STATS506-Problem Set 4

Shenyi Tang

2024-10-27

Shenyi Tang's GitHub Repo For STATS 506 FA 2024

https://github.com/shenyi-tang/stats506-computing-methods-and-tools.git

```
library(tidyverse)
library(nycflights13)
library(ggplot2)
library(MetBrewer)
```

Problem 1 - Tidyverse

a. **Generate a table reporting the mean and median departure delay per airport. Generate a second table reporting the mean and median arrival delay per airport. Exclude any destination with under 10 flights.**

   1. **Order both tables in descending mean delay**
   2. **Both tables should use the airport names not the airport codes**
   3. **Both tables should print all rows**

```
# merge airports and flights to get the airport names of departure and arrival
data(flights)
data(airports)

f2 <- flights %>%
  left_join(.,
            airports %>% select(faa, name),
            by = c("origin" = "faa")) %>%
  rename(dept_name = name) %>%
  left_join(.,
            airports %>% select(faa, name),
            by = c("dest" = "faa")) %>%
  rename(dest_name = name)

# count the arrivals of destination airports
# and filter the destination airports with more than 10 flights
dest_apt <- f2 %>% group_by(dest_name) %>%
```

```
  summarise(cnt = length(flight)) %>%
  filter(cnt >= 10) %>%
  select(dest_name) %>%
  pull()


# Tibble 1 - departure delay
t1 <- f2 %>% filter(dest_name %in% dest_apt) %>%
  group_by(dept_name) %>%
  summarise(dept_delay_mean = mean(dep_delay, na.rm = TRUE),
            dept_delay_median = median(dep_delay, na.rm = TRUE)) %>%
  arrange(desc(dept_delay_mean))
print(t1, n=98)
```

```
# A tibble: 3 x 3
  dept_name           dept_delay_mean dept_delay_median
  <chr>                         <dbl>             <dbl>
1 Newark Liberty Intl            15.1                -1
2 John F Kennedy Intl            12.1                -1
3 La Guardia                     10.3                -3
```

```
# Tibble 2 - Arrival delay
t2 <- f2 %>% filter(dest_name %in% dest_apt) %>%
  group_by(dest_name) %>%
  summarise(arr_delay_mean = mean(arr_delay, na.rm = TRUE),
            arr_delay_median = median(arr_delay, na.rm = TRUE)) %>%
  arrange(desc(arr_delay_mean))
print(t2, n=98)
```

```
# A tibble: 99 x 3
   dest_name                              arr_delay_mean arr_delay_median
   <chr>                                           <dbl>             <dbl>
 1 "Columbia Metropolitan"                          41.8                28
 2 "Tulsa Intl"                                     33.7                14
 3 "Will Rogers World"                              30.6                16
 4 "Jackson Hole Airport"                           28.1                15
 5 "Mc Ghee Tyson"                                  24.1                 2
 6 "Dane Co Rgnl Truax Fld"                         20.2                 1
 7 "Richmond Intl"                                  20.1                 1
 8 "Akron Canton Regional Airport"                  19.7                 3
 9 "Des Moines Intl"                                19.0                 0
10 "Gerald R Ford Intl"                             18.2                 1
11 "Birmingham Intl"                                16.9                -2
12 "Theodore Francis Green State"                   16.2                 1
13 "Greenville-Spartanburg International"           15.9                -0.5
14 "Cincinnati Northern Kentucky Intl"              15.4                -3
15 "Savannah Hilton Head Intl"                      15.1                -1
16 "Manchester Regional Airport"                    14.8                -3
17 "Eppley Afld"                                    14.7                -2
```

