Homework Answer - Week 3

Vikram Viswanath

2025-09-15

Preface

Starlink is a satellite network designed to provide internet to remote and underserved areas. Operated by the company SpaceX, it is comprised of a “constellation” of satellites orbiting the Earth at lower altitude than conventional internet satellites, removing the need for large infrastructure on the ground to receive the signal (CNET, 2022). In Jul. 2023, there were over 4,500 Starlink satellites orbiting the planet — each about the size of a sofa — with plans of bringing this number up to 42,000 in the coming years (The New York Times, 2023).

In 2021, the Brazilian government started a dialogue with SpaceX to increase internet provision in remote areas of Brazil, particularly in the Amazon region (Business Insider, 2021). SpaceX was eventually authorized to provide Starlink internet in 2022 (Valor International, 2022).

In this assignment, we will use R to investigate whether the plan of increasing access in remote areas of Brazil is coming along or not. The goal is to help you gain more familiarity with data import, tidy data, and relational data. In this homework we will continue providing some code snippets to serve as “scaffolding” to help guide you through each step. Over the course of the semester we will provide less scaffolding and more open-ended questions. As always, please come to office hours and reach out to your teaching staff if you have any questions.

We will work with two data sets:

1. **Internet access in Brazil (2022-2024).xlsx**, constructed with data found from the Brazilian regulatory agency for telecommunications (source). It measures Internet access by counting the number of fixed connections to the Internet for each month from Jan. 2022 to Dec. 2024. Counts are broken down by state, provider, type of area within the state, and whether the area is considered remote or not.
2. **brazil_regions.csv**, which identifies which of the 5 regions each of the 27 Brazilian states is located in. In this data set, each column represents a state, and each row, a region. A cell containing the number 1 indicates that the state is located in that region; conversely, the number 0 means that the state is not located in that region.

1. Import `Internet access in Brazil (2022-2024).xlsx` and assign it to a new object named `access`. You may need to specify the range of cells in the spreadsheet that should be imported. We'll consider the level of observation to be a state-year-month.

