Affordable Healthy Diet Dataset Analysis

```r
library(tidyverse)
library(dplyr)
set.seed(5011)
```

Introduction

So the dataset that we will be using can be found on WorldBank.org; a random slice of which is displayed below.

```r
file_name <- str_subset(dir(),"^F.+[.]csv$")
# this is the generic way of extracting the csv file only by telling the finder that look for
dietDf <- read_csv(file_name,show_col_types = FALSE)
# here we have assigned the dataset to dietdf variable
dietDf |>
    select("TIME_PERIOD",
    "REF_AREA_LABEL",
    "OBS_VALUE","INDICATOR_LABEL") |>
    slice_sample(n=10)
```

```
# A tibble: 10 x 4
   TIME_PERIOD REF_AREA_LABEL     OBS_VALUE INDICATOR_LABEL
         <dbl> <chr>                  <dbl> <chr>
 1        2022 Russian Federation       1.5 Percentage of the population unable~
 2        2022 Hungary                  1.1 Number of people unable to afford a~
 3        2023 Guinea                   6.8 Number of people unable to afford a~
 4        2024 Benin                   53.4 Percentage of the population unable~
 5        2020 Brazil                  40.4 Number of people unable to afford a~
 6        2021 Kenya                   41.3 Number of people unable to afford a~
 7        2020 Japan                    8   Percentage of the population unable~
 8        2020 Lithuania                8.8 Percentage of the population unable~
```

```
 9        2024 South Africa             61.7 Percentage of the population unable~
10        2019 China                   326.  Number of people unable to afford a~
```

```
# with this we have selected a random slice from the dataset.
```

Note that the measurements labels are:

```
# A tibble: 2 x 1
  INDICATOR_LABEL
  <chr>
1 Percentage of the population unable to afford a healthy diet (percent)
2 Number of people unable to afford a healthy diet (million)
```

Proposed questions during EDA

After we receive the dataset, our job will be to determine questions to be answered by this dataset. So, What kind of questions can this data answer?

Usually we have work a company for us to have some questions, that is why we won't go much into the questions in detail, but still some generic questions can be like this -

- Do we know a lot about a healthy, affordable diet?
- Per country, how many people can not afford the least expensive, healthy diet?
- What fraction of that country's population can not afford a healthy diet?
- Is this problem getting worse?
- What about the world as a whole? What fraction of the world's population can not afford a healthy diet?
- On a world scale, is the problem getting worse?

Let's first take care of the NA-values issue.

Tidy-ing

Our next issue is to address the tidiness of the dataset. Can two measurements be in the same column of a tidy dataframe? No!

By that we can say for `country` we had a similar column namely `REF_AREA_LABEL`, and then similarly for `year` we had `TIME_PERIOD`, etc.

In the following chunk, please `select()` only the following columns:

```
- country = REF_AREA_LABEL,
- year = TIME_PERIOD,
- label= INDICATOR_LABEL,
- value = OBS_VALUE
```

- In the same chunk, let's tidy the dataframe using `pivot_wider()` with these arguments:

```
- id_cols = c(country,year),
- names_from = label,
- values_from = value
```

- We make sure that we assign our changes to `dietDf`.
- Below we have just gotten rid of the `redundancy`.

```r
# tidy_diet_data1
dietDf <- dietDf |> select(country = REF_AREA_LABEL,
         year = TIME_PERIOD,
         label= INDICATOR_LABEL,
         value = OBS_VALUE)

dietDf <- pivot_wider(dietDf,
                      id_cols = c(country,year),
                      names_from = label,
                      values_from = value)

dietDf
```

```
# A tibble: 1,202 x 4
   country  year Percentage of the population unable to~1 Number of people una~2
   <chr>   <dbl>                                    <dbl>                  <dbl>
 1 Albania  2017                                     24.3                    0.7
 2 Albania  2018                                     17.6                    0.5
```

```
 3 Albania  2019                                        14.6                  0.4
 4 Albania  2020                                        13.9                  0.4
 5 Albania  2021                                        12.6                  0.4
 6 Albania  2022                                        11.7                  0.3
 7 Algeria  2017                                        18.8                  7.8
 8 Algeria  2018                                        18.1                  7.7
 9 Algeria  2019                                        17.5                  7.6
10 Algeria  2020                                        19.2                  8.5
# i 1,192 more rows
# i abbreviated names:
#   1: `Percentage of the population unable to afford a healthy diet (percent)`,
#   2: `Number of people unable to afford a healthy diet (million)`
```

Great, but more tidying to do! In this chunk let us do just that:

