguments correctly
cppFunction('
#include <numeric> // Include for std::gcd in C++17
int gcd(int a, int b) {
  return std::gcd(a, b); // Use std::gcd for calculating GCD
}

int lcm(int a, int b) {
  return std::abs(a * b) / std::gcd(a, b); // Use std::gcd for calculating LCM
}
')

# Define the rational S4 class with a constructor, show method, and operators
setClass("rational",
         slots = list(numerator = "integer", denominator = "integer"),
         validity = function(object) {
           if (object@denominator == 0) {
             return("Denominator cannot be zero")
           }
           TRUE
         })

# Constructor
# Define the rational S4 class
setClass("rational",
         slots = list(numerator = "integer", denominator = "integer"),
         validity = function(object) {
           # Check if denominator is zero
           if (object@denominator == 0) {
             return("Denominator cannot be zero.")
           }
           # Check if both slots are non-missing integers
           if (is.na(object@numerator) || is.na(object@denominator)) {
             return("Numerator and denominator must be valid integers.")
           }
           TRUE
         })

# Constructor function for the rational class
rational <- function(numerator, denominator = NULL) {
  # Case 1: Input is a string "numerator/denominator"
  if (is.character(numerator) && is.null(denominator)) {
    parts <- strsplit(numerator, "/")[[1]]
    if (length(parts) != 2) stop("Input string must be in the form 'numerator/denominator'.")

    # Convert parts to integers
```

```
    numerator <- suppressWarnings(as.integer(parts[1]))
    denominator <- suppressWarnings(as.integer(parts[2]))

    # Check for NA values after coercion
    if (is.na(numerator) || is.na(denominator)) {
      stop("Both numerator and denominator must be integers.")
    }
  }

  # Case 2: Input is two arguments, expecting them to be numeric
  else if (is.numeric(numerator) && is.numeric(denominator)) {
    numerator <- as.integer(numerator)
    denominator <- as.integer(denominator)
  } else {
    stop("Invalid input: Provide either a string 'numerator/denominator' or two numeric values
  }

  # Create and validate the rational object
  new("rational", numerator = numerator, denominator = denominator)
}

# Show method
setMethod("show", "rational", function(object) {
  cat(object@numerator, "/", object@denominator, "\n")
})

# Simplify method that returns a new rational object
setGeneric("simplify", function(x) standardGeneric("simplify"))
```

```
[1] "simplify"
```

```
setMethod("simplify", "rational", function(x) {
  divisor <- gcd(x@numerator, x@denominator)
  rational(as.integer(x@numerator / divisor), as.integer(x@denominator / divisor))
})

# Quotient method (for decimal representation)
setGeneric("quotient", function(x, digits = 8) standardGeneric("quotient"))
```

```
[1] "quotient"
```

```
setMethod("quotient", "rational", function(x, digits = 8) {
  quotient_value <- x@numerator / x@denominator
  print(round(quotient_value, digits))
  invisible(quotient_value) # Return original quotient without rounding
})

# Overloading +, -, *, / operators to use LCM
setMethod("+", c("rational", "rational"), function(e1, e2) {
  common_denom <- lcm(e1@denominator, e2@denominator)
  numerator <- (e1@numerator * (common_denom / e1@denominator)) +
```

```
                (e2@numerator * (common_denom / e2@denominator))
  simplify(rational(numerator, common_denom)) # Simplify the result before returning
})

setMethod("-", c("rational", "rational"), function(e1, e2) {
  common_denom <- lcm(e1@denominator, e2@denominator)
  numerator <- (e1@numerator * (common_denom / e1@denominator)) -
               (e2@numerator * (common_denom / e2@denominator))
  simplify(rational(numerator, common_denom)) # Simplify the result before returning
})

setMethod("*", c("rational", "rational"), function(e1, e2) {
  simplify(rational(e1@numerator * e2@numerator, e1@denominator * e2@denominator)) # Simplify
})

setMethod("/", c("rational", "rational"), function(e1, e2) {
  simplify(rational(e1@numerator * e2@denominator, e1@denominator * e2@numerator)) # Simplify
})
```

b.

