PS 138Z - The Politics of Immigration: Section 7

2024-03-7

Introduction

Today, we will continue using the **Mexican Migration Project (MMP) Database** to produce descriptive statistics. Also, we will explore in greater depth the tools offered by the `ggplot2` package to produce customizable plots.

Descriptive statistics: sex and age

To begin, let's load the data using the `read.csv()` function. This time, we will assign the dataset to an object called `mmp.data`.

```
mmp.data <- read.csv("mmp_subset.csv")
```

The `tidyverse` package includes `ggplot2`, `dplyr`, dplyr, and several other useful tools for data analysis. Let's upload `tidyverse` package only. Remember that you can upload a (pre-installed) package with the `library()` function

```
library(tidyverse)
```

```
## Warning: package 'tibble' was built under R version 4.2.3
```

Like the previous exercise, we want to focus on people migrating to the U.S. from Mexico. As before, we can use the variable `us_immigrant` to filter the data we are interested in:

This exercise focuses on people migrating to the U.S. from Mexico. For this, we have created a dummy variable that takes on a value of `1` for people who have migrated at least once into the U.S. and a value of `0` otherwise. Using this variable and the `filter()` function, we can extract the data we need and assign it to a new object called `df.filter`:

```
df.filter <- mmp.data  %>% filter(us_immigrant == 1)
```

Before describing the data, let's check again if `sex` and `age` have missing values coded as `NA`.

```
any(is.na(df.filter$sex))
```

```
## [1] FALSE
```

```
any(is.na(df.filter$age))
```

```
## [1] TRUE
```

We can see that `age` has missing values coded as `NA` but `sex` not. The next step is to find out if there are missing values coded as `8888` or `9999`. Remember that there are multiple ways to do it. One of them is using the `range()` function.

```
range(df.filter$sex)
```

```
## [1] 1 2
```

```
range(df.filter$age, na.rm = TRUE)
```

```
## [1]    5 8888
```

Note that we must use the argument `na.rm = TRUE` when calculating the range of `age` because some values in this variable are coded as `NA`. The variable `sex` ranges from `1` to `2`, which is completely normal (check the codebook). The minimum value of `age` is `5` (a normal value), but the maximum is `8888`. Therefore, we know that some missing values in this variable are coded as `8888`, and we have to do something about it.

Lets replace th values coded as `8888` in `age` by `NA`:

```
df.filter$age[df.filter$age == 8888] <- NA
```

We replaced these values because we know that R interprets `8888` as numerical data (not as actual missing values), which is bad because we do not want these numbers to be considered in our calculations when producing descriptive statistics. After the replacement is done, we can check again the range of `age`:

```
range(df.filter$age, na.rm = TRUE)
```

```
## [1]  5 99
```

Now that we have cleaned the data, we can start producing the other descriptive statistics in addition to the range. Use the `mean()` and `median()` functions to find the number of males and females and the mean and median age. Remember that `age` still has `NA`s, so you should use the argument `na.rm = TRUE` again.

```
# table()
# mean()
# median()
```

We might be interested in calculating the mean `age` by `sex` to determine whether the age distributions of males and females are similar. We can make this calculation with the `group_by()` and the `summarise_at()` functions.

```
df.filter %>%
  group_by(sex) %>%
  summarise_at(vars(age), mean, na.rm = TRUE)
```

```
## # A tibble: 2 x 2
##     sex   age
##   <int> <dbl>
## 1     1  42.5
## 2     2  40.9
```

We can find the mean, median, minimum, and maximum `age` by `sex` simultaneously:

```
df.filter %>%
  group_by(sex) %>%
  summarise_at(vars(age), c(mean, median, min, max), na.rm = TRUE)
```

```
## # A tibble: 2 x 5
##     sex   fn1   fn2   fn3   fn4
##   <int> <dbl> <dbl> <int> <int>
## 1     1  42.5    41     5    99
## 2     2  40.9    39    17    86
```

Plots: sex and age

Now, we will use the `ggplot2` package to produce some histograms showing the distribution of the variable `age`. Let's start by creating a new dataframe that only contains two columns: `sex` and `age`. Remember that

we can do this using the `cbind()` function.

```
df.plot <- as.data.frame(cbind(df.filter$sex, df.filter$age))
```