- `mutate()` the column `year` to integer,
- `rename()` the percentage column to `percent`,
- `rename()` the `starts_with("Num")` column to `countPerCountry`
- `mutate()` the `countPerCountry` and multiply by `1e+06`

```
#tidy_diet_data2
dietDf <- dietDf %>%
  mutate(year = as.numeric(year)) %>%
  rename(percent = `Percentage of the population unable to afford a healthy diet (percent)`,
         countPerCountry = `Number of people unable to afford a healthy diet (million)`) %>%
  mutate(countPerCountry = countPerCountry * 1000000)

dietDf
```

```
# A tibble: 1,202 x 4
   country  year percent countPerCountry
   <chr>   <dbl>   <dbl>           <dbl>
 1 Albania  2017    24.3          700000
 2 Albania  2018    17.6          500000
 3 Albania  2019    14.6          400000
 4 Albania  2020    13.9          400000
 5 Albania  2021    12.6          400000
 6 Albania  2022    11.7          300000
 7 Algeria  2017    18.8         7800000
 8 Algeria  2018    18.1         7700000
 9 Algeria  2019    17.5         7600000
10 Algeria  2020    19.2         8500000
# i 1,192 more rows
```

Now, let us explore **dietDf** for tidyness and make any additional changes necessary. After running all chunks, we may may do this from the **console** pane, with **View(dietDf)**. (**Another way of doing that without using the native code-box**)

Now, let's consider NAs in this dataset. Consider these two results:

```
# This counts the total number of missing or N/A values in our target column
sum(is.na(dietDf$countPerCountry))
```

```
[1] 80
```

```
sum(is.na(dietDf$percent)&is.na(dietDf$countPerCountry))
```

```
[1] 0
```

Huh? How can the **countPerCountry** have many NAs, yet the **percent** has none in common with it!? It is clear that the **countPerCountry** values are not missing with malice. If only we can find the population figures for countries per year, we could calculate the missing data! Fortunately, other tables at WorldBank.org contain this information, and we happen to have a second ***.csv** in the archive from that table. In the following chunk, we load this ***.csv** and tidy it. Let us load this file into the tibble **pop**. Also now let's use **pivot_longer()** to create a **year** column, with values from the column headers, and **pop** column, with population figures. We also need to make sure that **pop$year** has a numerical type. Alongside, making sure that **pop$pop** is a numerical measurement, so we will use **str_replace()** to find and replace the "k"s, "M"s, and "B"s. After that, it may be converted using **as.integer()**.

```
# load_and_tidy_pop_data
file_name <- str_subset(dir(),"^pop.+[.]csv$")
pop <- read_csv(file_name,
show_col_types=FALSE)
# please complete this chunk below
pop <- pop %>%
  pivot_longer(cols = - country,
               names_to = "year",
               values_to = "pop") %>%
  mutate(year=as.numeric(year)) %>%
  mutate(pop = str_replace(pop, "M","e6"),
         pop = str_replace(pop, "k", "e3")) %>%

  mutate(pop = as.numeric(pop))
```

```
Warning: There was 1 warning in `mutate()`.
i In argument: `pop = as.numeric(pop)`.
Caused by warning:
! NAs introduced by coercion
```

```r
pop
```

```
# A tibble: 59,297 x 3
   country      year     pop
   <chr>       <dbl>   <dbl>
 1 Afghanistan  1800 3280000
 2 Afghanistan  1801 3280000
 3 Afghanistan  1802 3280000
 4 Afghanistan  1803 3280000
 5 Afghanistan  1804 3280000
 6 Afghanistan  1805 3280000
 7 Afghanistan  1806 3280000
 8 Afghanistan  1807 3280000
 9 Afghanistan  1808 3280000
10 Afghanistan  1809 3280000
# i 59,287 more rows
```

It will be easier to work with one dataframe, rather than 2. In the following chunk, let us use `left_join()` to join `dietDf`(left) with `pop`(right).

```r
#left_join_to_pop_data

dietDf <- left_join(dietDf,pop, by = c("country", "year"))

dietDf
```

```
# A tibble: 1,202 x 5
   country  year percent countPerCountry      pop
   <chr>   <dbl>   <dbl>           <dbl>    <dbl>
 1 Albania  2017    24.3          700000  2900000
 2 Albania  2018    17.6          500000  2890000
 3 Albania  2019    14.6          400000  2890000
 4 Albania  2020    13.9          400000  2870000
 5 Albania  2021    12.6          400000  2850000
 6 Albania  2022    11.7          300000  2830000
 7 Algeria  2017    18.8         7800000 41700000
```

```
 8 Algeria  2018    18.1         7700000 42500000
 9 Algeria  2019    17.5         7600000 43300000
10 Algeria  2020    19.2         8500000 44000000
# i 1,192 more rows
```