Based on what we saw in class, is this data frame "tidy"? Why or why not? Also, look at the column types (character, double, etc.); do they seem appropriate given the type of information contained in each one?

```r
access <- read_excel("Internet access in Brazil (2022-2024).xlsx", skip = 5)

glimpse(access)
```

```
Rows: 223
Columns: 40
$ state       <chr> "AC", "AC", "AC", "AC", "AC", "AC", "AC", "AC", "AC", "AC"~
$ provider    <chr> "Other", "Other", "Other", "Other", "Other", "Starlink", "~
$ type_area   <chr> "Suburban", "Suburban", "Rural", "Rural", "Urban", "Suburb~
$ remote_area <lgl> FALSE, TRUE, FALSE, TRUE, FALSE, FALSE, TRUE, FALSE, TRUE,~
$ `202201`    <dbl> 3190, 10538, 2640, 787, 85151, NA, NA, NA, NA, NA, 5999, 2~
$ `202202`    <dbl> 3205, 10496, 2426, 867, 85982, NA, NA, NA, NA, NA, 3140, 2~
$ `202203`    <dbl> 3340, 11639, 2713, 872, 89015, NA, NA, NA, NA, NA, 5173, 2~
$ `202204`    <dbl> 3331, 12187, 4569, 846, 90127, NA, NA, NA, NA, NA, 5122, 2~
$ `202205`    <dbl> 3295, 11662, 4472, 892, 88254, NA, NA, NA, NA, NA, 4598, 2~
$ `202206`    <dbl> 3297, 12752, 4473, 906, 88780, NA, NA, NA, NA, NA, 4850, 2~
$ `202207`    <dbl> 3306, 13847, 4470, 869, 89827, NA, NA, NA, NA, NA, 5069, 2~
$ `202208`    <dbl> 3383, 14900, 4543, 893, 91721, NA, NA, NA, NA, NA, 5089, 2~
$ `202209`    <dbl> 3321, 15292, 4362, 855, 92230, NA, NA, NA, NA, NA, 5077, 3~
$ `202210`    <dbl> 3439, 15762, 4408, 886, 91469, NA, NA, NA, NA, NA, 5312, 3~
$ `202211`    <dbl> 3367, 16213, 2348, 919, 93529, NA, NA, NA, NA, NA, 4805, 3~
$ `202212`    <dbl> 3417, 16761, 2429, 906, 92413, NA, NA, NA, NA, NA, 4525, 3~
$ `202301`    <dbl> 3378, 5375, 2492, 891, 94708, 0, 0, 0, 0, 0, 4814, 30458, ~
$ `202302`    <dbl> 3361, 5317, 2538, 876, 95375, 1, 14, 7, 20, 26, 3380, 2992~
$ `202303`    <dbl> 3276, 4944, 2403, 853, 95399, 4, 22, 10, 32, 41, 4189, 295~
$ `202304`    <dbl> 3188, 4747, 2302, 802, 96228, 12, 26, 16, 36, 43, 4043, 27~
$ `202305`    <dbl> 3181, 4714, 2323, 768, 97003, 54, 108, 62, 141, 184, 4196,~
$ `202306`    <dbl> 3136, 4629, 2328, 743, 98336, 84, 150, 81, 250, 279, 5743,~
$ `202307`    <dbl> 3092, 5564, 2301, 712, 98642, 126, 219, 129, 332, 341, 609~
$ `202308`    <dbl> 3081, 5505, 2271, 681, 99147, 155, 267, 182, 399, 394, 650~
$ `202309`    <dbl> 3011, 5448, 2240, 651, 99298, 165, 288, 214, 415, 365, 604~
$ `202310`    <dbl> 3010, 5420, 2296, 630, 100337, 170, 301, 226, 433, 402, 68~
$ `202311`    <dbl> 2956, 5283, 2358, 580, 100379, 178, 326, 248, 431, 417, 65~
$ `202312`    <dbl> 2939, 5264, 2372, 655, 101094, 189, 352, 279, 449, 440, 64~
$ `202401`    <dbl> 2424, 4714, 1764, 434, 55597, 218, 386, 312, 478, 490, 748~
$ `202402`    <dbl> 2497, 4891, 2816, 676, 56263, 236, 414, 337, 488, 517, 606~
$ `202403`    <dbl> 2531, 4929, 2845, 709, 56836, 259, 460, 357, 517, 537, 617~
$ `202404`    <dbl> 2585, 4954, 2996, 763, 57451, 319, 533, 409, 649, 596, 635~
$ `202405`    <dbl> 2665, 4957, 3790, 745, 59116, 515, 773, 624, 931, 864, 639~
$ `202406`    <dbl> 2700, 4992, 3857, 737, 59556, 601, 901, 733, 1046, 950, 67~
$ `202407`    <dbl> 2872, 5970, 4236, 892, 104076, 600, 926, 770, 1052, 955, 7~
$ `202408`    <dbl> 2894, 5981, 4230, 833, 104420, 657, 1019, 892, 1126, 1079,~
$ `202409`    <dbl> 2951, 5253, 4333, 830, 106469, 780, 1217, 1077, 1258, 1309~
$ `202410`    <dbl> 3035, 5226, 3828, 1570, 106116, 872, 1412, 1262, 1366, 148~
$ `202411`    <dbl> 2685, 5246, 4486, 1582, 104386, 906, 1423, 1352, 1417, 155~
$ `202412`    <dbl> 2704, 5736, 4598, 1602, 104840, 978, 1512, 1458, 1506, 164~
```

This data set is not tidy because it has multiple columns for year-month combinations (like "202201", "202202", etc.) which should be pivoted into rows.

No, the column types do not seem appropriate for the data they contain.

2. To start tidying the data, focus first on the columns whose names appear to be year-month combinations (in YYYYMM format – e.g., "202202" is Feb. 2022). Use `pivot_longer` to transform the data frame, and then create two new columns: `year`, containing the year of the observation, and `month`, containing the month. Assign the result to a new object named `access_long`. Print the data frame `access_long`.

```r
access_long <- access |>
  pivot_longer(
    cols = matches("^[0-9]{6}$"),
    names_to = "year_month",
    values_to = "connections"
  ) |>
  mutate(
    year = as.integer(substr(year_month, 1, 4)),
    month = as.integer(substr(year_month, 5, 6))
  )

access_long |>
  select(state, provider, remote_area, year, month, connections) |>
  slice_head(n = 15) |>
  kable(digits = 0, col.names =
  c("State", "Provider", "Remote Area", "Year", "Month", "Connections"))
```