The next step is to drop any row in this dataframe for which at least one variable (`sex` or `age`) has a missing value (`NA`). We do this using the `na.omit()` function:

```
df.plot <- na.omit(df.plot)
```

Finally, we reassign the variable names, which R modified replace by tow generic names (`V1` and `V2`) when we created the new dataframe.

```
names(df.plot)[1] <- "sex"
names(df.plot)[2] <- "age"
```

The data look exactly how we expect:

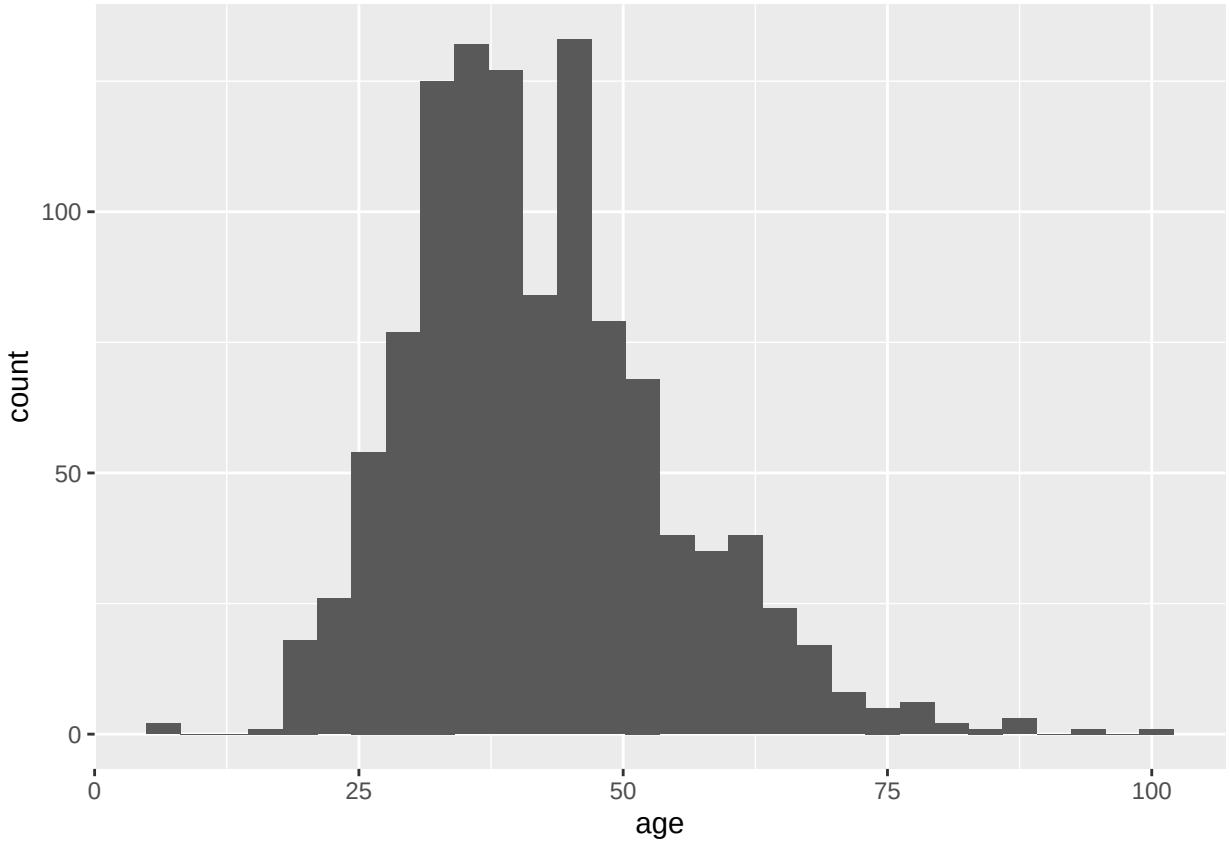
```
head(df.plot, 5)
```

```
##   sex age
## 1   1  26
## 2   1  99
## 3   1  59
## 4   1  27
## 5   1  52
```

