Statistics 506- Problem Set #4

Garrett Pinkston

Link to GitHub

github: https://github.com/garrettpinkston2015/Computational-Methods

Problem 1 - Tidyverse

Install and load the package nycflights13.

```
# load in data
library(nycflights13)
```

a) Generate a table (which can just be a nicely printed tibble) reporting the mean and median departure delay per airport. Generate a second table (which again can be a nicely printed tibble) reporting the mean and median arrival delay per airport. Exclude any destination with under 10 flights. Do this exclusion through code, not manually.

Additionally,

- Order both tables in descending mean delay.
- Both tables should use the airport names not the airport codes.
- Both tables should print all rows.

```
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
#display maximum rows of tibble
options(tibble.print_max = Inf)


depStats <- flights %>%
  #group by destination
  group_by(dest) %>%

  # filter where destinations are 10 or more
  filter(n() >= 10) %>%

  # specify function output via summary
  summarize(
    mean_dep_delay = mean(dep_delay, na.rm = TRUE), # return mean in summary
    median_dep_delay = median(dep_delay, na.rm = TRUE) # return median in summary
  ) %>%
  ungroup() %>% #ungroup the data

  #join the two datasets using "dest" and "faa"
  left_join(airports, by = c("dest" = "faa")) %>% #

  #select relevant columns of the output
  select(name, mean_dep_delay, median_dep_delay) %>%

  #arrange by descending mean delay
  arrange(desc(mean_dep_delay))

# print output
print(depStats)
```

```
# A tibble: 102 x 3
    name                                mean_dep_delay median_dep_delay
    <chr>                                        <dbl>            <dbl>
  1 "Columbia Metropolitan"                       35.6               14
  2 "Tulsa Intl"                                  34.9                8
  3 "Will Rogers World"                           30.6               10
  4 "Birmingham Intl"                             29.7                1
  5 "Mc Ghee Tyson"                               28.5                0
```

6	"Jackson Hole Airport"	26.5	13.5
7	"Des Moines Intl"	26.2	-1
8	"Richmond Intl"	23.6	-1
9	"Albany Intl"	23.6	1
10	"Dane Co Rgnl Truax Fld"	23.6	-1
11	"Cherry Capital Airport"	22.1	-3
12	"Theodore Francis Green State"	21.8	0
13	"Charlottesville-Albemarle"	21.4	-2.5
14	"South Bend Rgnl"	21.1	14
15	"Manchester Regional Airport"	21.0	0
16	"Akron Canton Regional Airport"	20.8	0
17	"San Antonio Intl"	20.7	1
18	"Kansas City Intl"	20.3	-1
19	"Eppley Afld"	20.2	-1
20	"Gerald R Ford Intl"	19.5	-1
21	"Cincinnati Northern Kentucky Intl"	19.5	-2
22	"Bangor Intl"	19.5	-2
23	"Wilmington Intl"	19.4	-3
24	"Piedmont Triad"	19.4	-1
25	"Greenville-Spartanburg International"	19.3	-1
26	"General Mitchell Intl"	18.8	0
27	"Sacramento Intl"	18.7	2
28	"Chicago Midway Intl"	18.6	2
29	"Savannah Hilton Head Intl"	18.3	-1
30	"Bradley Intl"	17.7	-1
31	"Montrose Regional Airport"	17.6	3
32	"Norfolk Intl"	17.6	-2
33	"James M Cox Dayton Intl"	17.5	-2
34	"Yeager"	17	-4
35	"Washington Dulles Intl"	17.0	-2
36	"Jacksonville Intl"	16.5	-1
37	"Portland Intl Jetport"	16.5	-2
38	"Louisville International Airport"	16.4	-2
39	"Baltimore Washington Intl"	16.4	-2
40	"Portland Intl"	16.3	1
41	"Greater Rochester Intl"	16.2	-2
42	"Lambert St Louis Intl"	16.0	-1
43	"Nashville Intl"	16.0	-1
44	"Myrtle Beach Intl"	15.8	-1
45	"Memphis Intl"	15.7	-1
46	"Eagle Co Rgnl"	15.5	-1
47	"Denver Intl"	15.2	1
48	"Charleston Afb Intl"	14.7	-2

