Descriptive Statistics & Data Visualization by R

Shunsuke Tsuda

EC114 Introduction to Quantitative Economics
2026 Autumn



Today’s contents

- Summarize data characteristics using descriptive statistics
- Learn various methods to visual data

- Reference: The R Graph Gallery https://r-graph-gallery.com/index.html
- I thank Tomohiro Hara for sharing his lecture slides

Data used: NIDS (National Income Dynamics Study)

- Same as last time
- Household-level data from South Africa
  - A nationally representative sample of the country's population
  - 7,300 households and 28,000 individuals
  - Panel data: Surveyed the same households and individuals 5 times between 2008 and 2017
  - Surveyed various indicators via interviews, such as poverty, household well-being, household composition, health, and labor market participation
  - Many variables are publicly available
  - (Some variables that could lead to personal identification—such as detailed location, date of birth, and detailed occupation—can only viewed and analyzed by researchers on-site)

Reading data (Review)

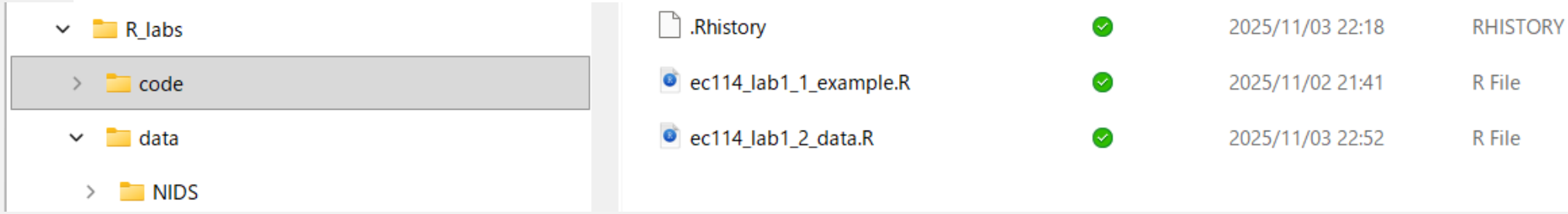
- Write down packages to be used for analysis
  - This time, we will use tidyverse, ggExtra, and table1

```
library(tidyverse)
library(ggExtra)
library(table1)
```

- Load the data, using a relative path for portability

```
[data name]=read.csv(file='[file name]', header=T)
```

```
12  library("this.path")
13  setwd(this.dir())      # Use a relative path, which makes the entire project "portable"
14  ##### Inputs
15  data_nidsw1   = read.csv(file='../data/NIDS/nids_w1_for_educ.csv', header=T)
```



Descriptive statistics

- **Descriptive statistics:** Statistics that show the characteristics of variables

Types of variables

- **Categorical variable:** Variables that cannot be measured numerically
  - Categories or groups to which each individual belongs
  - Taking the average has no meaning
- **Quantitative variable:** Data that can be measured numerically
- Are the following variables categorical or quantitative?
  - Age, Name, Birth year, Place of origin, Income

Briefly check the data contents

`summary([dataframe])`

- `[dataframe]` is your dataframe name
- You can check the summary of each variable by `summary`
- The minimum value, maximum value, mean, quartiles, and number of NAs for the data are displayed in the Console

```
> summary(data_nidsw1)
      hhid             pid             dob_y          gender
 Min.   :101012   Min.   :301011   Min.   :1910   Min.   :1.000
 1st Qu.:104192   1st Qu.:306393   1st Qu.:1957   1st Qu.:1.000
 Median :107358   Median :312914   Median :1973   Median :2.000
 Mean   :110139   Mean   :315249   Mean   :1982   Mean   :1.599
 3rd Qu.:110697   3rd Qu.:319219   3rd Qu.:1985   3rd Qu.:2.000
 Max.   :175202   Max.   :399997   Max.   :9999   Max.   :2.000
                                   NA's   :646    NA's   :646
```

