Lab 2: Data Tidying

Overview

In this assignment you will work to tidy, clean, and analyze two different datasets, the first is a small dataset contained in a csv file called flightdelays.csv, and the second called MixedDrinkRecipes-Prep.csv.

The most important book chapters which cover the techniques you will practice here are R4DS Chapters 5 and 7. Also helpful are the `tidyr` vignette on pivoting and the ggplot help page on the geom_dotplot.

Submit your completed assignment on the course brightspace page by uploading your `.qmd` file and a compiled `pdf` or link to a compiled `html`, which you could host on your `github` or `rpubs` page as you wish.

Part 1: Airplane flight delays

Consider the following dataset:

		Los_Angeles	Phoenix	San_Diego	San_Francisco	Seattle
ALASKA	On_Time	497	221	212	503	1841
	Delayed	62	12	20	102	305
AM WEST	On_Time	694	4840	383	320	301
	Delayed	117	415	65	129	61

The above table describes arrival delays for two different airlines across several destinations. The numbers correspond the the number of flights that were in either the delayed category or the on time category.

Problems

Problem 1: Read the information from `flightdelays.csv` into R, and use `tidyr` and `dplyr` to convert this data into a tidy/tall format with names and complete data for all columns. Your final data frame should have `City`, `On_Time_Flights` and `Delayed_Flights` as columns (the exact names are up to you). In addition to `pivot_longer`, `pivot_wider` and `rename`, you might find the `tidyr` function `fill` helpful for completing this task efficiently. Although this is a small dataset that you could easily reshape by hand, you should solve this problem using tidyverse functions that do the work for you.

Problem 2: Take the data-frame that you tidied and cleaned in Problem 1 and create additional columns which contain the fraction of on-time and delayed flights at each airport. Then create a Cleveland Multiway Dot Plot (see this tutorial page for a description for how) to visualize the difference in flight delays between the two airlines at each city in the dataset. Compare the airlines and airports using the dot-plot- what are your conclusions?

Optional: If you want to make a fancier visualization consider adding text labels containing the airline names above the dots using `geom_text` and `position = position_nudge(...)` with appropriate arguments.

Part 2: Mixed Drink Recipes

In the second part of this assignment we will be working with a dataset containing ingredients for different types of mixed drinks. This dataset is untidy and messy- it is in a wide data format and contains some inconsistencies that should be fixed.

Problems

Problem 3: Load the mixed drink recipe dataset into R from the file `MixedDrinkRecipes-prep.csv`, which you can download from my github page by clicking here. The variables `ingredient1` through `ingredient6` list the ingredients of the cocktail listed in the `name` column. Notice that there are many `NA` values in the ingredient columns, indicating that most cocktails have under 6 ingredients.

Tidy this dataset using `pivot_longer` to create a new data frame where each there is a row corresponding to each ingredient of all the cocktails, and an additional variable specifying the "rank" of that cocktail in the original recipe, i.e. it should look like this:

name	category	Ingredient_Rank	Ingredient
Gauguin	Cocktail Classics	1	Light Rum
Gauguin	Cocktail Classics	2	Passion Fruit Syrup
Gauguin	Cocktail Classics	3	Lemon Juice

name	category	Ingredient_Rank	Ingredient
Gauguin	Cocktail Classics	4	Lime Juice
Fort Lauderdale	Cocktail Classics	1	Light Rum

where the data-type of `Ingredient_Rank` is an integer. Hint: Use the `parse_number()` function in mutate after your initial pivot.

Problem 4: Some of the ingredients in the ingredient list have different names, but are nearly the same thing. An example of such a pair is `Lemon Juice` and `Juice of a lemon`, which are considered different ingredients in this dataset, but which perhaps should be treated as the same depending on the analysis you are doing. Make a list of the ingredients appearing in the ingredient list ranked by how commonly they occur along with the number of occurrences, and print the first 10 elements of the list here. Then check more ingredients (I suggest looking at more ingredients and even sorting them alphabetically using `arrange(asc(ingredient))`) and see if you can spot pairs of ingredients that are similar but have different names. Use `if_else`(click here for if_else) or `case_when` in combination with `mutate` to make it so that the pairs of ingredients you found have the same name. You don't have to find all pairs, but find at least 5 pairs of ingredients to rename. Because the purpose of this renaming is to facilitate a hypothetical future analysis, you can choose your own criteria for similarity as long as it is somewhat justifiable.

Notice that there are some ingredients that appear to be two or more ingredients strung together with commas. These would be candidates for more cleaning though this exercise doesn't ask you to fix them.

Problem 5: Some operations are easier to do on `wide` data rather than `tall` data. Find the 10 most common pairs of ingredients occurring in the top 2 ingredients in a recipe. It is much easier to do this with a `wide` dataset, so use `pivot_wider` to change the data so that each row contains all of the ingredients of a single cocktail, just like in the format of the original data-set. Then use `count` on the `1` and `2` columns to determine the most common pairs (see chapter 3 for a refresher on `count`).

