ECO 395M: Exercises 1

Due Friday, Feb 11 at 5 PM

1) Data visualization: flights at ABIA

This problem involves data exploration and data visualization using `ggplot2` and the `tidyverse`, but is entirely open-ended.

Consider the data in ABIA.csv, which contains information on every commercial flight in 2008 that either departed from or landed at Austin-Bergstrom Interational Airport. The variable codebook is as follows:

- Year all 2008
- Month 1-12
- DayofMonth 1-31
- DayOfWeek 1 (Monday) - 7 (Sunday)
- DepTime actual departure time (local, hhmm)
- CRSDepTime scheduled departure time (local, hhmm)
- ArrTime actual arrival time (local, hhmm)
- CRSArrTime scheduled arrival time (local, hhmm)
- UniqueCarrier unique carrier code
- FlightNum flight number
- TailNum plane tail number
- ActualElapsedTime in minutes
- CRSElapsedTime in minutes
- AirTime in minutes
- ArrDelay arrival delay, in minutes
- DepDelay departure delay, in minutes
- Origin origin IATA airport code
- Dest destination IATA airport code
- Distance in miles
- TaxiIn taxi in time, in minutes
- TaxiOut taxi out time in minutes
- Cancelled was the flight cancelled?
- CancellationCode reason for cancellation (A = carrier, B = weather, C = NAS, D = security)
- Diverted 1 = yes, 0 = no
- CarrierDelay in minutes
- WeatherDelay in minutes
- NASDelay in minutes
- SecurityDelay in minutes
- LateAircraftDelay in minutes

Your task is to create a figure, or set of related figures, that tell an interesting story about flights into and out of Austin. You should annotate your figure(s), of course, but strive to make them as easy to understand as possible at a quick glance. (A single figure shouldn't need many, many paragraphs to convey its meaning.) For example, you might consider one of the following questions:

- What is the best time of day to fly to minimize delays, and does this change by airline?
- What is the best time of year to fly to minimize delays, and does this change by destination? (You'd probably want to focus on a handful of popular destinations.)
- How do patterns of flights to different destinations or parts of the country change over the course of the

year?

- What are the bad airports to fly to and does this change by time of year or day?

But anything interesting will fly :-). If you want to try your hand at mapping or looking at geography, you can cross-reference the airport codes here: https://github.com/datasets/airport-codes. Combine this with a mapping package like ggmap, and you should have lots of possibilities!

2) Wrangling the Billboard Top 100

Return to the data in billboard.csv that we briefly analyzed in class, and that contains data on every song to appear on the weekly Billboard Top 100 chart since 1958. Each row of this data corresponds to a single song in a single week. For our purposes, the relevant columns here are:

- performer: who performed the song
- song: the title of the song
- year: year (1958 to 2021)
- week: chart week of that year (1, 2, etc)
- week_position: what position that song occupied that week on the Billboard top 100 chart.

Use your skills in data wrangling and plotting to answer the following three questions.

Part A: Make a table of the top 10 most popular songs since 1958, as measured by the *total number of weeks that a song spent on the Billboard Top 100.* Note that these data end in week 22 of 2021, so the most popular songs of 2021 will not have up-to-the-minute data; please send our apologies to The Weeknd.

Your table should have **10 rows** and **3 columns**: `performer`, `song`, and `count`, where `count` represents the number of weeks that song appeared in the Billboard Top 100 over the period represented by this data set. Make sure the entries are sorted in descending order of the `count` variable, so that the more popular songs appear at the top of the table. Give your table a short caption describing what is shown in the table.
(*Note*: you'll want to use both `performer` and `song` in any `group_by` operations, to account for the fact that multiple unique songs can share the same title.)

Part B: Is the "musical diversity" of the Billboard Top 100 changing over time? Let's find out. We'll measure the musical diversity of given year as *the number of unique songs that appeared in the Billboard Top 100 that year.* Make a line graph that plots this measure of musical diversity over the years. The x axis should show the year, while the y axis should show the number of unique songs appearing at any position on the Billboard Top 100 chart in any week that year. For this part, please filter the data set so that it excludes the years 1958 and 2021, since we do not have complete data on either of those years. Give the figure an informative caption in which you explain what is shown in the figure and comment on any interesting trends you see.