Points to note when looking at summary

- Do Min/Max take meaningful values?
  - E.g. in the case below, `dob_y` is the variable for birth year, but max is 9999 → 9999 as a year is strange
  - Handle these missing values when using data
- Is the number of NAs reasonable?
- Grasp whether it is a quantitative variable or a qualitative variable
- You should check usable variables referring to a variable catalogue, etc.

```
> summary(data_nidsw1)
      hhid             pid             dob_y          gender      
 Min.   :101012   Min.   :301011   Min.   :1910   Min.   :1.000  
 1st Qu.:104192   1st Qu.:306393   1st Qu.:1957   1st Qu.:1.000  
 Median :107358   Median :312914   Median :1973   Median :2.000  
 Mean   :110139   Mean   :315249   Mean   :1982   Mean   :1.599  
 3rd Qu.:110697   3rd Qu.:319219   3rd Qu.:1985   3rd Qu.:2.000  
 Max.   :175202   Max.   :399997   Max.   :9999   Max.   :2.000  
                                   NA's   :646    NA's   :646    
```

Review: Handling missing values

- Change 9999 and 3333 in `dob_y` to `NA`
- Change `-3` in `popgroup`, `home_lng`, `marital`, `pension`, `income`, `unemp_insurance` to `NA`
- Change -3 and -5 in `educ` to `NA`
  - Note that `missing` and `NA/not applicable` were treated equally this time, but depending on the case, distinction may be necessary
- dplyr code can be grouped into the same command
  - E.g. Using `mutate` to rewrite missing values for multiple variables (see the code)
- It is good to check with `summary` again if the processing went well

Statistics used for descriptive statistics

- Statistics often used in descriptive statistics
  - Mean: `mean()`
  - Median: `median()`
  - Standard deviation: `sd()`
  - Minimum value: `min()`
  - Maximum value: `max()`
- Use `$` when accessing a specific variable in a dataframe
- If the data contains missing values, add the option `na.rm=TRUE`

E.g. `mean(data_nidsw1$income, na.rm=TRUE)`

Checking the contents of categorical variables

- For categorical variables, it is more important to check which categories exist and to what extent, rather than max, min, average, etc
- Can be displayed with `table([dataframe]$[varname])` or `with([dataframe], table([varname]))`

E.g. Check categories of `popgroup`:

```
table(data_nidsw1$popgroup)
```

Factorization of variables

- Variables have "possible values" recorded as class attributes
  - All the variables in the data we are using is recorded as `integer`
  - Others include `numeric` (decimals), `character` (text)
- E.g. `popgroup` is a variable with 4 categories: `African, Coloured, Asian/Indian, White`
  - Since the data is saved as `integer`, change it to an easy-to-understand `factor` (category)

```
data_nidsw1<-mutate(data_nidsw1,
popgroup=factor(popgroup, levels=1:4,
labels=c("African", "Coloured", "Asian/Indian",
"White")))
```

- `factor()`: The function that factorizes variables
- `Levels`: Set the range of factors
- `Labels`: Specify the content of labels
- `c()`: Specifies the content of labels as a vector

Exercise of factorizing variables

Let's try factorizing the following variables:

- `gender (1->Male, 2->Female)`
- `employed (1->Employed, 2->Unemployed)`

If you make the table again after factorizing, it should be displayed in `factor` format

E.g. `table(datanidsw1$popgroup)`

Check the relationship between two categorical variables

```
with([dataframe],table([var1],[var2])
```

: You can aggregate data that fits two categorical variables

E.g. Race and Unemployment

```
> with(data_nidsw1, table(popgroup, employed))
               employed
popgroup        Employed Unemployed
  African           1248       4867
  Coloured           417        684
  Asian/Indian        31         64
  White              177        258
```

Filters and pipe operators to analyze on a part of the data

- E.g. Look at the relationship between race and unemployment using data for males only

```
data_nidsw1 %>%
  filter(gender=="Male") %>%
  with(table(popgroup, employed))
```

```
> data_nidsw1 %>%
+   filter(gender=="Male") %>%
+   with(table(popgroup, employed))
                employed
popgroup         Employed Unemployed
  African             665       1771
  Coloured            217        219
  Asian/Indian         17         21
  White                86        107
```