State	Provider	Remote Area	Year	Month	Connections
AC	Other	FALSE	2022	1	3190
AC	Other	FALSE	2022	2	3205
AC	Other	FALSE	2022	3	3340
AC	Other	FALSE	2022	4	3331
AC	Other	FALSE	2022	5	3295
AC	Other	FALSE	2022	6	3297
AC	Other	FALSE	2022	7	3306
AC	Other	FALSE	2022	8	3383
AC	Other	FALSE	2022	9	3321
AC	Other	FALSE	2022	10	3439
AC	Other	FALSE	2022	11	3367
AC	Other	FALSE	2022	12	3417
AC	Other	FALSE	2023	1	3378
AC	Other	FALSE	2023	2	3361
AC	Other	FALSE	2023	3	3276

3. We'll ignore the distinction between area types (rural, suburban and urban), so aggregate the data frame `access_long` up to the `state-provider-remote_area-year-month` level by grouping and calculating the total number of connections (watch out for missing data!). Ungroup the data frame after summarizing. You do not need to retain `year_month`. Assign the resulting data frame to `access_aggreg` and print it.

```r
access_aggreg <- access_long |>
  group_by(state, provider, remote_area, year, month) |>
  summarize(connections = sum(connections, na.rm = TRUE), .groups = "drop")

access_aggreg |>
  slice_head(n = 15) |>
  kable(digits = 0, col.names =
  c("State", "Provider", "Remote Area", "Year", "Month", "Connections"))
```

State	Provider	Remote Area	Year	Month	Connections
AC	Other	FALSE	2022	1	90981
AC	Other	FALSE	2022	2	91613
AC	Other	FALSE	2022	3	95068
AC	Other	FALSE	2022	4	98027
AC	Other	FALSE	2022	5	96021
AC	Other	FALSE	2022	6	96550
AC	Other	FALSE	2022	7	97603
AC	Other	FALSE	2022	8	99647
AC	Other	FALSE	2022	9	99913
AC	Other	FALSE	2022	10	99316
AC	Other	FALSE	2022	11	99244
AC	Other	FALSE	2022	12	98259
AC	Other	FALSE	2023	1	100578
AC	Other	FALSE	2023	2	101274
AC	Other	FALSE	2023	3	101078

4. Finish tidying this data set by using `pivot_wider` so that `access_aggreg` shows for any given state, year, and month: (i) the number of Starlink connections in remote areas; (ii) Starlink connections in non-remote areas; (iii) connections to other providers in remote areas; and (iv) connections to other providers in non-remote areas. Assign the resulting data frame to `access_wide` and print it.

Note the new columns created have non-intuitive names. Use `rename` to give them names you think are intuitive for you and anyone who might look at your code.

```r
access_wide <- access_aggreg |>
  pivot_wider(
    names_from = c(provider, remote_area),
    values_from = connections,
    values_fill = 0
  ) |>
  rename(
    `Starlink-R` = `Starlink_TRUE`,
    `Starlink-NR` = `Starlink_FALSE`,
    `Providers-R` = `Other_TRUE`,
    `Providers-NR` = `Other_FALSE`
  )

access_wide |>
  slice_head(n = 10) |>
  kable(digits = 0, col.names =
  c("State", "Year", "Month", "Starlink-R",
  "Starlink-NR", "Providers-R", "Providers-NR"))
```

State	Year	Month	Starlink–R	Starlink–NR	Providers–R	Providers-NR
AC	2022	1	90981	11325	0	0
AC	2022	2	91613	11363	0	0
AC	2022	3	95068	12511	0	0
AC	2022	4	98027	13033	0	0
AC	2022	5	96021	12554	0	0
AC	2022	6	96550	13658	0	0
AC	2022	7	97603	14716	0	0
AC	2022	8	99647	15793	0	0
AC	2022	9	99913	16147	0	0
AC	2022	10	99316	16648	0	0

5. We now have the number of connections of each type by provider in each state at any given year and month. However, we still need to find out which states are located in the Amazon area.

