Data Science Activities Portfolio

Kartikeya Hitesh Rana

2025-11-19

Armed Forces Data Wrangling Redux

This section presents corrected and improved data wrangling code from previous activities, demonstrating the ability to transform military personnel data into analyzable formats. The code successfully processes Active-Duty Personnel data by Service Branch, Sex, and Pay Grade, creating both grouped summaries and individual-level records that can be used for subsequent analyses.

Data Wrangling Approach

The wrangling process involves reading the US Armed Forces dataset, reshaping it from wide to long format, and cleaning the categorical variables. The `pivot_longer()` function transforms multiple columns into key-value pairs, while `separate()` breaks compound column names into distinct Branch and Sex variables. This approach ensures the data structure is tidy and ready for analysis without requiring manual alterations on different systems.

Frequency Table Creation

Building on the wrangled data, a frequency table function was developed to summarize personnel counts by Pay Grade and Branch for a specified sex. This function filters the data, groups by the relevant categories, and produces a summary table showing the distribution of military personnel across different organizational levels. The table provides insights into the composition of each service branch by rank structure.

Popularity of Baby Names

The visualization shows trends in the popularity of four selected female names—Emma, Sophia, Jennifer, and Margaret—from 1880 to 2017. These names were chosen to represent different eras of American naming trends: Margaret represents classic early-20th-century names, Jennifer exemplifies the dramatic rise and fall of names in the 1970s-1990s, while Emma and Sophia demonstrate contemporary naming preferences that have surged in popularity since 2000.

The line graph reveals distinct patterns for each name. Jennifer shows the most dramatic peak, reaching over 50,000 births annually in the late 1970s before declining sharply. Margaret maintained steady moderate popularity in the early 1900s with a peak around 1920. Emma and Sophia both show recent surges beginning around 1995, with both names reaching over 20,000 births annually by the 2010s. These patterns reflect broader cultural shifts in naming conventions and the cyclical nature of name popularity across generations.

Plotting a Mathematical Function

This section revisits the box construction problem, where a rectangular piece of material is transformed into an open-top box by cutting equal squares from each corner and folding up the sides. The challenge involves calculating the volume of the resulting box as a function of the cut size, denoted as x.

Volume Function

The `box_volume()` function takes three parameters: the cut size x, and the original dimensions (length and width) of the rectangular material. The function first validates that the cut size is positive and less than half of the smaller dimension, ensuring a physically valid box. It then calculates the new dimensions after cutting and folding: the base dimensions are reduced by 2x (cutting from both sides), and the height equals x. The volume is computed as the product of these three dimensions.

Visualization Using ggplot2

To visualize how box volume changes with different cut sizes, the function was applied across a range of x values and plotted using `ggplot2` with the `stat_function()` approach. The resulting curve shows a clear maximum volume at an intermediate cut size—cutting too little or too much both result in suboptimal volumes. This optimization problem demonstrates the practical application of mathematical functions in design and manufacturing contexts.

What You Feel You've Learned So Far

Throughout these activities, several key data science competencies have been developed. The data wrangling exercises have reinforced the importance of tidy data principles and demonstrated how proper data structure facilitates analysis. Working with the Armed Forces dataset required careful attention to data types, handling of missing values, and understanding how different reshaping operations affect the data structure.

Visualization has emerged as a powerful tool for communicating patterns and trends. The baby names analysis showed how effective graphics can reveal historical trends and make comparisons across categories. The choice of visual encoding—using different colors and line types—proved essential for distinguishing between multiple time series. Additionally, the mathematical function plotting illustrated how visualization can aid in understanding optimization problems and functional relationships.

The process of debugging and refining code has been equally valuable. Creating code that runs without alteration on different systems requires careful file path management, explicit package loading, and thorough testing. Writing functions that accept parameters rather than hard-coding values increases code flexibility and reusability. These practices contribute to more robust and reproducible data analysis workflows.