49	"Syracuse Hancock Intl"	14.4	-2
50	"William P Hobby"	14.3	0
51	"Louis Armstrong New Orleans Intl"	14.2	-2
52	"Indianapolis Intl"	14.0	-2
53	"Albuquerque International Sunport"	13.7	0
54	"Pittsburgh Intl"	13.7	-2
55	"Burlington Intl"	13.6	-2
56	"Chicago Ohare Intl"	13.6	-2
57	"Bob Hope"	13.5	-1
58	"Buffalo Niagara Intl"	13.4	-2
59	"Cleveland Hopkins Intl"	13.4	-2
60	"Metropolitan Oakland Intl"	13.3	0
61	"Minneapolis St Paul Intl"	13.3	-2
62	"Austin Bergstrom Intl"	13.0	-1
63	"Palm Beach Intl"	13.0	0
64	"San Francisco Intl"	12.9	0
65	"Fort Lauderdale Hollywood Intl"	12.7	-1
66	"Hartsfield Jackson Atlanta Intl"	12.5	-2
67	"Raleigh Durham Intl"	12.4	-2
68	<NA>	12.4	-1
69	"Yampa Valley"	12.3	6.5
70	"Port Columbus Intl"	12.2	-3
71	"Tampa Intl"	12.1	-1
72	"Philadelphia Intl"	12.0	-3
73	"Detroit Metro Wayne Co"	11.8	-3
74	"Gallatin Field"	11.5	0
75	"Orlando Intl"	11.3	-1
76	"Long Beach"	11.2	-1
77	"San Diego Intl"	11.1	0
78	"George Bush Intercontinental"	10.8	0
79	"Seattle Tacoma Intl"	10.7	-1
80	"Phoenix Sky Harbor Intl"	10.4	-1
81	"Ronald Reagan Washington Natl"	10.3	-3
82	<NA>	10.1	-1
83	"Norman Y Mineta San Jose Intl"	10.1	-1
84	<NA>	9.81	-1
85	"Mc Carran Intl"	9.42	-1
86	"Los Angeles Intl"	9.40	-1
87	"Honolulu Intl"	9.29	-1
88	"Charlotte Douglas Intl"	9.22	-3
89	"Salt Lake City Intl"	9.03	-1
90	"Miami Intl"	8.88	-2
91	"General Edward Lawrence Logan Intl"	8.73	-3

```
 92 "Dallas Fort Worth Intl"                    8.68            -3
 93 "Southwest Florida Intl"                    8.28            -2
 94 "Asheville Regional Airport"                8.19            -3
 95 "John Wayne Arpt Orange Co"                 7.76            -1
 96 "Sarasota Bradenton Intl"                   7.26            -3
 97 "Martha\\\\'s Vineyard"                     7.05            -2
 98 "NW Arkansas Regional"                      6.46            -5
 99 "Nantucket Mem"                             6.46            -3
100  <NA>                                      4.61            -2
101 "Key West Intl"                            3.65             0
102 "Palm Springs Intl"                       -2.94            -4
```

```r
arrivalStats <- flights %>%
  #group by the destination
  group_by(dest) %>%

  # filter where destinations are 10 or more
  filter(n() >= 10) %>%

  # specify output via summary
  summarize(
    mean_arr_delay = mean(arr_delay, na.rm = TRUE), # create mean arrival stat
    median_arr_delay = median(arr_delay, na.rm = TRUE) # create median arrival
  ) %>%
  ungroup() %>% #ungroup data

  #join datasets by airports
  left_join(airports, by = c("dest" = "faa")) %>%

  #select relevant columns of the output
  select(name, mean_arr_delay, median_arr_delay) %>%

  #arrange by descending mean delay
  arrange(desc(mean_arr_delay))

print(arrivalStats)
```

```
# A tibble: 102 x 3
    name                                mean_arr_delay median_arr_delay
    <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	<NA>	8.25	-1
52	"Bob Hope"	8.18	-3
53	"Fort Lauderdale Hollywood Intl"	8.08	-3
54	"Bangor Intl"	8.03	-9
55	"Asheville Regional Airport"	8.00	-1
56	<NA>	7.87	0
57	"Pittsburgh Intl"	7.68	-5
58	"Gallatin Field"	7.6	-2
59	"NW Arkansas Regional"	7.47	-2
60	"Tampa Intl"	7.41	-4
61	"Charlotte Douglas Intl"	7.36	-3
62	"Minneapolis St Paul Intl"	7.27	-5
63	"William P Hobby"	7.18	-4
64	"Bradley Intl"	7.05	-10
65	"San Antonio Intl"	6.95	-9
66	"South Bend Rgnl"	6.5	-3.5
67	"Louis Armstrong New Orleans Intl"	6.49	-6
68	"Key West Intl"	6.35	7
69	"Eagle Co Rgnl"	6.30	-4
70	"Austin Bergstrom Intl"	6.02	-5
71	"Chicago Ohare Intl"	5.88	-8
72	"Orlando Intl"	5.45	-5
73	"Detroit Metro Wayne Co"	5.43	-7
74	"Portland Intl"	5.14	-5
75	"Nantucket Mem"	4.85	-3
76	"Wilmington Intl"	4.64	-7
77	"Myrtle Beach Intl"	4.60	-13
78	"Albuquerque International Sunport"	4.38	-5.5
79	"George Bush Intercontinental"	4.24	-5
80	"Norman Y Mineta San Jose Intl"	3.45	-7
81	"Southwest Florida Intl"	3.24	-5
82	"San Diego Intl"	3.14	-5
83	"Sarasota Bradenton Intl"	3.08	-5
84	"Metropolitan Oakland Intl"	3.08	-9
85	"General Edward Lawrence Logan Intl"	2.91	-9
86	"San Francisco Intl"	2.67	-8
87	<NA>	2.52	-6
88	"Yampa Valley"	2.14	2