Import `brazil_regions.csv` and assign it to a new object named `regions`. The Brazilian Amazon is mostly contained in the North region, so *later* we'll want to single out states that are located there. With that in mind, is this data frame tidy? If not, why is that a problem, given what we want to do with this data?

```r
regions <- read_csv("brazil_regions.csv")

regions
```

```
# A tibble: 5 x 28
  region    AC    AL    AM    AP    BA    CE    DF    ES    GO    MA    MG    MS
  <chr>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1 North      1     0     1     1     0     0     0     0     0     0     0     0
2 North~     0     1     0     0     1     1     0     0     0     1     0     0
3 Centr~     0     0     0     0     0     0     1     0     1     0     0     1
4 South~     0     0     0     0     0     0     0     1     0     0     1     0
5 South      0     0     0     0     0     0     0     0     0     0     0     0
# i 15 more variables: MT <dbl>, PA <dbl>, PB <dbl>, PE <dbl>, PI <dbl>,
#   PR <dbl>, RJ <dbl>, RN <dbl>, RO <dbl>, RR <dbl>, RS <dbl>, SC <dbl>,
#   SE <dbl>, SP <dbl>, TO <dbl>
```

No, the data frame is not tidy, because it has regions as rows and states as columns, This makes the process to filter for specific regions or linking it with other datasets that have states as observations making it very hard to manipulate the data.

6. Use `pivot_longer` to tidy up `regions` and assign it to `states`. Print the resulting data frame. Then `filter` for rows in `states` corresponding to the North region, `pull` the states' names stored in `state`, and assign it to a new object named `north_states`. What are these states?

```r
states <- regions |>
  pivot_longer(
    cols = -region,
    names_to = "state",
    values_to = "in_region"
  )


states |>
  slice_head(n = 15) |>
  kable(digits = 0, col.names =
  c("Region", "State", "In Region"))
```

Region	State	In Region
North	AC	1
North	AL	0
North	AM	1
North	AP	1
North	BA	0
North	CE	0
North	DF	0
North	ES	0
North	GO	0
North	MA	0
North	MG	0
North	MS	0
North	MT	0
North	PA	1
North	PB	0

```r
north_states <- states |>
  filter(region == "North", in_region == 1) |>
  pull(state)

north_states
```

```
[1] "AC" "AM" "AP" "PA" "RO" "RR" "TO"
```

The states located in the North region are: AC, AM, AP, PA, RO, RR, TO.

7. Let's go back to the internet data. Filter `access_wide` so that only North states remain in the data frame, and assign the resulting data frame to `access_tidy`. When was Starlink first introduced in each state? You can present the answer to this question in a small data frame with 7 rows.

```r
access_tidy <- access_wide |>
  filter(state %in% north_states)

starlink_intro <- access_tidy |>
  filter(`Starlink-R` > 0 | `Starlink-NR` > 0) |>
  arrange(state, year, month) |>
  group_by(state) |>
  slice_head(n = 1) |>
  select(state, year, month)

starlink_intro |>
  kable(digits = 0, col.names =
  c("State", "Year", "Month"))
```

State	Year	Month
AC	2023	2
AM	2022	8
AP	2022	11
PA	2022	10
RO	2022	11
RR	2022	10
TO	2022	9

8. Using only 2024 data, and separately for each state, calculate Starlink's share of remote connections as one column and the share of non-remote connections as another column. Would you say Starlink's entry in the Brazilian internet market is having a larger impact in remote or in non-remote areas of the Amazon? Recalling what you read in the introduction, is this result consistent with SpaceX's expectations for Starlink?

```r
shares_2024 <- access_tidy |>
  filter(year == 2024) |>
  mutate(
    remote_share = `Starlink-R` / (`Starlink-R` + `Providers-R`),
    nonremote_share = `Starlink-NR` / (`Starlink-NR` + `Providers-NR`)
  ) |>
  group_by(state) |>
  summarize(
    avg_remote_share = mean(remote_share, na.rm = TRUE),
    avg_nonremote_share = mean(nonremote_share, na.rm = TRUE),
    .groups = "drop"
  )

shares_2024 |>
  kable(digits = 3, col.names =
  c("State", "Avg Remote Share", "Avg Non-Remote Share"))
```

State	Avg Remote Share	Avg Non-Remote Share
AC	0.226	0.025
AM	0.513	0.022
AP	0.126	0.014
PA	0.135	0.018
RO	0.012	0.009
RR	0.369	0.182
TO	0.089	0.024

We see that Starlink's entry in the Brazilian internet market is having a larger impact in remote areas of the Amazon compared to non-remote areas. This result is shows the expectations for SpaceX's Starlink, as the service was aimed at serving remote areas over non-remote areas.