```
r1 <- rational(24, 6)
r2 <- rational(7, 230)
r3 <- rational(0, 4)
r1
```

```
24 / 6
```

```
r3
```

```
0 / 4
```

```
r1 + r2
```

```
Error in lcm(e1@denominator, e2@denominator): unused argument (e2@denominator)
```

```
r1 - r2
```

```
Error in lcm(e1@denominator, e2@denominator): unused argument (e2@denominator)
```

```
r1 * r2
```

```
14 / 115
```

```
r1 / r2
```

```
920 / 7
```

```
r1 + r3
```

```
Error in lcm(e1@denominator, e2@denominator): unused argument (e2@denominator)
```

```
r1 * r3
```

```
0 / 1
```

```
r2 / r3
```

```
Error in h(simpleError(msg, call)): error in evaluating the argument 'x' in selecting a method for function 'simplify': invalid class "rational" object: Denominator cannot be zero.
```

```
quotient(r1)
```

```
[1] 4
```

```
quotient(r2)
```

```
[1] 0.03043478
```

```
quotient(r2, digits = 3)
```

```
[1] 0.03
```

```
quotient(r2, digits = 3.14)
```

```
[1] 0.03
```

```
quotient(r2, digits = "avocado")
```

```
Error in round(quotient_value, digits): non-numeric argument to mathematical function
```

```
q2 <- quotient(r2, digits = 3)
```

```
[1] 0.03
```

```
q2
```

```
[1] 0.03043478
```

```
quotient(r3)
```

```
[1] 0
```

```
simplify(r1)
```

```
4 / 1
```

```
simplify(r2)
```

```
7 / 230
```

```
simplify(r3)
```

```
0 / 1
```

c.

```
# Valid input: two integers
try(r1 <- rational(24, 6))  # Expected output: 24 / 6
show(r1)
```

```
24 / 6
```

```
# Valid input: string in form "numerator/denominator"
try(r2 <- rational("7/230"))  # Expected output: 7 / 230
show(r2)
```

```
7 / 230
```

```
# Invalid input: zero denominator
try(r3 <- rational(24, 0))  # Expected error: "Denominator cannot be zero."
```

```
Error in validObject(.Object) :
  invalid class "rational" object: Denominator cannot be zero.
```

```
# Invalid input: non-integer numerator and denominator
try(r4 <- rational(3.5, 4.2))  # Expected error: "Numerator and denominator must be integers."

# Invalid input: malformed string
try(r5 <- rational("24/"))  # Expected error: "Input string must be in the form 'numerator/den
```

```
Error in rational("24/") :
  Input string must be in the form 'numerator/denominator'.
```