Nice work so far! We have all the data we need in one table. Our current goal is to remove as many of the NAs as the data allows. In the following chunk, by using `mutate()` we create a new column `my_count`, found by multiplying `pop` with `percent/100`. Since `pop` also has NAs, this new column will have NAs too.

```r
# calc_my_count_~_pop_data_and_percent

dietDf <- dietDf %>% mutate(my_count = pop * (percent/100))
```

The following chunk gives us hope. Below we have calculated the number of rows where both `my_count` and `countPerCountry` are NAs.

```r
# uncomment when ready
 sum(is.na(dietDf$my_count) & is.na(dietDf$countPerCountry))
```

```
[1] 0
```

Super lucky! There is no overlap between the NAs, and we will be able to replace all the NAs in `countPerCounty` with values from `my_count`. At the end of the chunk, we have again checked for NAs in the amended column `countPerCountry`.

```r
# replace NAs in `countPerCountry`, then check for NAs in same
dietDf <- dietDf %>% mutate(countPerCountry = if_else(is.na(countPerCountry), my_count, coun

sum(is.na(dietDf$countPerCountry))
```

```
[1] 0
```

One more step for NA replacement! Let us amend the column `pop`, which has NAs, with values calculated from `100*countPerCountry` / `percent`. At the end, let's recheck for NAs.

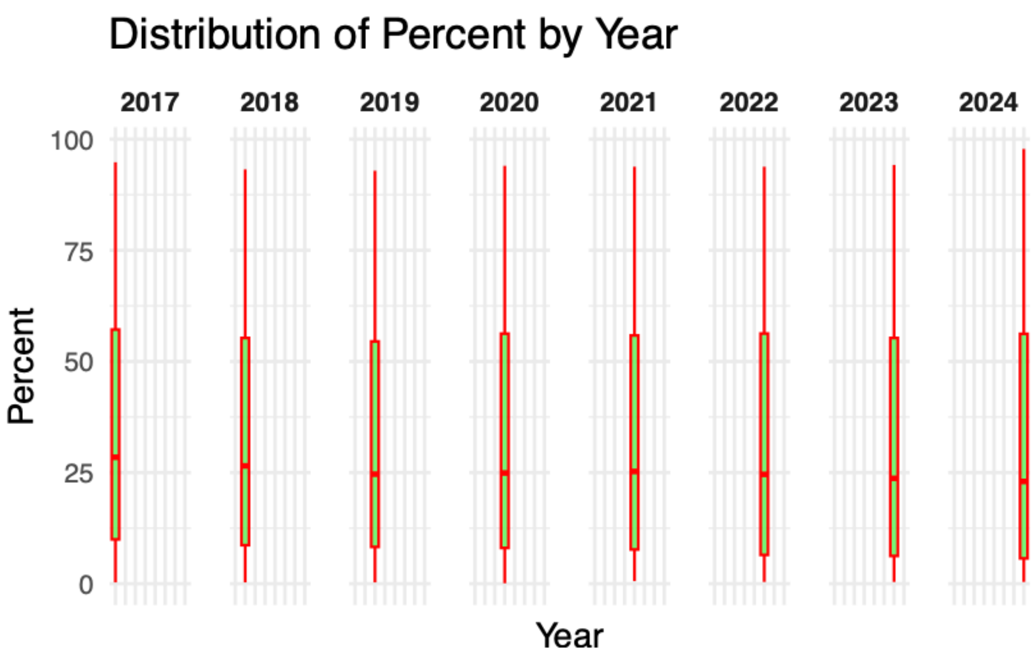
```r
#replace_NAs_in_pop_with_calculated
dietDf <- dietDf %>% mutate(pop = if_else(is.na(pop), (100*countPerCountry/percent),pop))

sum(is.na(dietDf$pop))
```

```
[1] 0
```

Awesome! Now that our data is tidy and wrangled, we may begin our EDA. Below, let us use `facet_wrap(~year,nrow=1)` to display a sequence of vertical boxplots of `percent`, indexed by `year`.

```
# boxplots_facetted_by_year
# assuming your tibble is called pop_dietDf
ggplot(dietDf, aes(x = factor(year), y = percent)) +
  geom_boxplot(fill = "green", color = "red", alpha = 0.5) +
  facet_wrap(~year, nrow = 1) +
  labs(
    title = "Distribution of Percent by Year",
    x = "Year",
    y = "Percent"
  ) +
  theme_minimal(base_size = 13) +
  theme(
    strip.text = element_text(size = 10, face = "bold"),
    axis.text.x = element_blank(),  # This hides x-axis text since facets already label them
    panel.spacing = unit(1, "lines")
  )
```



Below, we have provided a few sentences summarizing our observations about the boxplots.