We can use this new dataframe to create a histogram showing the the distribution of `age`:

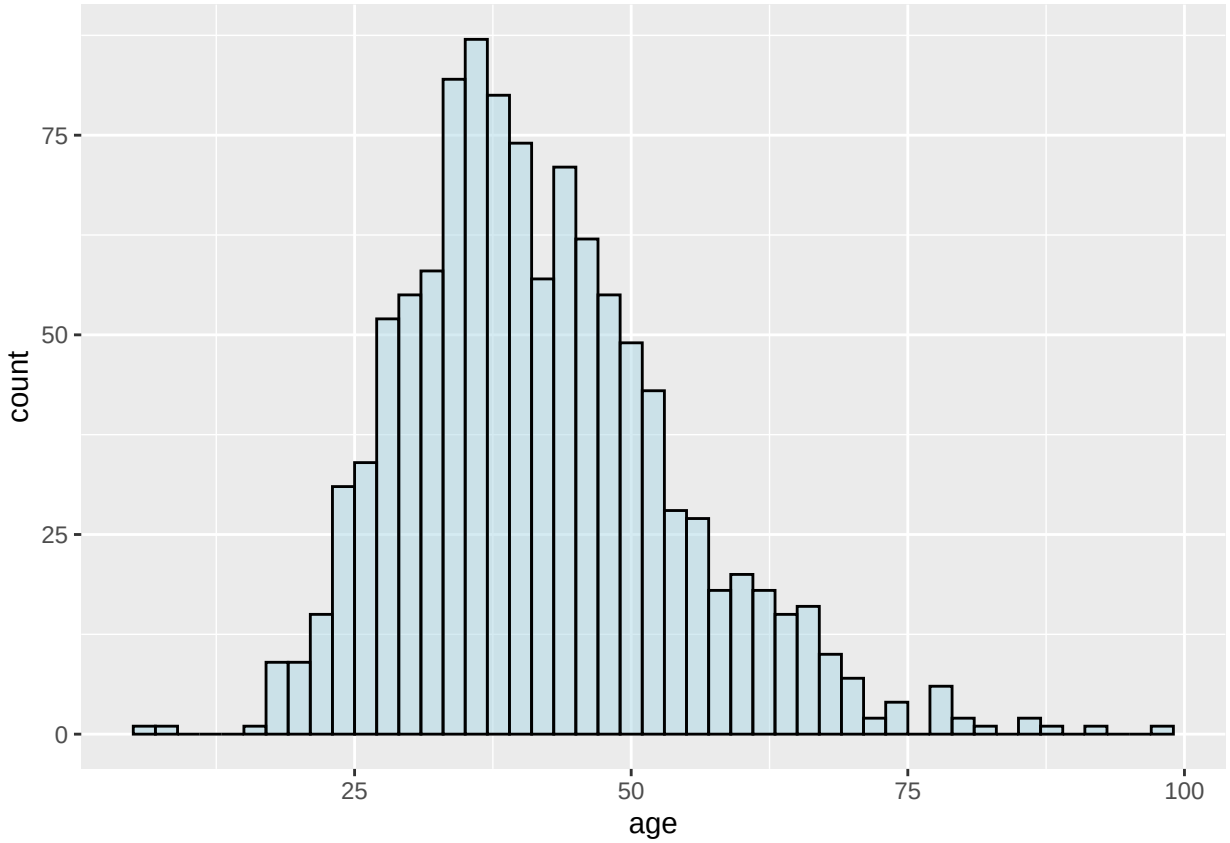
```
plot1 <- ggplot(data = df.plot, aes(x = age)) +
  geom_histogram()
plot1
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



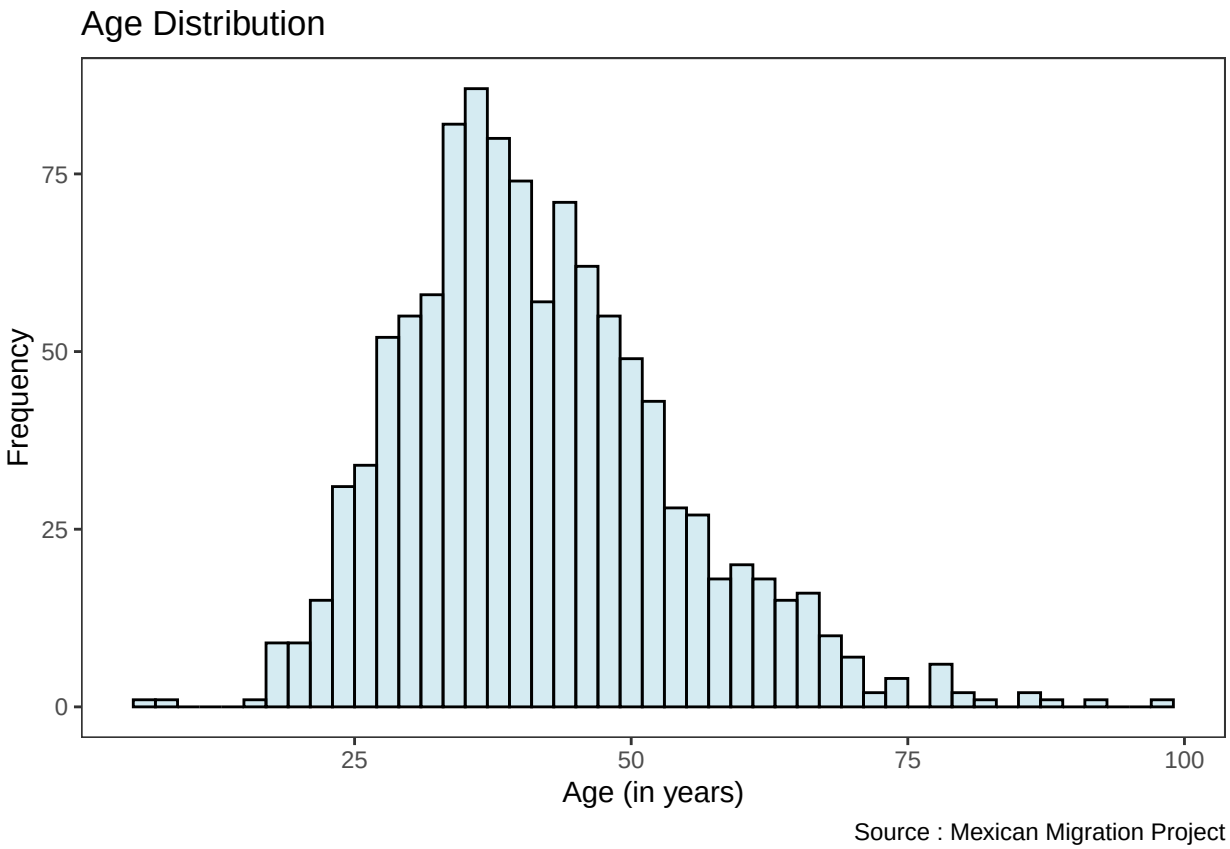
Note that in the aesthetics we have only defined x because we are not examining more variables. Also, note that we must execute the command `plot1` to visualize the plot. This is because R separates the task of creating an object from the task of visualizing. Let's change the binwidth and add some color to improve legibility:

```
plot1 <- ggplot(data = df.plot, aes(x = age)) +
  geom_histogram(binwidth = 2, color="black", fill="lightblue", alpha = 0.5)
plot1
```



Finally, let's add a title, axis labels, and a note at the bottom with the data source and change the plot theme.

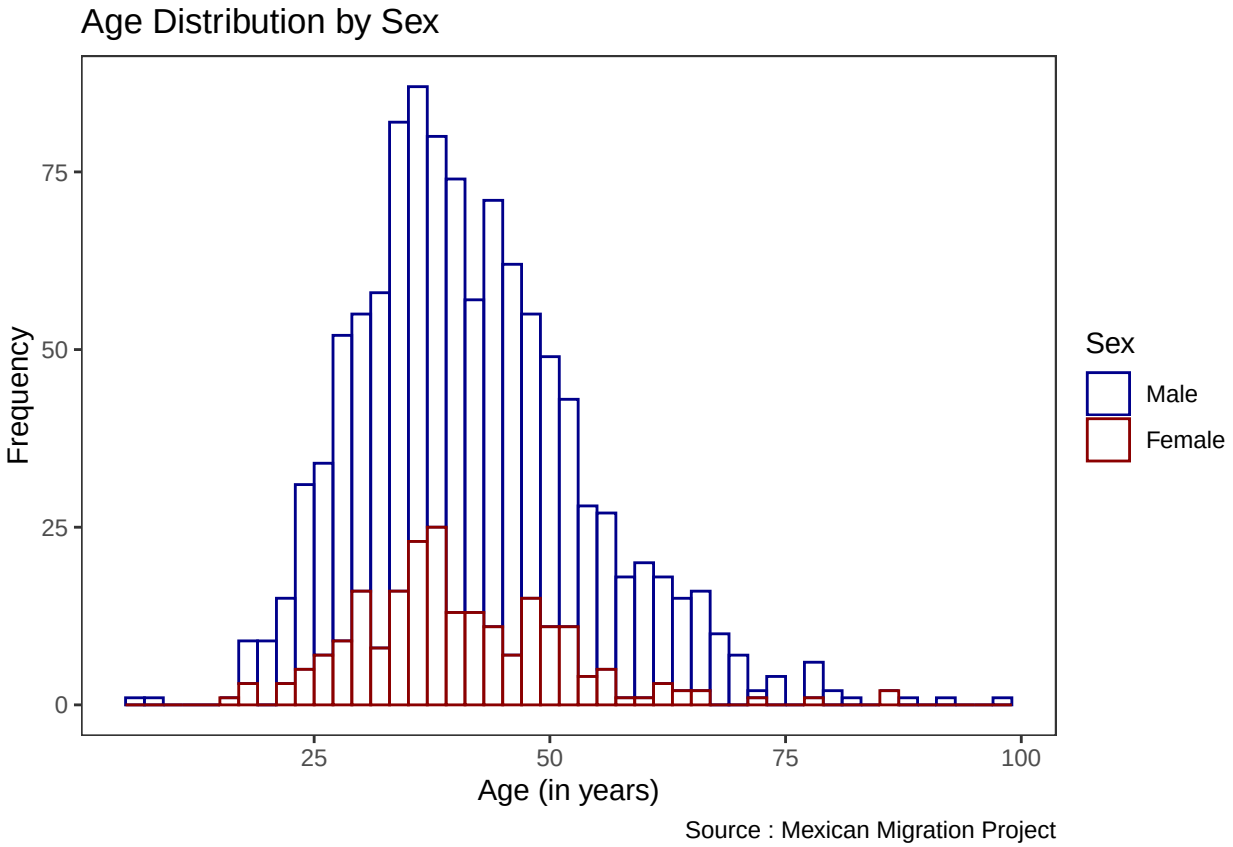
```
plot1 <- plot1 + labs(
  title = "Age Distribution",
  caption = "Source : Mexican Migration Project",
  x = "Age (in years)",
  y = "Frequency") +
  theme_test()
plot1
```



There are many other themes you can try: `theme_gray()`, `theme_bw()`, `theme_linedraw()`, `theme_light()`, `theme_dark()`, `theme_minimal()`, `theme_classic()`, `theme_void()`, and `theme_test()`.

We can use the argument the `color` argument in aesthetics to create separate histograms for each sex:

```
plot2 <- ggplot(df.plot, aes(x=age, color= factor(sex))) +
  geom_histogram(binwidth = 2, fill="white", alpha=0.5) +
  labs(
    title = "Age Distribution by Sex",
    caption = "Source : Mexican Migration Project",
    x = "Age (in years)",
    y = "Frequency",
    color = "Sex") +
  scale_color_manual(labels = c("Male", "Female"), values = c("darkblue", "darkred")) +
  theme_test()
plot2
```



Note that we need to convert `sex` into a factor variable using the function `factor()`.

Descriptive statistics: sex and educational level

Now, calculate the same descriptive statistics and produce the same plots for the school years completed (`edyrs`) and sex. Remember that you must start by cleaning the data.