```
 89 "Phoenix Sky Harbor Intl"                2.10              -6
 90 "Montrose Regional Airport"              1.79             -10.5
 91 "Los Angeles Intl"                       0.547             -7
 92 "Dallas Fort Worth Intl"                 0.322             -9
 93 "Miami Intl"                             0.299             -9
 94 "Mc Carran Intl"                         0.258             -8
 95 "Salt Lake City Intl"                    0.176             -8
 96 "Long Beach"                            -0.0620           -10
 97 "Martha\\\\'s Vineyard"                 -0.286            -11
 98 "Seattle Tacoma Intl"                   -1.10             -11
 99 "Honolulu Intl"                         -1.37              -7
100  <NA>                                   -3.84              -9
101 "John Wayne Arpt Orange Co"             -7.87             -11
102 "Palm Springs Intl"                    -12.7              -13.5
```

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

```r
flights_planes <- flights %>%
  #join datasets by tail number
  left_join(planes, by = "tailnum")

flights_speeds <- flights_planes %>%
  # filter isna
  filter(!is.na(air_time), !is.na(distance), !is.na(model)) %>%

  #calculate speed in mph
  mutate(speed_mph = distance / (air_time / 60))

model_speeds <- flights_speeds %>%
  #group by model
  group_by(model) %>%

  # specify output via summary
  summarize(
    avg_speed_mph = mean(speed_mph, na.rm = TRUE), # calculate average speed
    num_flights = n() #count number of flights
  ) %>%
  ungroup() #ungroup data