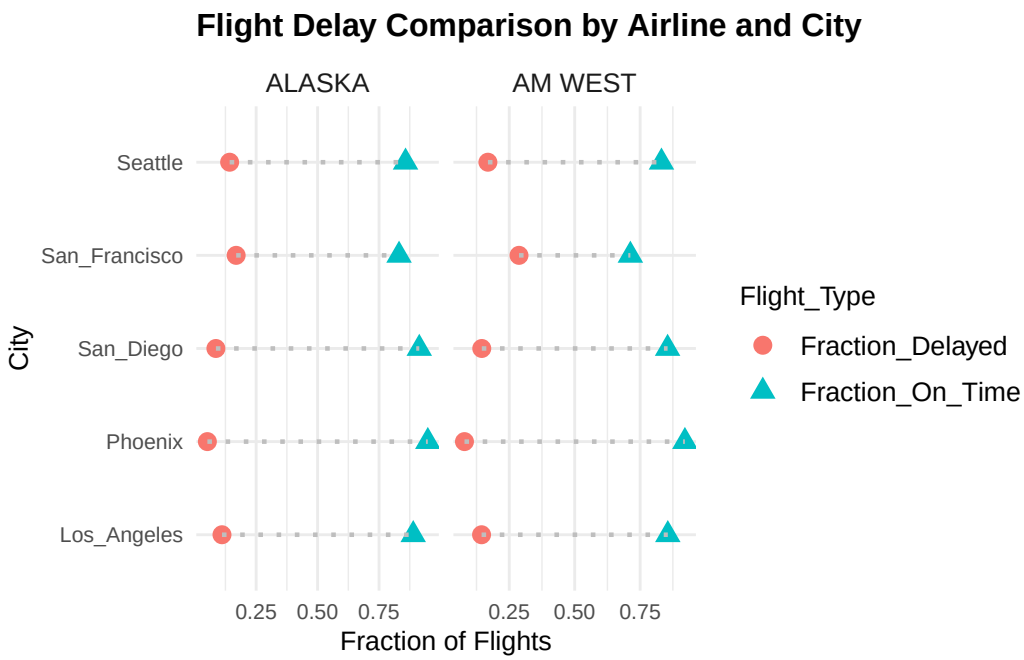
Note: You may be interested to read about the `widyr` package here: widyr page. It is designed to solve problems like this one and uses internal pivot steps to accomplish it so that the final result is tidy. I'm actually unaware of any easy ways of solving problem 5 without pivoting to a wide dataset.

```
library(tidyverse)
library(ggplot2)

flights = read_csv("labData/flightdelays.csv")
knitr::opts_chunk$set(echo = TRUE)
flights_clean <- flights %>%
```

```r
  rename(Airline = ...1, Status = ...2) %>%
  fill(Airline) %>%
  pivot_longer(cols = Los_Angeles:Seattle,
               names_to = "City",
               values_to = "Flights") %>%
  pivot_wider(names_from = Status, values_from = Flights, names_prefix = "Flights_") %>%
  group_by(Airline, City) %>%
  summarise(
    On_Time_Flights = sum(Flights_On_Time, na.rm = TRUE),
    Delayed_Flights = sum(Flights_Delayed, na.rm = TRUE)
  ) %>%
  mutate(
    Total_Flights = On_Time_Flights + Delayed_Flights,
    Fraction_On_Time = On_Time_Flights / Total_Flights,
    Fraction_Delayed = Delayed_Flights / Total_Flights
  ) %>%
  ungroup()

 long_flights_clean <- flights_clean %>%
  pivot_longer(
    cols = c(Fraction_On_Time, Fraction_Delayed),
    names_to = "Flight_Type",
    values_to = "Fraction"
  )
ggplot(long_flights_clean, aes(x = Fraction,
                               y = City, color = Flight_Type, shape = Flight_Type)) +
  geom_point(size = 3) +
  geom_line(aes(group = City), color = "grey", size = .8, linetype = "dotted") +
  facet_wrap(~Airline, scales = "free_x") +
  labs(
    x = "Fraction of Flights",
    y = "City",
    title = "Flight Delay Comparison by Airline and City"
  ) +
  theme_minimal() +
  theme(
    axis.title.x = element_text(size = 10),
    axis.title.y = element_text(size = 10),
    axis.text.x = element_text(size = 8),
    axis.text.y = element_text(size = 8),
    plot.title = element_text(size = 12, face = "bold"),
    strip.text = element_text(size = 10),
```

```
    legend.text = element_text(size = 10),
    legend.title = element_text(size = 10)
  )
```

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