DATA607 - Project 2

Data Transformation

Donald Butler

10/03/2021

Introduction

Choose any three of the "wide" datasets identified in the Week 6 Discussion items.

Load required R Libraries

```
library(tidyverse)
```

Aruba Weather

The Aruba weather dataset may not be very interesting from a data analysis point of view, but it came to mind when looking for a "wide" dataset. When my family was planning a vacation to Aruba, we wanted to find the best time of year to go, so we considered this weather data. We've been to Aruba three times and always go in the first week of December.

Import raw data

```
aw_file = 'https://raw.githubusercontent.com/dab31415/DATA607/main/Projects/Project_2/ArubaWeather.csv'
aw_raw <-read_csv(aw_file,show_col_types = FALSE)
names(aw_raw)[1] <- 'weather_attr'
aw_raw
```

```
## # A tibble: 4 x 13
##   weather_attr   Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov
##   <chr>        <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 High          86    87    88    89    90    90    90    91    91    90    89
## 2 Low           76    76    77    78    80    80    80    80    80    80    79
## 3 Daylight      11.5  11.5  12    12.5  12.5  13    13    12.5  12    12    11.5
## 4 Rainfall       1.6   0.8   0.3   0.5   0.6   0.7   1.3   1     1.8   3.1   3.7
## # ... with 1 more variable: Dec <dbl>
```

Tidy Dataset

We will tidy the raw dataset by performing the following steps.
1. Pivot on the month columns creating a new row for each month.
2. Pivot on the weather_attr column creating a new statistic for each attribute.

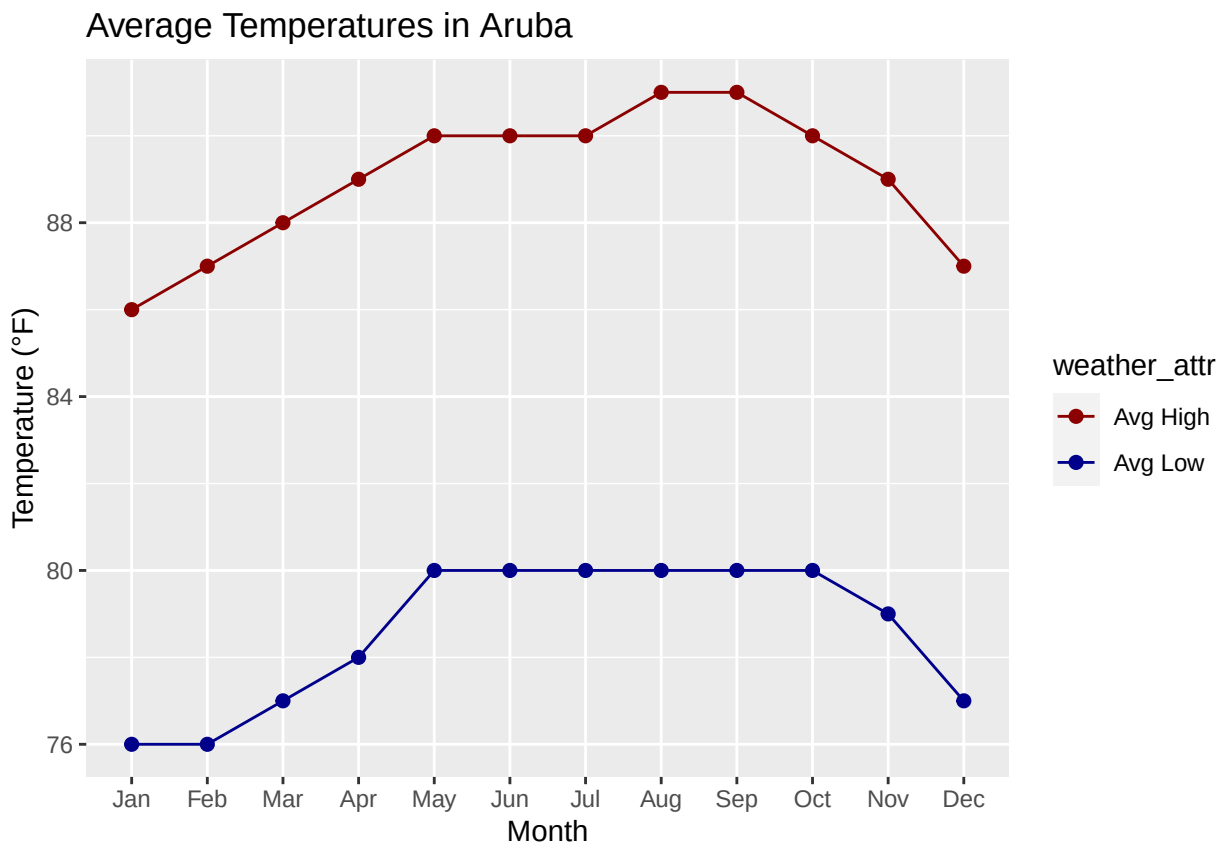
To prevent ggplot from ordering the month column alphabetically, we will specify the levels as a factor.

```
aw_tidy <- aw_raw %>%
  pivot_longer(-weather_attr, names_to = 'month_name', values_to = 'weather_value') %>%
  pivot_wider(names_from = weather_attr, values_from = weather_value)

names(aw_tidy) <- c('month_name','high_temp','low_temp','daylight','rainfall')

# Specify month as an ordered factor for plotting
aw_tidy$month_name <- factor(aw_tidy$month_name, levels = month.abb)

aw_tidy
```

```
## # A tibble: 12 x 5
##    month_name high_temp low_temp daylight rainfall
##    <fct>          <dbl>    <dbl>    <dbl>    <dbl>
##  1 Jan               86       76     11.5      1.6
##  2 Feb               87       76     11.5      0.8
##  3 Mar               88       77     12        0.3
##  4 Apr               89       78     12.5      0.5
##  5 May               90       80     12.5      0.6
##  6 Jun               90       80     13        0.7
##  7 Jul               90       80     13        1.3
##  8 Aug               91       80     12.5      1
##  9 Sep               91       80     12        1.8
## 10 Oct               90       80     12        3.1
## 11 Nov               89       79     11.5      3.7
## 12 Dec               87       77     11.5      3.2
```