# finding the fastest plane
```

```r
fastest_model <- model_speeds %>%
  filter(avg_speed_mph == max(avg_speed_mph)) %>% #filter max speed
  slice(1) #take first instance

print(fastest_model)
```

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

Problem 2 - get_temp()

Load the Chicago NNMAPS 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:

- month: Month, either a numeric 1-12 or a string.
- year: A numeric year.
- data: The data set to obtain data from.
- celsius: Logically indicating whther the results should be in celsius. Default FALSE.
- average_fn: A function with which to compute the mean. Default is mean.
- The output should be a numeric vector of length 1. The code inside the function should, as with the rest of this problem, use the tidyverse. Be sure to sanitize the input.

Prove your code works by evaluating the following. Your code should produce the result, or a reasonable error message.

```r
# read in nnmaps dataset
nnmaps <- read.csv("/Users/garrettpinkston/Desktop/Michigan/STAT506/Data/chicago-nmmaps.csv")
```

```r
library(dplyr)
library(stringr)

#' Retrieves the average temperature for a specified month and year.
#'
#' @param month Either a numeric value (1-12) or a string representing the month
#' (e.g., "Jan" or "February").
#' @param year A numeric value between 1997 and 2000, specifying the year for
#' which the temperature is requested.
#' @param data A dataset containing temperature data, with columns including
```

```r
#' year, month, and temp.
#' @param celsius Logical; if TRUE, converts the average temperature to Celsius.
#' Default is FALSE (Fahrenheit).
#' @param average_fn A function used to compute the average (e.g., mean).
#' Default is mean.
#'
#' @return The average temperature for the specified month and year, in
#' Fahrenheit or Celsius as specified.
#' @export
#'
#' @examples
#' get_temp("Apr", 1999, data = nnmaps)
#' 49.7

get_temp <- function(month, year, data, celsius = FALSE, average_fn = mean) {

  # Check if the year input is valid: a single numeric value between 1997 and 2000.
  if (!is.numeric(year) || length(year) != 1 || ((year > 2000) || (year < 1997))) {
    stop("Year must be provided as a single numeric value between 1997 and 2000.")
  }

  # Determine if the month is numeric or character
  if (is.numeric(month)) {
    # Validate numeric month range (1-12)
    if (month < 1 || month > 12) {
      stop("Month must be a number between 1 and 12.")
    }
    month_type <- "numeric"
  } else if (is.character(month)) {
    # If month is a character, convert it to a three-letter abbreviation in lowercase
    month <- str_sub(str_to_lower(month), 1, 3)
    month_type <- "string"
  } else {
    # Error if month is not numeric or string
    stop("Month must be either a numeric (1-12) or a string
         (e.g., 'Jan', 'February').")
  }

  # Filter the data based on the month and year type (numeric or string)
  if (month_type == "numeric") {
    result <- data %>%
      filter(year == !!year, month_numeric == !!month) %>%
```

```
      summarize(avg_temp = average_fn(temp, na.rm = TRUE))
  } else {
    result <- data %>%
      filter(year == !!year, str_sub(str_to_lower(.$month), 1, 3) == !!month) %>%
      summarize(avg_temp = average_fn(temp))
  }

  # Error if no data matches the specified criteria
  if (nrow(result) == 0) {
    stop("No matching data for the specified month and year.")
  }

  # Extract the average temperature
  avg_temp <- result$avg_temp
  # Convert to Celsius if requested
  if (celsius) {
    avg_temp <- (avg_temp - 32) * (5/9)
  }

  # Return the final average temperature
  return(avg_temp)
}
```

```
# running through example codes

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

```
[1] 49.8
```

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

```
[1] 9.888889
```

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

```
[1] 55
```

```
get_temp(13, 1998, data = nnmaps)
```

```
Error in get_temp(13, 1998, data = nnmaps): Month must be a number between 1 and 12.
```

```
get_temp(2, 2005, data = nnmaps)
```

```
Error in get_temp(2, 2005, data = nnmaps): Year must be provided as a single numeric value be
```