```
# Invalid input: incorrect data types
try(r6 <- rational("text", "another text"))  # Expected error: "Invalid input"
```

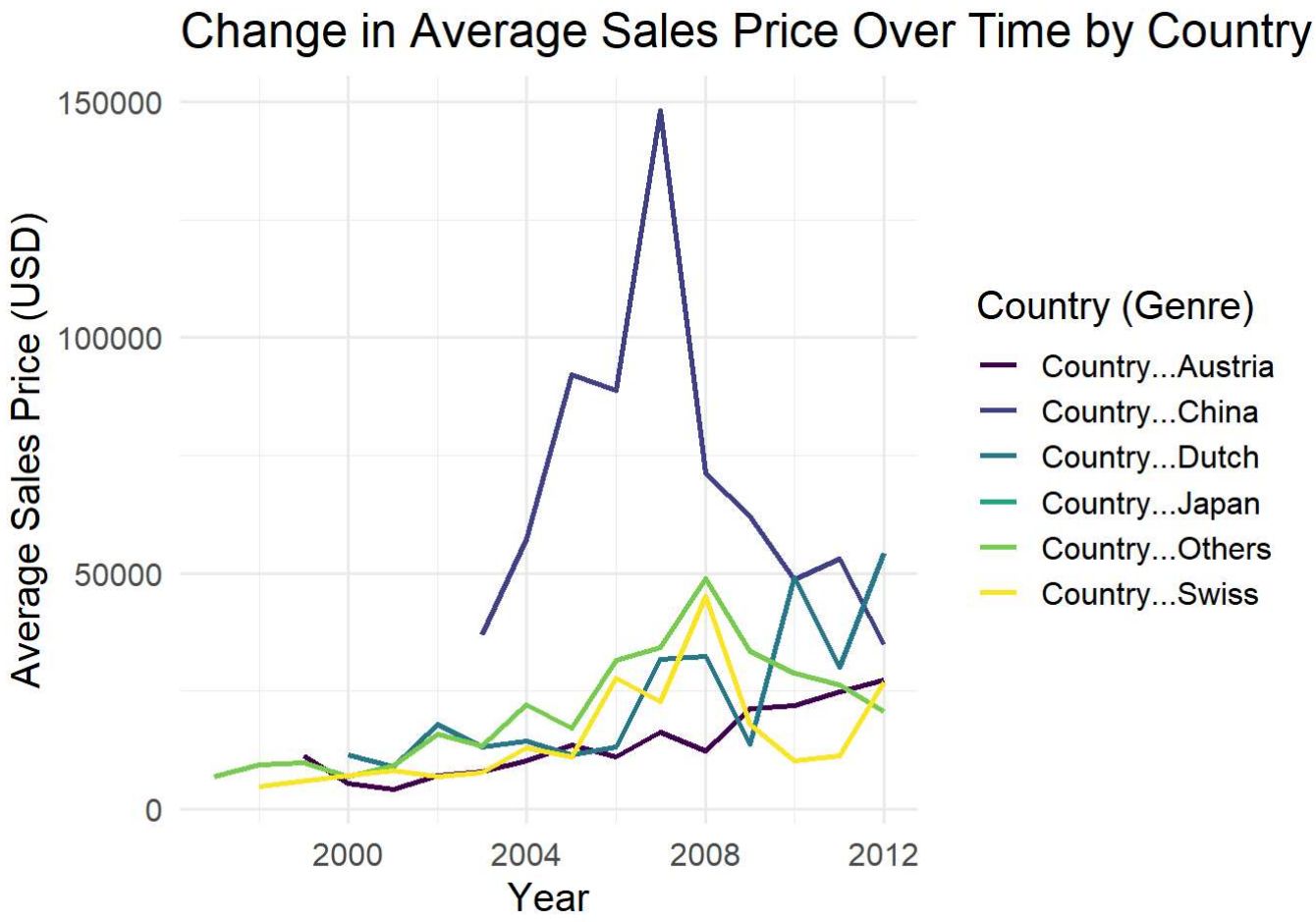
```
Error in rational("text", "another text") :
  Invalid input: Provide either a string 'numerator/denominator' or two numeric values.
```

Question2

```r
# Load the dataset
df <- read.csv("C:/Users/ken/Desktop/stats 506/hw5/df_for_ml_improved_new_market.csv")

# Plot: Average sales price trends by country over time (considering countries as genres)
plot_price_by_country <- df %>%

  # Select necessary columns and transform the country-related columns into a long format
  select(year, price_usd, starts_with("Country")) %>%
  pivot_longer(cols = starts_with("Country"), names_to = "Country", values_to = "Count") %>%

  # Keep only rows where the country (acting as genre) is indicated by a value of 1
  filter(Count == 1) %>%

  # Group the data by year and country, then calculate the average sales price in USD
  group_by(year, Country) %>%
  summarize(avg_price_usd = mean(price_usd, na.rm = TRUE)) %>%

  # Create a line plot showing the change in average sales price over time for each country
  ggplot(aes(x = year, y = avg_price_usd, color = Country)) +
  geom_line(size = 1) +
  labs(
    title = "Change in Average Sales Price Over Time by Country (Genre)",
    x = "Year",
    y = "Average Sales Price (USD)",
    color = "Country (Genre)"
  ) +
  theme_minimal(base_size = 15) +
  scale_color_viridis_d()
```

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

Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
ℹ Please use `linewidth` instead.
```