Code Appendix

Activity 4: Box Volume Function

```r
# Activity 4 box
box_volume <- function(x, length, width) {
  if (x <= 0 || x >= min(length/2, width/2)) {
    return(0)
  }

  new_length <- length - 2*x
  new_width <- width - 2*x
  height <- x
  volume <- new_length * new_width * height
  return(volume)
}
```

Activity 8: Armed Forces Data Wrangling

```r
# Activity 8 Question 5
library(tidyverse)

armed_forces <- read_csv("US.csv")
glimpse(armed_forces)
View(armed_forces)
names(armed_forces)

groups_df <- armed_forces %>%
  pivot_longer(
    cols = -`Pay Grade`,
    names_to = "Category",
    values_to = "Count"
  ) %>%
  separate(Category, into = c("Branch", "Sex"), sep = " ", extra = "merge") %>%
  filter(!is.na(Count), Count > 0) %>%
  mutate(
    Branch = str_trim(Branch),
    Sex = str_trim(Sex),
    Pay_Grade = `Pay Grade`
  ) %>%
  select(Pay_Grade, Branch, Sex, Count)

individuals_df <- groups_df %>%
  uncount(weights = Count)

original_total <- armed_forces %>%
  select(-`Pay Grade`) %>%
  summarise(across(everything(), ~sum(., na.rm = TRUE))) %>%
  rowSums()

groups_total <- sum(groups_df$Count, na.rm = TRUE)
individuals_total <- nrow(individuals_df)
```

Activity 10: Frequency Table Function

```r
library(tidyverse)
```

```r
branch_sex <- c(
  "Army_Male", "Army_Female", "Army_Total",
  "Navy_Male", "Navy_Female", "Navy_Total",
  "Marine_Male", "Marine_Female", "Marine_Total",
  "AirForce_Male", "AirForce_Female", "AirForce_Total",
  "SpaceForce_Male", "SpaceForce_Female", "SpaceForce_Total",
  "Total_Male", "Total_Female", "Total_Total"
)

col_names <- c("Pay_Grade", branch_sex)
data <- read_csv("US.csv", skip=2, col_names = col_names)
data <- data %>% filter(!(Pay_Grade %in% c("Total Enlisted","Total Warrant Officers","Total (

long_data <- data %>%
  pivot_longer(-Pay_Grade, names_to = c("Branch", "Sex"), names_pattern = "(.*)_(Male|Female
  filter(Sex %in% c("Male", "Female"), !is.na(Count))

long_data$Count <- as.numeric(gsub(",", "", long_data$Count))

freq_table <- function(df, sex_label) {
  sex_data <- df %>% filter(Sex==sex_label)
  table <- sex_data %>%
    group_by(Pay_Grade, Branch) %>%
    summarise(Count = sum(Count, na.rm=TRUE), .groups='drop') %>%
    pivot_wider(names_from = Branch, values_from = Count, values_fill = 0)
  return(table)
}

# Example usage for male soldiers
male_table <- freq_table(long_data, "Male")
print(male_table)
```

Activity 13: Baby Names Visualization

```r
# R code for the appropriate data wrangling of the BabyNames data frame to focus on just the

library(dcData)
library(dplyr)

selected_names <- c("Emma", "Sophia", "Jennifer", "Margaret")
```

```r
BabyNames_filtered <- BabyNames %>%
  filter(name %in% selected_names) %>%
  group_by(year, name) %>%
  summarise(total_count = sum(count), .groups = 'drop')

# R code for your data visualization showing the popularity of your selected names over time

library(ggplot2)

ggplot(BabyNames_filtered, aes(x = year, y = total_count, color = name, linetype = name)) +
  geom_line(size = 1) +
  labs(
    title = "Popularity of Selected Female Names Over Time",
    x = "Year",
    y = "Total Number of People with Name",
    color = "Name",
    linetype = "Name"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(size = 14, face = "bold"),
    legend.position = "right",
    panel.grid.major = element_line(color = "gray90")
  )
```