18	"Yeager"	14.7	-1.5
19	"Kansas City Intl"	14.5	0
20	"Albany Intl"	14.4	-4
21	"General Mitchell Intl"	14.2	0
22	"Piedmont Triad"	14.1	-2
23	"Washington Dulles Intl"	13.9	-3
24	"Cherry Capital Airport"	13.0	-10
25	"James M Cox Dayton Intl"	12.7	-3
26	"Louisville International Airport"	12.7	-2
27	"Chicago Midway Intl"	12.4	-1
28	"Sacramento Intl"	12.1	4
29	"Jacksonville Intl"	11.8	-2
30	"Nashville Intl"	11.8	-2
31	"Portland Intl Jetport"	11.7	-4
32	"Greater Rochester Intl"	11.6	-5
33	"Hartsfield Jackson Atlanta Intl"	11.3	-1
34	"Lambert St Louis Intl"	11.1	-3
35	"Norfolk Intl"	10.9	-4
36	"Baltimore Washington Intl"	10.7	-5
37	"Memphis Intl"	10.6	-2.5
38	"Port Columbus Intl"	10.6	-3
39	"Charleston Afb Intl"	10.6	-4
40	"Philadelphia Intl"	10.1	-3
41	"Raleigh Durham Intl"	10.1	-3
42	"Indianapolis Intl"	9.94	-3
43	"Charlottesville-Albemarle"	9.5	-5
44	"Cleveland Hopkins Intl"	9.18	-5
45	"Ronald Reagan Washington Natl"	9.07	-2
46	"Burlington Intl"	8.95	-4
47	"Buffalo Niagara Intl"	8.95	-5
48	"Syracuse Hancock Intl"	8.90	-5
49	"Denver Intl"	8.61	-2
50	"Palm Beach Intl"	8.56	-3
51	"Bob Hope"	8.18	-3
52	"Fort Lauderdale Hollywood Intl"	8.08	-3
53	"Bangor Intl"	8.03	-9
54	"Asheville Regional Airport"	8.00	-1
55	"Pittsburgh Intl"	7.68	-5
56	"Gallatin Field"	7.6	-2
57	"NW Arkansas Regional"	7.47	-2
58	"Tampa Intl"	7.41	-4
59	"Charlotte Douglas Intl"	7.36	-3
60	"Minneapolis St Paul Intl"	7.27	-5
61	"William P Hobby"	7.18	-4
62	"Bradley Intl"	7.05	-10
63	"San Antonio Intl"	6.95	-9
64	"South Bend Rgnl"	6.5	-3.5
65	"Louis Armstrong New Orleans Intl"	6.49	-6
66	"Key West Intl"	6.35	7
67	"Eagle Co Rgnl"	6.30	-4

```
68 "Austin Bergstrom Intl"                      6.02              -5
69 "Chicago Ohare Intl"                         5.88              -8
70 "Orlando Intl"                               5.45              -5
71 "Detroit Metro Wayne Co"                     5.43              -7
72 "Portland Intl"                              5.14              -5
73 "Nantucket Mem"                              4.85              -3
74 "Wilmington Intl"                            4.64              -7
75 "Myrtle Beach Intl"                          4.60             -13
76 "Albuquerque International Sunport"          4.38              -5.5
77 "George Bush Intercontinental"               4.24              -5
78 "Norman Y Mineta San Jose Intl"              3.45              -7
79 "Southwest Florida Intl"                     3.24              -5
80 "San Diego Intl"                             3.14              -5
81 "Sarasota Bradenton Intl"                    3.08              -5
82 "Metropolitan Oakland Intl"                  3.08              -9
83  <NA>                                        3.01              -5
84 "General Edward Lawrence Logan Intl"         2.91              -9
85 "San Francisco Intl"                         2.67              -8
86 "Yampa Valley"                               2.14               2
87 "Phoenix Sky Harbor Intl"                    2.10              -6
88 "Montrose Regional Airport"                  1.79             -10.5
89 "Los Angeles Intl"                           0.547             -7
90 "Dallas Fort Worth Intl"                     0.322             -9
91 "Miami Intl"                                 0.299             -9
92 "Mc Carran Intl"                             0.258             -8
93 "Salt Lake City Intl"                        0.176             -8
94 "Long Beach"                                -0.0620           -10
95 "Martha\\\\'s Vineyard"                     -0.286            -11
96 "Seattle Tacoma Intl"                       -1.10             -11
97 "Honolulu Intl"                             -1.37              -7
98 "John Wayne Arpt Orange Co"                 -7.87             -11
# i 1 more row
```

b. How many flights did the aircraft model with the fastest average speed take? Produce a tibble with 1 row, and entries for the model, average speed (in MPH) and number of flights.

```r
# calculate the flight speed
# join with planes
f3 <- f2 %>%
  mutate(flight.speed = distance / (air_time / 60)) %>%
  left_join(., planes, by = c("tailnum" = "tailnum"))

# filter out the model with fastest average speed
fastest.model <- f3 %>%
  group_by(model) %>%
  summarise(avg_speed = mean(flight.speed, na.rm = TRUE)) %>%
  arrange(desc(avg_speed)) %>%
  slice(1) %>%
  select(model) %>%
```

```
  pull()

# detailed information of the fastest model
fastest.model.information <- f3 %>%
  group_by(model) %>%
  summarise(avg_speed = mean(flight.speed, na.rm = TRUE),
            num_of_flights = length(flight)) %>%
  filter(model == fastest.model)

fastest.model.information
```

```
# A tibble: 1 x 3
  model   avg_speed num_of_flights
  <chr>       <dbl>          <int>
1 777-222      483.              4
```