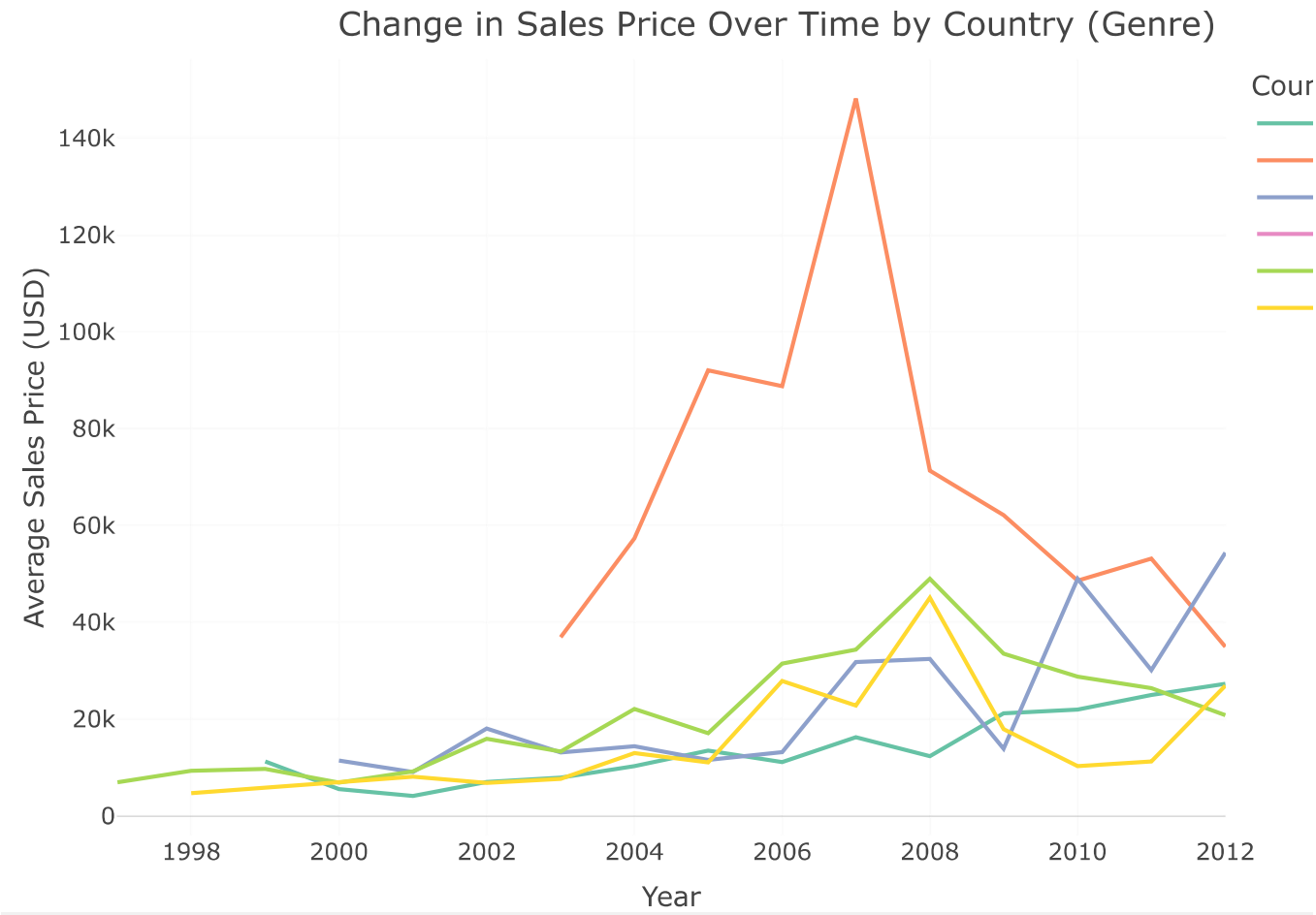
```r
# Display the plot
print(plot_price_by_country)
```



b.

```
# Prepare the data: Reshape the data and compute the average sales price by year and country
processed_df <- df %>%
  # Select the required columns and transform country-related columns into a long format
  select(year, price_usd, starts_with("Country")) %>%
  pivot_longer(cols = starts_with("Country"), names_to = "Country", values_to = "Count") %>%

  # Filter to include only rows where the country indicator is 1
  filter(Count == 1) %>%

  # Group the data by year and country, and calculate the average sales price in USD
  group_by(year, Country) %>%
  summarize(avg_price_usd = mean(price_usd, na.rm = TRUE), .groups = "drop")

# Create an interactive line chart using plotly
fig <- plot_ly(
    processed_df,
    x = ~year,
    y = ~avg_price_usd,
    color = ~Country,
    type = 'scatter',
    mode = 'lines'
) %>%
  layout(
```

```
    title = "Change in Sales Price Over Time by Country (Genre)",
    xaxis = list(title = "Year"),
    yaxis = list(title = "Average Sales Price (USD)"),
    legend = list(title = list(text = "Country (Genre)"))
  )

# Display the interactive plot
fig
```