Add totals for each row/column to the table

- Add `addmargin()` with the pipe operator

```
data_nidsw1 %>%
  with(table(popgroup, employed)) %>%
  addmargins()
```

```
> ### Add margins
> data_nidsw1 %>%
+   with(table(popgroup, employed)) %>%
+   addmargins()
                employed
popgroup         Employed Unemployed  Sum
  African            1248       4867 6115
  Coloured            417        684 1101
  Asian/Indian         31         64   95
  White               177        258  435
  Sum                1873       5873 7746
```

Change table notation to percentage

- Add `prop.table()` with the pipe operator
- Furthermore, adding `addmargin()` can add frequency (%) to the margins
- 3 patterns for adding frequency
  - Overall percentage
  - Percentage in row direction
  - Percentage in column direction
- The number of decimal places to display is determined by `round(digit=)`
- Such an aggregation table is called a cross-tabulation
- Cross-tabulation tables are very useful for showing the state of data before conducting full-scale analysis
- Including them in research papers or reports can roughly convey the contents of the data

Cross-tabulation (1): Overall percentage

- Add `prop.table()` & `addmargin()` with the pipe operator
- `round(digit=4)*100` for 4-digit display and percentage display
- Each cell is the realization probability

E.g. 16.11% = overall probability of `popgroup==African` & `employed==Employed`

```
> # Table with proportion (1) overall proportions
> data_nidsw1 %>%
+   with(table(popgroup, employed)) %>%
+   prop.table() %>%
+   addmargins() %>%
+   round(digits=4)*100
                employed
popgroup         Employed Unemployed    Sum
  African           16.11      62.83  78.94
  Coloured           5.38       8.83  14.21
  Asian/Indian       0.40       0.83   1.23
  White              2.29       3.33   5.62
  Sum               24.18      75.82 100.00
```

- Adding horizontally calculates the percentage of `popgroup==African`
- Adding vertically calculates the percentage of `employed==Employed`

Cross-tabulation (2): Conditioning on each row

- Add totals in the column direction with `addmargins(margin=1)`
  - Specify only column direction with `margin=1`
- Percentage notation with `prop.table(margin=1)`
- Add sum in the row direction with `addmargins(margin=2)`
- Each cell is the **conditional probability** given `popgroup`

E.g. 20.41% = probability of `employed==Employed` when `popgroup==African`

```
> # Table with proportion (2) conditional on row
> data_nidsw1 %>%
+   with(table(popgroup, employed)) %>%
+   addmargins(margin=1) %>%
+   prop.table(margin=1) %>%
+   round(digits=4) %>%
+   addmargins(margin=2)*100
                employed
popgroup         Employed Unemployed    Sum
  African           20.41      79.59 100.00
  Coloured          37.87      62.13 100.00
  Asian/Indian      32.63      67.37 100.00
  White             40.69      59.31 100.00
  Sum               24.18      75.82 100.00
```

- Adding horizontally results in 100%
- The below "sum" still presents the percentage of `employed==Employed`

Cross-tabulation (3): Conditioning on each column

Exercise: Apply the method from the previous page conditioned on row direction to create a cross-tabulation conditioned on `employed`

Summary: Different uses of cross-tabulation

- Want to know the employment rate of each race

→(2): Condition on each row (`popgroup`)

- Want to know the racial breakdown of employed and unemployed people

→(3): Condition on each column (`employed`)

- Want to know the percentage of each race and employed/unemployed people within the total population

→(1): Find the percentage of each cell

Exporting descriptive statistics

- Including descriptive statistics in papers/reports to show the overview of data used in analysis is very beneficial
- There are multiple packages that can export descriptive statistics in a form that can be shown in reports, but this time we will use `stargazer`

- Please install `stargazer` and load it into the library

Simple descriptive statistics table

```
table<-data_nidsw1%>%
select(income, educ)%>%
stargazer(type="text")
```

- Select necessary variables with `select`
- Output as text data with `type="text"`

```
==========================================================
Statistic   N      Mean     St. Dev.   Min   Max
----------------------------------------------------------
income     1,622 3,032.000 4,326.000   0   42,000
educ       6,743   9.179      3.150     0     18
----------------------------------------------------------
```