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

```
[1] 7.301587
```

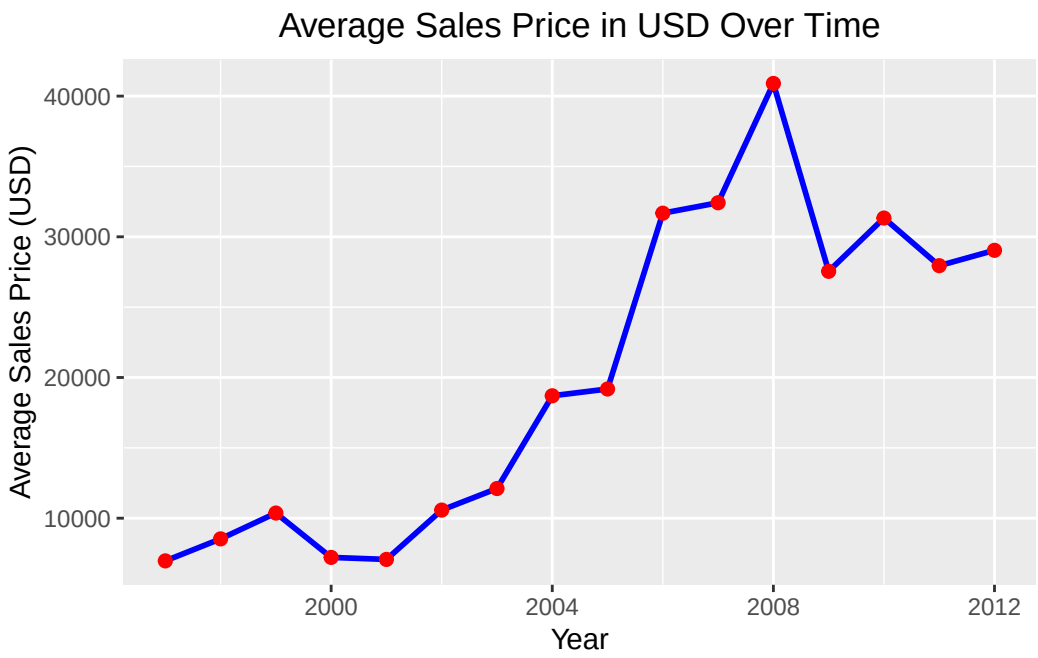
Problem 3 - Visualization

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

```
art <- read.csv("/Users/garrettpinkston/Desktop/Michigan/STAT506/Data/df_for_ml_improved_new
```

```
# load in ggplot2
library(ggplot2)

art %>%

  # group data by 'year' to calculate average sales price per year
  group_by(year) %>%

  # calculate the average sales price in USD for each year
  summarize(avg_sales_price = mean(price_usd, na.rm = TRUE)) %>%

  # initialize ggplot, setting 'year' as x-axis and 'avg_sales_price' as y-axis
  ggplot(aes(x = year, y = avg_sales_price)) +

  # add a line plot for average sales price with blue color
  geom_line(color = "blue", linewidth = 1) +

  # add red points at each year
  geom_point(color = "red", size = 2) +
```

```
  # add title and axes labels
  labs(title = "Average Sales Price in USD Over Time",
       x = "Year",
       y = "Average Sales Price (USD)") +

  # align plot title
  theme(plot.title = element_text(hjust = 0.5))
```