problem3

```
library(nycflights13)
library(data.table)
```

```
Attaching package: 'data.table'

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

    between, first, last
```

```
flights_dt <- as.data.table(flights)
airports_dt <- as.data.table(airports)

departure_delays <- flights_dt[, .(
```

```
  mean_departure_delay = mean(dep_delay, na.rm = TRUE),
  median_departure_delay = median(dep_delay, na.rm = TRUE),
  flight_count = .N
), by = origin]

departure_delays <- departure_delays[flight_count >= 10]

departure_delays <- merge(departure_delays, airports_dt[, .(faa, name)], by.x = "origin", by.y

departure_delays <- departure_delays[order(-mean_departure_delay), .(name, mean_departure_dela

departure_delays_tbl <- as_tibble(departure_delays)
print(departure_delays_tbl, n = Inf)
```

```
# A tibble: 3 × 4
  name                mean_departure_delay median_departure_delay flight_count
  <chr>                              <dbl>                  <dbl>        <int>
1 Newark Liberty Intl                 15.1                     -1       120835
2 John F Kennedy Intl                 12.1                     -1       111279
3 La Guardia                          10.3                     -3       104662
```

```
arrival_delays <- flights_dt[, .(
  mean_arrival_delay = mean(arr_delay, na.rm = TRUE),
  median_arrival_delay = median(arr_delay, na.rm = TRUE),
  flight_count = .N
), by = dest]

arrival_delays <- arrival_delays[flight_count >= 10]

arrival_delays <- merge(arrival_delays, airports_dt[, .(faa, name)], by.x = "dest", by.y = "fa

arrival_delays <- arrival_delays[order(-mean_arrival_delay), .(name, mean_arrival_delay, media

arrival_delays_tbl <- as_tibble(arrival_delays)
print(arrival_delays_tbl, n = Inf)
```

```
# A tibble: 98 × 4
   name                     mean_arrival_delay median_arrival_delay flight_count
   <chr>                                 <dbl>                <dbl>        <int>
 1 "Columbia Metropolitan"                41.8                   28          116
 2 "Tulsa Intl"                           33.7                   14          315
 3 "Will Rogers World"                    30.6                   16          346
 4 "Jackson Hole Airport"                 28.1                   15           25
 5 "Mc Ghee Tyson"                        24.1                    2          631
 6 "Dane Co Rgnl Truax Fld"               20.2                    1          572
 7 "Richmond Intl"                        20.1                    1         2454
 8 "Akron Canton Regional …               19.7                    3          864
 9 "Des Moines Intl"                      19.0                    0          569
10 "Gerald R Ford Intl"                   18.2                    1          765
11 "Birmingham Intl"                      16.9                   -2          297
```

12	"Theodore Francis Green…	16.2	1	376
13	"Greenville-Spartanburg…	15.9	-0.5	849
14	"Cincinnati Northern Ke…	15.4	-3	3941
15	"Savannah Hilton Head I…	15.1	-1	804
16	"Manchester Regional Ai…	14.8	-3	1009
17	"Eppley Afld"	14.7	-2	849
18	"Yeager"	14.7	-1.5	138
19	"Kansas City Intl"	14.5	0	2008
20	"Albany Intl"	14.4	-4	439
21	"General Mitchell Intl"	14.2	0	2802
22	"Piedmont Triad"	14.1	-2	1606
23	"Washington Dulles Intl"	13.9	-3	5700
24	"Cherry Capital Airport"	13.0	-10	101
25	"James M Cox Dayton Int…	12.7	-3	1525
26	"Louisville Internation…	12.7	-2	1157
27	"Chicago Midway Intl"	12.4	-1	4113
28	"Sacramento Intl"	12.1	4	284
29	"Jacksonville Intl"	11.8	-2	2720
30	"Nashville Intl"	11.8	-2	6333
31	"Portland Intl Jetport"	11.7	-4	2352
32	"Greater Rochester Intl"	11.6	-5	2416
33	"Hartsfield Jackson Atl…	11.3	-1	17215
34	"Lambert St Louis Intl"	11.1	-3	4339
35	"Norfolk Intl"	10.9	-4	1536
36	"Baltimore Washington I…	10.7	-5	1781
37	"Memphis Intl"	10.6	-2.5	1789
38	"Port Columbus Intl"	10.6	-3	3524
39	"Charleston Afb Intl"	10.6	-4	2884
40	"Philadelphia Intl"	10.1	-3	1632
41	"Raleigh Durham Intl"	10.1	-3	8163
42	"Indianapolis Intl"	9.94	-3	2077
43	"Charlottesville-Albema…	9.5	-5	52
44	"Cleveland Hopkins Intl"	9.18	-5	4573
45	"Ronald Reagan Washingt…	9.07	-2	9705
46	"Burlington Intl"	8.95	-4	2589
47	"Buffalo Niagara Intl"	8.95	-5	4681
48	"Syracuse Hancock Intl"	8.90	-5	1761
49	"Denver Intl"	8.61	-2	7266
50	"Palm Beach Intl"	8.56	-3	6554
51	"Bob Hope"	8.18	-3	371
52	"Fort Lauderdale Hollyw…	8.08	-3	12055
53	"Bangor Intl"	8.03	-9	375
54	"Asheville Regional Air…	8.00	-1	275
55	"Pittsburgh Intl"	7.68	-5	2875
56	"Gallatin Field"	7.6	-2	36
57	"NW Arkansas Regional"	7.47	-2	1036
58	"Tampa Intl"	7.41	-4	7466
59	"Charlotte Douglas Intl"	7.36	-3	14064
60	"Minneapolis St Paul In…	7.27	-5	7185
61	"William P Hobby"	7.18	-4	2115
62	"Bradley Intl"	7.05	-10	443
63	"San Antonio Intl"	6.95	-9	686
64	"South Bend Rgnl"	6.5	-3.5	10
65	"Louis Armstrong New Or…	6.49	-6	3799