Problem 2 - get_temp()

a. **Load the Chicago NMMAPS data we used in the visualization lectures. Write a function `get_temp()` that allows a user to request the average temperature for a given month. The arguments should be:**

   1. `month` **Month, either a numeric 1-12 or a string.**
   2. `year` **A numeric year.**
   3. `data` **The data set to obtain data from.**
   4. `celsius` **Logically indicating whether the results should be in celsius. Default** `FALSE`**.**
   5. `average_fn` **A function with which to compute the mean. Default is** `mean`**.**

```
#'
#' @title get_temp(), to get the temperature of specific year and month
#'
#' @param month, could be a numeric or a string
#' @param year, a numeric year
#' @param data, the original data set
#' @param celsius, a logical parameter, transfer the temperature to celsius if the
 ↪  argument equals to TRUE
#' @param average_fn, a function applied to the temperature
#'
#' @return a float number representing the temperature

nmmaps <- read_csv("chicago-nmmaps.csv",
                   show_col_types = FALSE)


get_temp <- function(Month, Year, data, celsius = FALSE, average_fn = mean){
  # sanitize the input
  # if argument Month is a numeric, it should between 1 and 12
```

```
  tryCatch({
    if(is.numeric(Month)){
      if(!Month %in% 1:12){
        stop("'Month' should be an integer between 1 and 12.")
      }
    }
    # if argument Month is a string, transfer it to integer
    else if(is.character(Month)){
      # pattern match
      Month <- ifelse(nchar(Month) > 3,
                      match(tolower(Month), tolower(month.name)),
                      match(tolower(Month), tolower(month.abb)))
      if(is.na(Month)){
        stop("It is an invalid month name. Please enter a valid month name.")
      }
    }
    if(!is.numeric(Year)){
      stop("'Year' should be an integer")
    }

    temp_data <- data %>%
      filter(year(date) == Year,
             month(date) == Month) %>%
      summarise(temp_stat = average_fn(temp[!is.na(temp)])) %>%
      pull(temp_stat)

    if(celsius == TRUE){
      temp_data <- (temp_data - 32) * 5 /9
    }

    return(temp_data)},
    error = function(e){
      message("Error:", e$message)
    }
  )
}

get_temp("Apr", 1999, data = nmmaps)
```

```
[1] 49.8
```

```
get_temp("Apr", 1999, data = nmmaps, celsius = TRUE)
```

```
[1] 9.888889
```

```
get_temp(10, 1998, data = nmmaps, average_fn = median)
```

```
[1] 55
```

```r
get_temp(13, 1998, data = nmmaps)
```

```
Error:'Month' should be an integer between 1 and 12.
```

```r
get_temp(2, 2005, data = nmmaps)
```

```
[1] NaN
```

```r
get_temp("November", 1999, data = nmmaps, celsius = TRUE,
         average_fn = function(x) {
           x %>% sort -> x
           x[2:(length(x) - 1)] %>% mean %>% return
         })
```