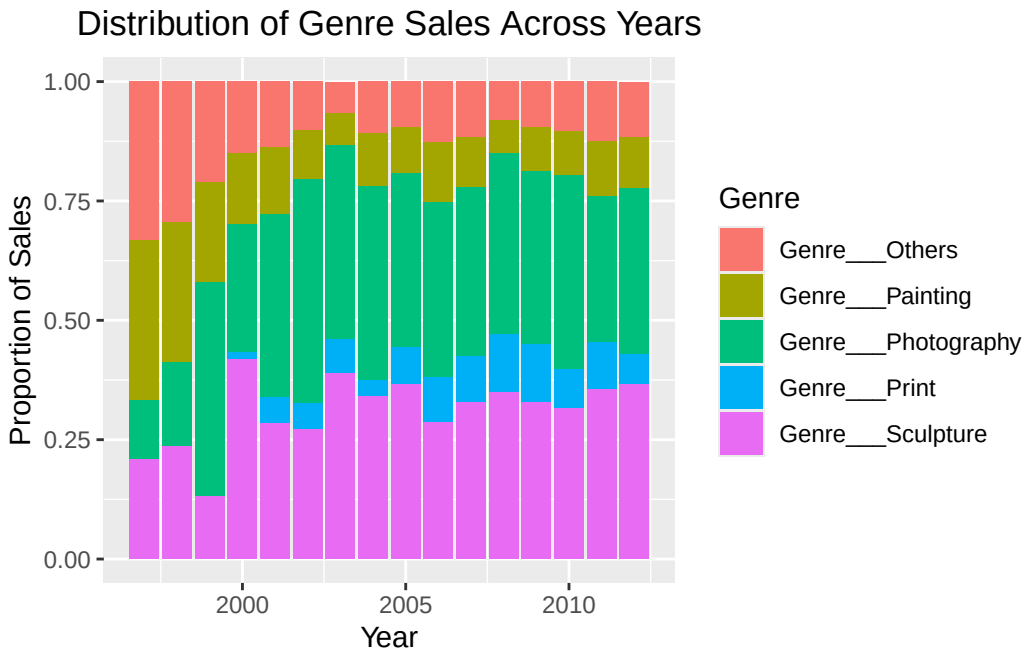
Based on the graph, we can tell that the average sales price has increased over time. The first point is around 7,000 around the year 1997 and it has risen to almost 29,000 in the year 2012. Based on the graph, we can visualize the dramatic increase in the average sales price over time.

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

```
# load in tidyr for data manipulation
library(tidyr)

# reshape the 'art' dataset to a long format
df_long <- art %>%
  # used pivot_longer to transform all columns starting with "Genre___" into
  # key-value pairs
  pivot_longer(cols = starts_with("Genre___"),
```

```
                names_to = "genre",        # New column for genre names
                values_to = "count") %>%   # New column for genre values
  # Filter when presence of a specific genre is in a sale
  filter(count == 1)

# visualize the yearly genre distribution
df_long %>%

  # group data by both year and genre
  group_by(year, genre) %>%

  # count occurrences for each genre per year
  summarize(count = n()) %>%

  # group data by year to compute the proportion of each genre
  group_by(year) %>%

  # calculate proportion of each genre's count relative to all other counts
  mutate(proportion = count / sum(count)) %>%

  # use ggplot, setting 'year' as x-axis and 'proportion' as y-axis
  ggplot(aes(x = year, y = proportion, fill = genre)) +

  # using a stacked bar plot to show proportions per genre each year
  geom_bar(stat = "identity", position = "fill") +

  # adding title, x-axis, y-axis, and legend labels
  labs(title = "Distribution of Genre Sales Across Years",
       x = "Year",
       y = "Proportion of Sales",
       fill = "Genre") +

  # center the plot title
  theme(plot.title = element_text(hjust = 0.5))
```

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



Based on the graph, we can see the distribution of genre of sales across years changes heavily. "Photography" consistently holds a significant proportion, with its presence growing after 2000, while "Painting" sees a decline in proportional representation across the years. "Sculpture" has maintained steady sales, while "Print" and "Others" fluctuate but generally contribute smaller proportions. This suggests a shift in sales preference toward Photography as the years progress, indicating its rising popularity in the art market.