There are number of ways to accomplish the data wrangling here. We offer you two hints on two possibilities:

1) You could use two distinct sets of data-wrangling steps. The first set of steps would get you a table that counts the number of times that a given song appears on the Top 100 in a given year. The second set of steps operate on the result of the first set of steps; it would count the number of unique songs that appeared on the Top 100 in each year, *irrespective of how many times* it had appeared.
2) You could use a single set of data-wrangling steps that combines the `length` and `unique` commands.

Part C: Let's define a "ten-week hit" as a single song that appeared on the Billboard Top 100 for at least ten weeks. There are 19 artists in U.S. musical history since 1958 who have had *at least 30 songs* that were "ten-week hits." Make a bar plot for these 19 artists, showing how many ten-week hits each one had in their musical career. Give the plot an informative caption in which you explain what is shown. Make sure that the individuals names of the artists are readable in your plot, and that they're not all jumbled together. If you find that your plot isn't readable with vertical bars, you can add a `coord_flip()` layer to your plot to make the bars (and labels) run horizontally instead.

Note that by default a bar plot will order the artists in alphabetical order. This is acceptable to turn in. But if you'd like to order them according to some other variable, you can use the `fct_reorder` function,

described in this blog post. This is optional.

3) Wrangling the Olympics

The data in olympics_top20.csv contains information on every Olympic medalist in the top 20 sports by participant count, all the way back to 1896. Use these data to answer the following questions. (The names of the columns should be self-explanatory.)

A) What is the 95th percentile of heights for female competitors across all Athletics events (i.e., track and field)? Note that `sport` is the broad sport (e.g. Athletics) whereas `event` is the specific event (e.g. 100 meter sprint).

B) Which single women's `event` had the greatest variability in competitor's heights across the entire history of the Olympics, as measured by the standard deviation?

C) How has the average age of Olympic swimmers changed over time? Does the trend look different for male swimmers relative to female swimmers? Create a data frame that can allow you to visualize these trends over time, then plot the data with a line graph with separate lines for male and female competitors. Give the plot an informative caption answering the two questions just posed.

4) K-nearest neighbors

The data in sclass.csv contains data on over 29,000 Mercedes S Class vehicles—essentially every such car in this class that was advertised on the secondary automobile market during 2014. For websites like Cars.com or Truecar that aim to provide market-based pricing information to consumers, the Mercedes S class is a notoriously difficult case. There is a huge range of sub-models that are all labeled "S Class,"" from large luxury sedans to high-performance sports cars; one sub-category of S class even serves as the official pace car in Formula 1 Races. Moreover, individual submodels involve cars with many different features. This extreme diversity—unusual for a single model of car—makes it difficult to provide accurate pricing predictions to consumers.

We'll revisit this data set later in the semester when we've got a larger toolkit for building predictive models. For now, let's focus on three variables in particular: - trim: categorical variable for car's trim level, e.g. 350, 63 AMG, etc. The trim is like a sub-model designation.
- mileage: mileage on the car - price: the sales price in dollars of the car

Your goal is to use K-nearest neighbors to build a predictive model for price, given mileage, separately for each of two trim levels: 350 and 65 AMG. (There are lots of other trim levels that you'll be ignoring for this question.) That is, you'll be treating the 350's and the 65 AMG's as two separate data sets. (Recall the `filter` command.)

For each of these two trim levels: 1) Split the data into a training and a testing set.
2) Run K-nearest-neighbors, for many different values of K, starting at K=2 and going as high as you need to. For each value of K, fit the model to the training set and make predictions on your test set. 3) Calculate the out-of-sample root mean-squared error (RMSE) for each value of K.

For each trim, make a plot of RMSE versus K, so that we can see where it bottoms out. Then for the optimal value of K, show a plot of the fitted model, i.e. predictions vs. x. (Again, separately for each of the two trim levels.)

Which trim yields a larger optimal value of K? Why do you think this is?