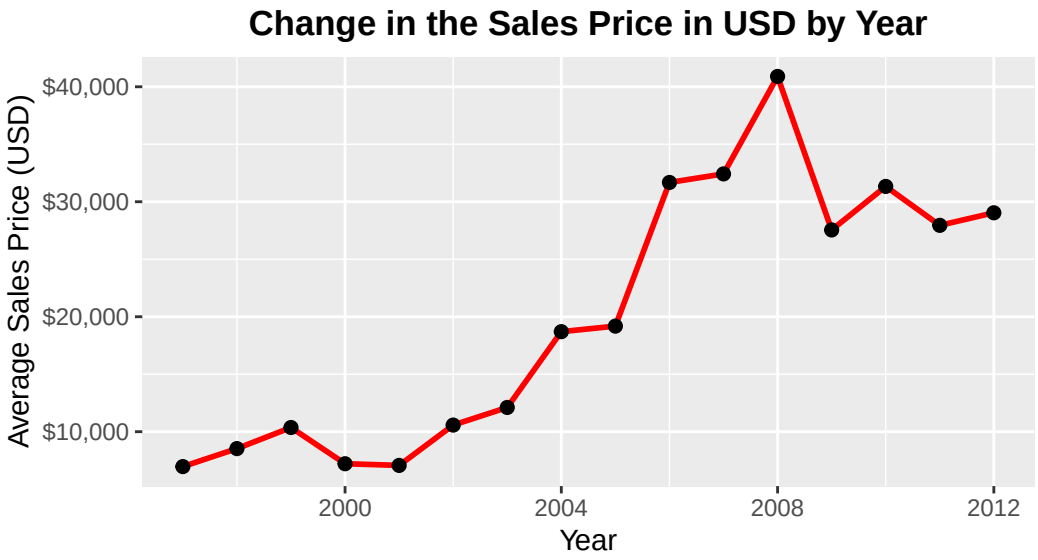
```
[1] 7.301587
```

Problem 3 - Visualization

```r
df <- read.csv("df_for_ml_improved_new_market.csv")
```

a. Is there a change in the sales price in USD over time?

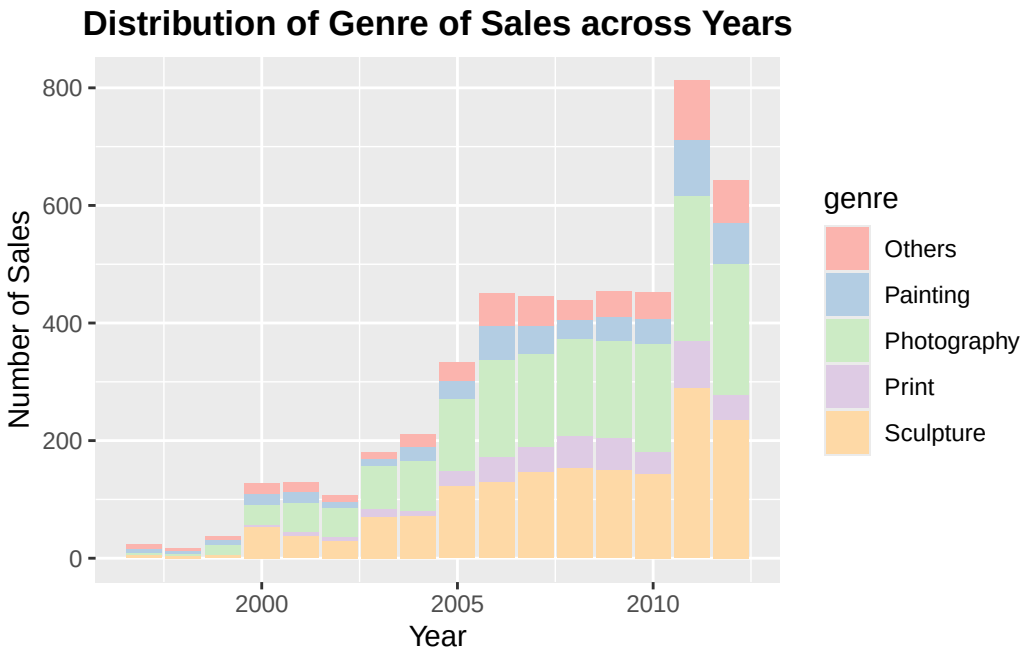
```r
# calculate the average price in each year
df %>%
  group_by(year) %>%
  summarise(avg_price = mean(price_usd, na.rm = TRUE)) %>%
  # line plot
  ggplot(aes(x = year, y = avg_price)) +
    geom_line(color = "red", lwd = 1) +
    geom_point(color = "black", size = 2) +
    labs(
      title = "Change in the Sales Price in USD by Year",
      x = "Year",
      y = "Average Sales Price (USD)"
    ) +
    scale_y_continuous(labels = scales::dollar) +
    # bold and center the title
    theme(plot.title = element_text(hjust = 0.5,
                                    face = "bold"))
```



- Yes, there are obvious changes over time. The average sales price gradually increase from 2000 to 2008. And there is a sharp drop in 2009.

b. Does the distribution of genre of sales across years appear to change?

```
# transfer the origin data set to longer table
# remove the prefix "Genre___"
pivot_df <- df %>%
  tidyr::pivot_longer(
    cols = c(Genre___Photography, Genre___Print, Genre___Sculpture, Genre___Painting,
     ↪  Genre___Others),
    names_to = 'genre',
    values_to = 'if.genre'
  ) %>%
  filter(if.genre == 1) %>%
  mutate(genre = gsub(".*___", "", genre))


pivot_df %>%
  group_by(year, genre) %>%
  summarise(count = n()) %>%
  ggplot(aes(x = year, y = count, fill = genre)) +
  geom_bar(stat = "identity", position = "stack") +
  labs(title = "Distribution of Genre of Sales across Years",
       x = "Year",
       y = "Number of Sales") +
  theme(plot.title = element_text(hjust = 0.5,
                                  face = "bold")) +
  scale_fill_brewer(palette = "Pastel1")
```