Exporting the output

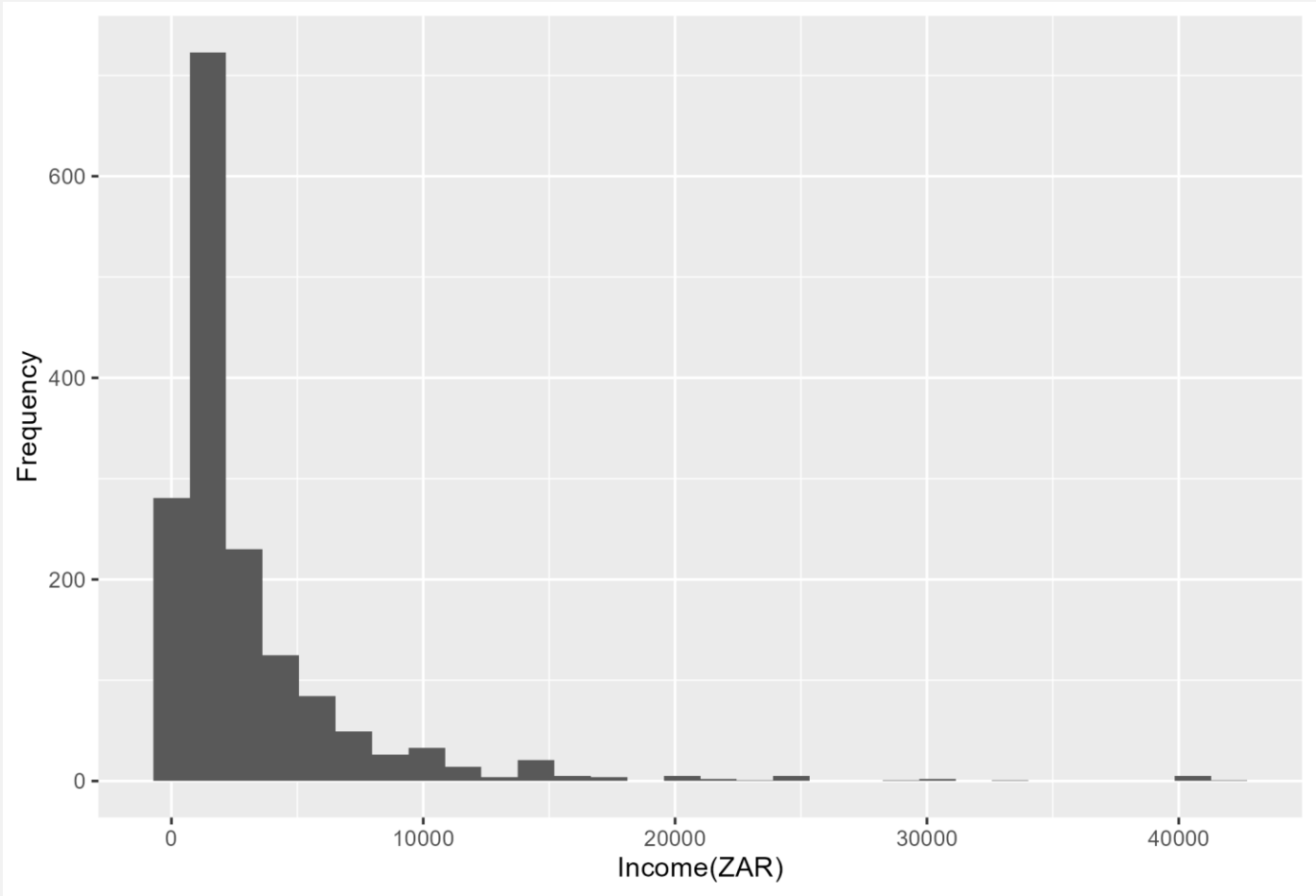
- `table_simple<-data_nidsw1%>% select(income, educ)%>% stargazer(type="text", out="../outputs/summary_1.txt")`
- Output can be saved by adding `out=" "` as an option
- `table_male<-data_nidsw1%>% filter(gender=="Male")%>% select(income, educ)%>% stargazer(type="text", out="../outputs/summary_2.txt")`
- By inserting `filter()`, output results for specific attributes only