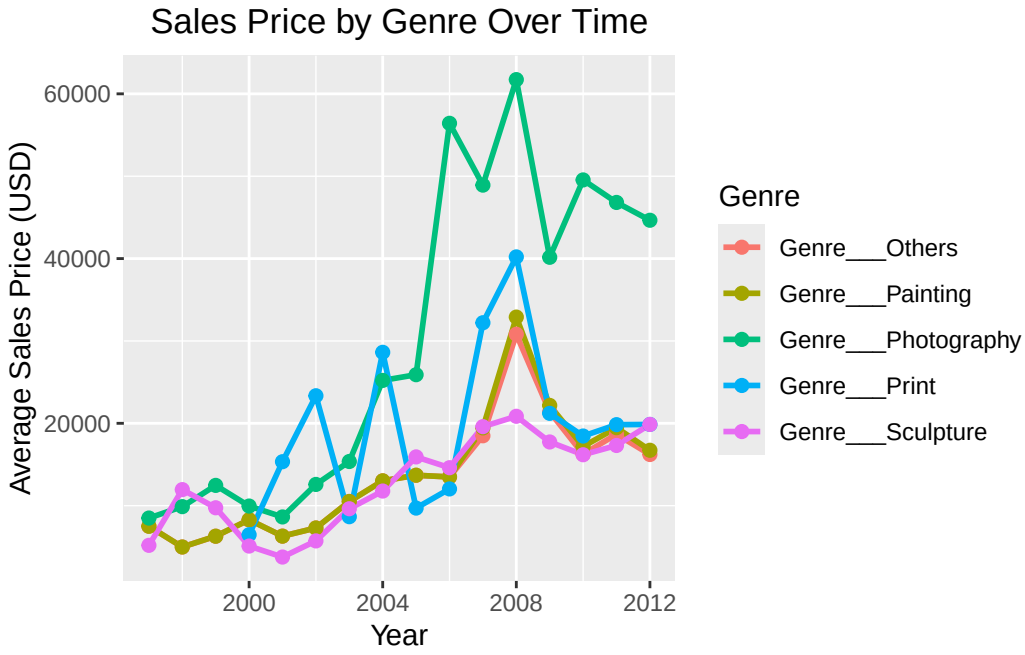
- How does the genre affect the change in sales price over time?

```
# reshape the 'art' dataset to a long format again
df_long <- art %>%
  # used pivot_longer to transform all columns starting with "Genre___" into
  # key-value pairs
  pivot_longer(cols = starts_with("Genre___"),
               names_to = "genre",       # New column for genre names
               values_to = "count") %>%   # New column for genre values
  # Filter when presence of a specific genre is in a sale
  filter(count == 1)

# visualize the yearly average sales prices
df_long %>%
  # group the data by 'year' and 'genre' to calculate average sales price
  group_by(year, genre) %>%
```

```r
  # calculate average sales price for each genre per year
  summarize(avg_sales_price = mean(price_usd, na.rm = TRUE)) %>%

  # use ggplot, setting 'year' as x, 'avg_sales_price' as y, colored by genre
  ggplot(aes(x = year, y = avg_sales_price, color = genre)) +

  # add a line for each genre's average sales price over time
  geom_line(size = 1) +

  # add points at each year for each genre
  geom_point(size = 2) +

  # adding title, x-axis, y-axis, and legend labels
  labs(title = "Sales Price by Genre Over Time",
       x = "Year",
       y = "Average Sales Price (USD)",
       color = "Genre") +

  # center title of the plot
  theme(plot.title = element_text(hjust = 0.5))
```

```
`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.
```



Here we can see that every genre has increased in popularity since the beginning year. Photography has increased the most, peaking in 2008 at a valuation around $60,000. "Print" also saw fluctuations and reached a peak around the same time but remained lower in value than "Photography." The other genres, including "Painting," "Sculpture," and "Others," remained relatively stable with smaller variations in their average sales prices over time. This suggests a surge in interest and value for "Photography" compared to other genres, particularly during the mid-2000s.