——>

So it appears that the distribution of percent remain quite consistent across all the years, showing similar statistical values (medians, spreads, etc)

Most countries cluster around moderate percent values, although there are quite a few outliers for each year.

<——>

In the following chunk, We create **`summaryDf`**. This yields per-year results of world population, **`worldpop`**, and the fraction of the globe's population which can not afford the least expensive healthy diet, **`fracWorldPop`**.

```
# More piping to yield per-year results of a world population
summaryDf <- dietDf |>
     group_by(year) |>
      summarize(totalUnhealthyDiet=
       sum(countPerCountry),
        worldpop=sum(pop,na.rm=TRUE),
         fracWorldPop=totalUnhealthyDiet/worldpop
         )
```

Our intention is to have two curves on the sample plot.

While there are many ways to do this, `ggplot2` does this easily if we create an "untidy" data frame, with a single column for all vertical variables, and a new column which explains which measurement it is.

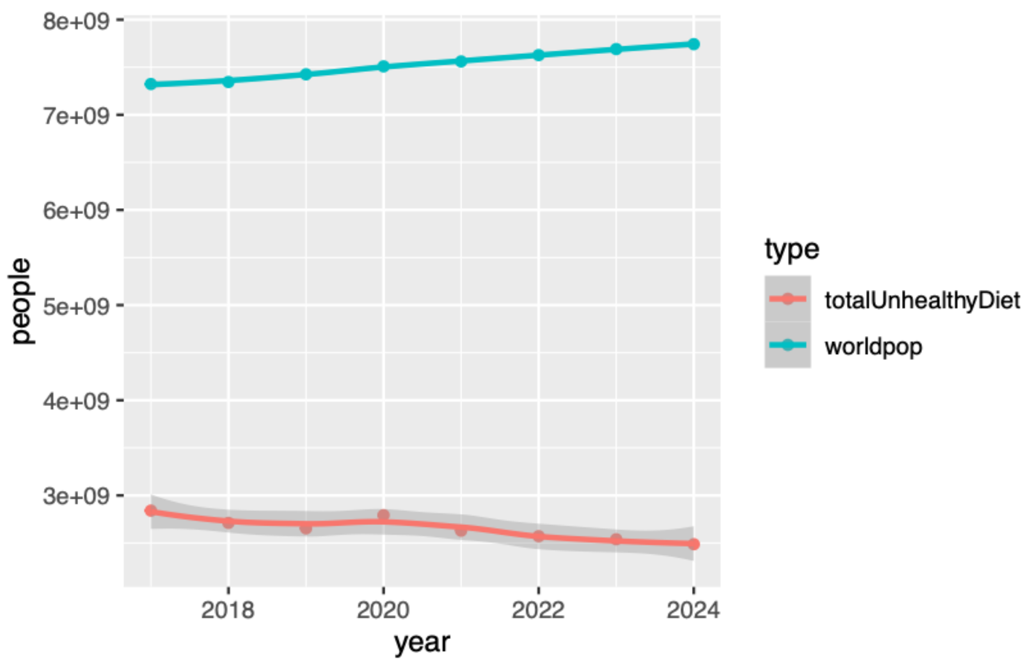
Fortunately, `pivot_longer()` is available to make this easy! Notice that the new column **`type`** is converted to a factor in the chunk below.

`ggplot()` prefers this form for `color`, `fill` too.

```
# uncomment when ready
plot1Df <- summaryDf |>
    select(-fracWorldPop) |>
    pivot_longer(cols=-1,
    names_to="type",
    values_to = "people"
    ) |>
        mutate(type=as.factor(type))
```

Finally! we are ready to create the plot with two, labeled curves. `Ggplot()` makes this a snap! Note that the column **`type`** determines the point and line color, and a legend is generated automatically.

```
# uncomment when ready
plot1Df |>
    ggplot(aes(x=year,y=people,color=type))+
     geom_point() + geom_smooth()
```



Please provide a two sentence comment on the plot above, to let the reader know about important observations.

——>

The Global population has shown a steady growth from 2017 to 2024. However, the total Unhealthy diet gradually declines over the same time period. Suggesting that the percentage of people with unhealthy diet is decreasing graudually.

<——>

For our final chunk, please plot `fracWorldPop` vs. `year`. Choose a vertical axis label:“fraction of people who can’t afford healthy diet” , a horizontal axis label “year”.

```
# plot_fracWorldPop vs. year
```

As always, provide a brief comment on your findings from this plot below.

——>

<——>