Average Temperatures

```
aw_tidy %>%
  pivot_longer(-month_name, names_to = 'weather_attr', values_to = 'weather_value') %>%
  filter(weather_attr %in% c('high_temp','low_temp')) %>%
  ggplot(aes(x = month_name, y = weather_value, group = weather_attr)) +
  geom_line(aes(color = weather_attr)) +
  geom_point(aes(color = weather_attr), size = 2) +
  ggtitle('Average Temperatures in Aruba') +
  xlab('Month') + ylab('Temperature (°F)') +
  scale_color_manual(labels = c('Avg High','Avg Low'), values = c('darkred','darkblue'))
```



Customer Churn

Import raw data

```
cc_file = 'https://raw.githubusercontent.com/dab31415/DATA607/main/Projects/Project_2/CustomerChurn.csv
cc_raw <-read_csv(cc_file,show_col_types = FALSE)
cc_raw
```

```
## # A tibble: 8 x 14
##   Division Description   Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep
##   <chr>    <chr>       <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 A        Gained         70    80   100   110    70    45    50    99   112
## 2 <NA>     Lost            0   -90   -30   -45   -95   -33  -110   -34   -34
## 3 B        Gained         80    80    90   120   100   119    75   119    90
## 4 <NA>     Lost            0   -15   -30   -25   -50   -77   -45   -77   -30
## 5 C        Gained         60    85    80    90   120    45    75    45    80
## 6 <NA>     Lost            0   -45   -27   -17   -33   -80   -45   -80   -27
## 7 Total    Gained        210   245   270   320   290   209   200   263   282
## 8 <NA>     Lost            0  -150   -87   -87  -178  -190  -200  -191   -91
## # ... with 3 more variables: Oct <dbl>, Nov <dbl>, Dec <dbl>
```

Tidy Dataset

We will tidy the raw dataset by performing the following steps.
1. Fill Division column down to update blank cells in the original dataset with the value from the row above.
2. Pivot on the month columns creating a new row for each month.
3. Pivot on the Description column creating a new statistic for the number of customers gained and lost each month.
4. Calculate and append the Net statistic as the sum of customers gained and lost in the month.
5. Calculate and append the Total statistic as the cumulative total number of customers through the month. Note: the dataset doesn't include the number of customers prior to January, and is assumed to be zero for chun calculations.
6. Calculate and append the Churn statistic as the number of customers lost in the month divided by the prior month's Total customers.