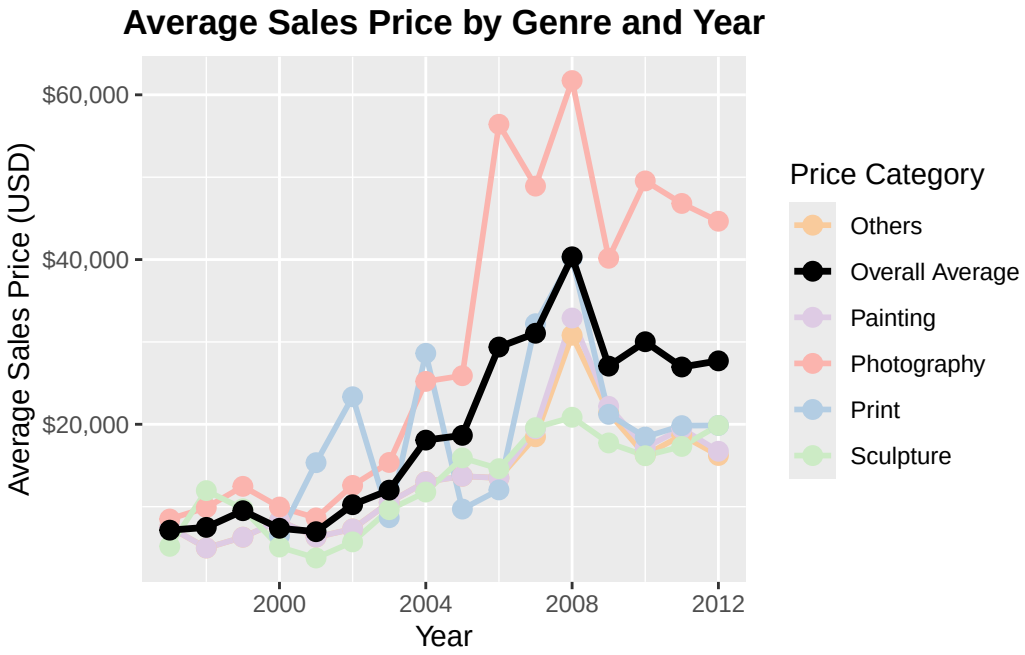
- Generally, sales of each genre increase over year. By 2011, the number of sales are much more higher than other years.

c. How does the genre affect the change in sales price over time?

```
# calculate the average of each genre in each year
genre_price_trends <- pivot_df %>%
  group_by(year, genre) %>%
  summarise(avg_price = mean(price_usd, na.rm = TRUE), .groups = 'drop')

# calculate the overall average price of each year
overall_price_trend <- pivot_df %>%
  group_by(year) %>%
  summarise(overall_avg_price = mean(price_usd, na.rm = TRUE), .groups = 'drop')

# do the line plot
ggplot() +
  # line for genre_price_trends
  geom_line(data = genre_price_trends,
            aes(x = year, y = avg_price, color = genre), lwd = 1) +
  geom_point(data = genre_price_trends,
             aes(x = year, y = avg_price, color = genre), size = 3) +

  # line for overall_price_trend
  geom_line(data = overall_price_trend,
            aes(x = year, y = overall_avg_price, color = "Overall Average"),
            size = 1.2) +
  geom_point(data = overall_price_trend,
             aes(x = year, y = overall_avg_price, color = "Overall Average"),
             size = 3) +
```

```r
# set format
labs(title = "Average Sales Price by Genre and Year",
     x = "Year",
     y = "Average Sales Price (USD)",
     color = "Price Category") +  # update the title of the legend
scale_y_continuous(labels = scales::dollar) +
scale_color_manual(values = c("Overall Average" = "black",
                              "Photography" = "#FBB4AE",
                              "Print" = "#B3CDE3",
                              "Sculpture" = "#CCEBC5",
                              "Painting" = "#DECBE4",
                              "Others" = "#F9CB9C")) +
theme(plot.title = element_text(hjust = 0.5, face = "bold"))
```

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



- According to the plot above, photography and prints showed the most dramatic price changes, while sculpture maintained the most stable pricing over time.