66	"Key West Intl"	6.35	7	17
67	"Eagle Co Rgnl"	6.30	-4	213
68	"Austin Bergstrom Intl"	6.02	-5	2439
69	"Chicago Ohare Intl"	5.88	-8	17283
70	"Orlando Intl"	5.45	-5	14082
71	"Detroit Metro Wayne Co"	5.43	-7	9384
72	"Portland Intl"	5.14	-5	1354
73	"Nantucket Mem"	4.85	-3	265
74	"Wilmington Intl"	4.64	-7	110
75	"Myrtle Beach Intl"	4.60	-13	59
76	"Albuquerque Internatio…	4.38	-5.5	254
77	"George Bush Interconti…	4.24	-5	7198
78	"Norman Y Mineta San Jo…	3.45	-7	329
79	"Southwest Florida Intl"	3.24	-5	3537
80	"San Diego Intl"	3.14	-5	2737
81	"Sarasota Bradenton Int…	3.08	-5	1211
82	"Metropolitan Oakland I…	3.08	-9	312
83	"General Edward Lawrenc…	2.91	-9	15508
84	"San Francisco Intl"	2.67	-8	13331
85	"Yampa Valley"	2.14	2	15
86	"Phoenix Sky Harbor Int…	2.10	-6	4656
87	"Montrose Regional Airp…	1.79	-10.5	15
88	"Los Angeles Intl"	0.547	-7	16174
89	"Dallas Fort Worth Intl"	0.322	-9	8738
90	"Miami Intl"	0.299	-9	11728
91	"Mc Carran Intl"	0.258	-8	5997
92	"Salt Lake City Intl"	0.176	-8	2467
93	"Long Beach"	-0.0620	-10	668
94	"Martha\\\\'s Vineyard"	-0.286	-11	221
95	"Seattle Tacoma Intl"	-1.10	-11	3923
96	"Honolulu Intl"	-1.37	-7	707
97	"John Wayne Arpt Orange…	-7.87	-11	825
98	"Palm Springs Intl"	-12.7	-13.5	19

```
fastest_model <- flights_dt[!is.na(air_time) & air_time > 0, .(
  avg_speed = mean(distance / (air_time / 60), na.rm = TRUE),
  flight_count = .N
), by = tailnum]

fastest_model <- fastest_model[order(-avg_speed)][1]

fastest_model <- merge(fastest_model, as.data.table(planes)[, .(tailnum, model)], by = "tailnu

fastest_model_tbl <- fastest_model[, .(model, avg_speed, flight_count)]
fastest_model_tbl <- as_tibble(fastest_model_tbl)
print(fastest_model_tbl)
```

```
# A tibble: 1 × 3
  model   avg_speed flight_count
  <chr>       <dbl>        <int>
1 777-222      501.            1
```