To prevent ggplot from ordering the month column alphabetically, we will specify the levels as a factor.

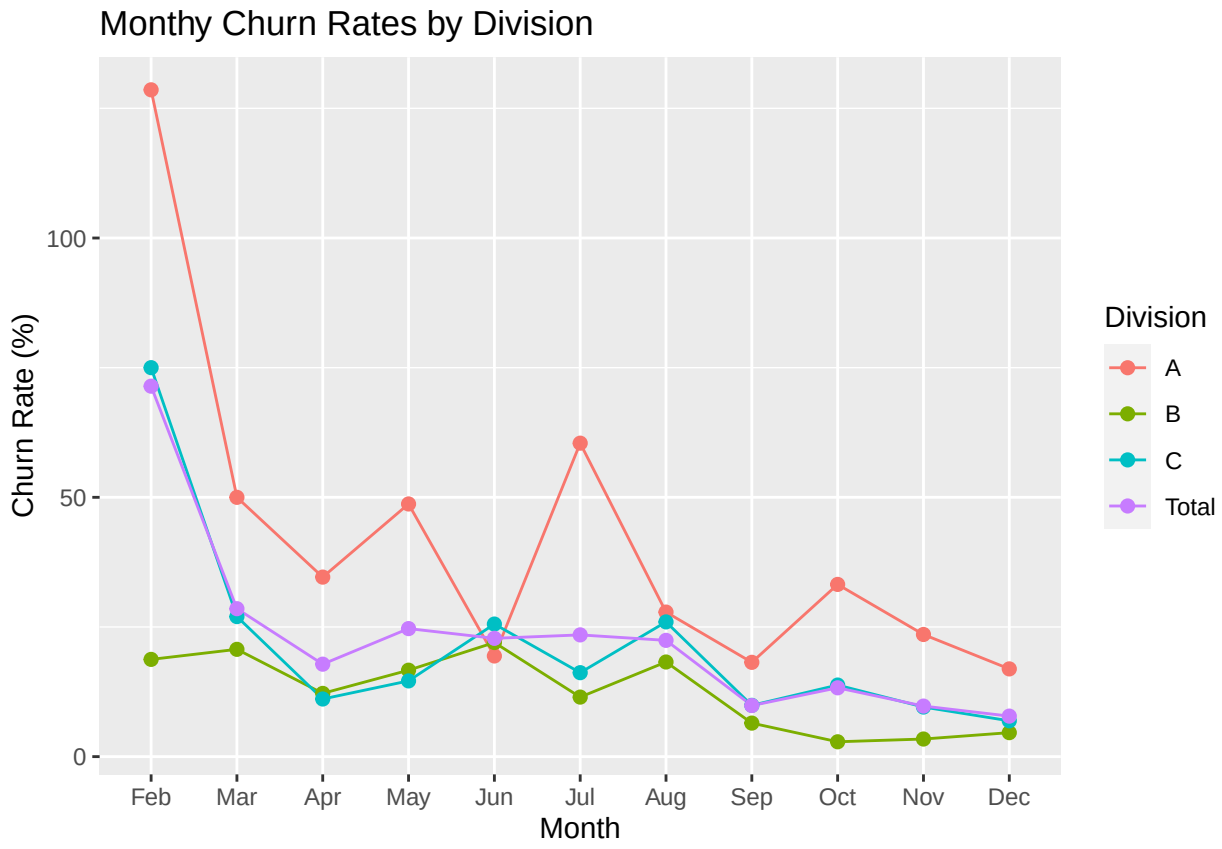
```
cc_tidy <- cc_raw %>%
  fill(Division, .direction = 'down') %>%
  pivot_longer(-c('Division','Description'),names_to = 'month_name', values_to = 'customers') %>%
  pivot_wider(names_from = Description, values_from = 'customers') %>%
  mutate(Net = Gained + Lost) %>%
  group_by(Division) %>%
  mutate(Total = cumsum(Net),
         # (Total - Net) = Prior Month's Total Customers
         Churn = 100 * (-1 * Lost) / (Total - Net))

# Specify month as an ordered factor for plotting
cc_tidy$month_name <- factor(cc_tidy$month_name, levels = month.abb)

cc_tidy
```

```
## # A tibble: 48 x 7
## # Groups:   Division [4]
##    Division month_name Gained  Lost   Net Total Churn
##    <chr>    <fct>       <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 A        Jan            70     0    70    70 NaN
##  2 A        Feb            80   -90   -10    60 129.
##  3 A        Mar           100   -30    70   130  50
##  4 A        Apr           110   -45    65   195  34.6
##  5 A        May            70   -95   -25   170  48.7
##  6 A        Jun            45   -33    12   182  19.4
##  7 A        Jul            50  -110   -60   122  60.4
##  8 A        Aug            99   -34    65   187  27.9
##  9 A        Sep           112   -34    78   265  18.2
## 10 A        Oct            99   -88    11   276  33.2
## # ... with 38 more rows
```

Plotting Churn Rates

```
cc_tidy %>%
  filter(!is.nan(Churn)) %>%
  ggplot(aes(x = month_name, y = Churn, group = Division)) +
```

```
  geom_line(aes(color = Division)) +
  geom_point(aes(color = Division), size = 2) +
  ggtitle('Monthy Churn Rates by Division') +
  xlab('Month') + ylab('Churn Rate (%)')
```



Analysis

The churn rates as calculated would be different if there were customers from the prior year. Division A has the highest monthly churn rate in nearly every month, and division B has the lowest in nearly all months.

Student Testing

Import raw data

```
st_file = 'https://raw.githubusercontent.com/dab31415/DATA607/main/Projects/Project_2/StudentTesting.csv
st_raw <-read_csv(st_file,show_col_types = FALSE)
st_raw
```

```
## # A tibble: 11 x 10
##    Student  Test1 TimeStudiedTest1 Test2 TimeStudiedTest2 Test3 TimeStudiedTest3
##    <chr>    <dbl>            <dbl> <dbl>            <dbl> <dbl>            <dbl>
```

```
##  1 Bob         95              45    88              40    92              50
##  2 John        85              35    60               8    75              10
##  3 Sam         78              15    75              16    80              17
##  4 Jenna       92              60    94              65    84              60
##  5 Sara        97              40    98              50    95              45
##  6 Jacob       50               5    40               2    NA              NA
##  7 Melinda     NA              NA    90              47    92              55
##  8 Billy       78              15    80              25    81              36
##  9 Kayla      100              40   100              40   100              45
## 10 Nick        90              35    94              32    94              30
## 11 Nicolete    75              20    80              20    85              23
## # ... with 3 more variables: Test4 <dbl>, TimeStudiedTest4 <dbl>, Gender <chr>
```

Tidy Dataset

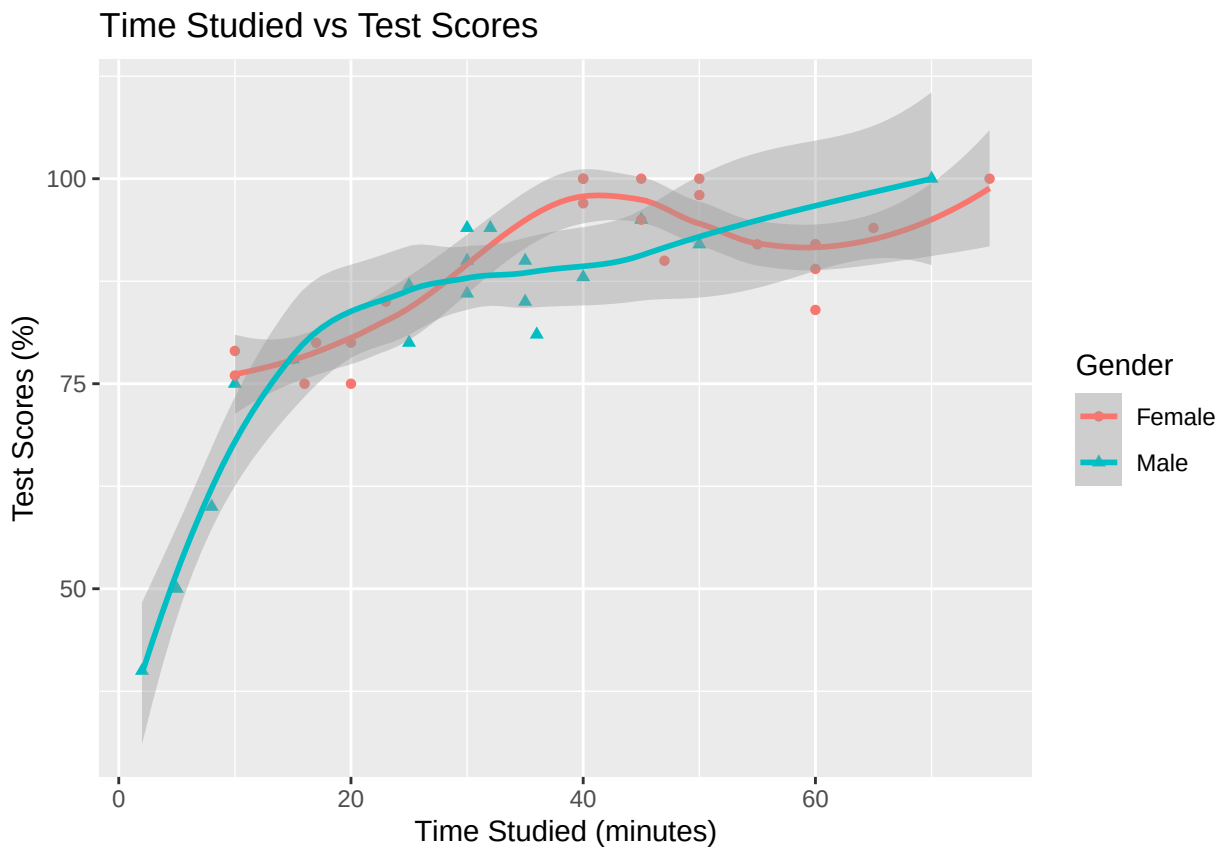
First update the column names for test scores so we have a value to pivot on. Then pivot on the Score and Time Studied columns to generate a row for each test number.

```
names(st_raw)[2] <- 'ScoreTest1'
names(st_raw)[4] <- 'ScoreTest2'
names(st_raw)[6] <- 'ScoreTest3'
names(st_raw)[8] <- 'ScoreTest4'

st_tidy <- st_raw %>%
  pivot_longer(-c(Student,Gender), names_to = c('.value','TestNum'), names_pattern = '(Score|TimeStudie

st_tidy
```

```
## # A tibble: 44 x 5
##    Student Gender TestNum Score TimeStudied
##    <chr>   <chr>  <chr>   <dbl>       <dbl>
##  1 Bob     Male   1          95          45
##  2 Bob     Male   2          88          40
##  3 Bob     Male   3          92          50
##  4 Bob     Male   4         100          70
##  5 John    Male   1          85          35
##  6 John    Male   2          60           8
##  7 John    Male   3          75          10
##  8 John    Male   4          87          25
##  9 Sam     Female 1          78          15
## 10 Sam     Female 2          75          16
## # ... with 34 more rows
```

Plotting Test Scores vs Study Time

```
st_tidy %>% ggplot(aes(x = TimeStudied, y = Score, shape = Gender, color = Gender)) +
  geom_point() +
  geom_smooth() +
  ggtitle('Time Studied vs Test Scores') +
  xlab('Time Studied (minutes)') + ylab('Test Scores (%)')
```


Time Studied vs Test Scores
Test Scores (%)
100
75
50
0
20
40
60
Time Studied (minutes)
Gender
Female
Male