Data visualization

- To know the characteristics of variables, it is sometimes important to show information visually, not just descriptive statistics
- The `ggplot2` package (contained in `tidyverse`) is useful for visualization

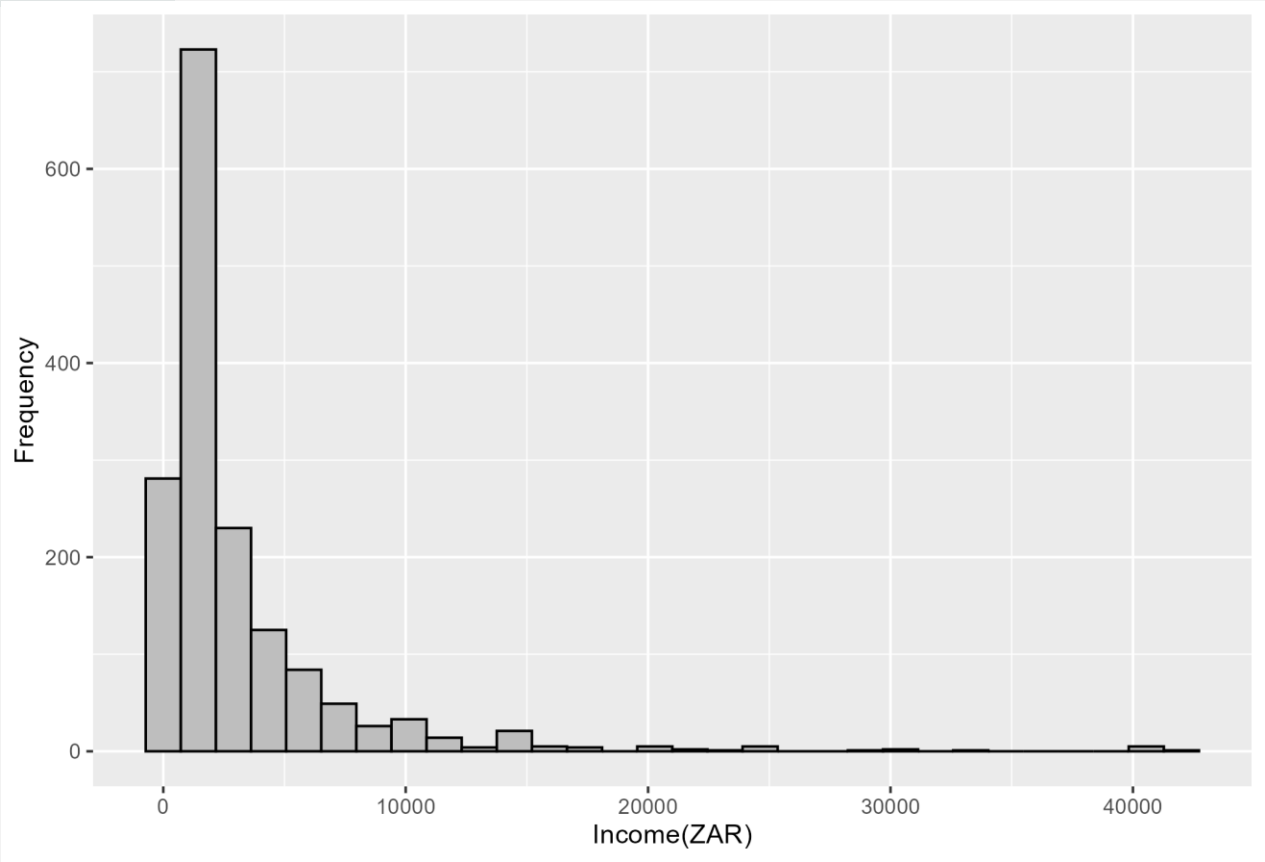
Histogram: Frequency

```
hist_income1<-ggplot(data_nidsw1,
aes(x=income)) +
  geom_histogram()+
  labs(x= "Income(ZAR)",
y="Frequency")
print(hist_inc1)
ggsave("../outputs/
hist_income1.png")
```



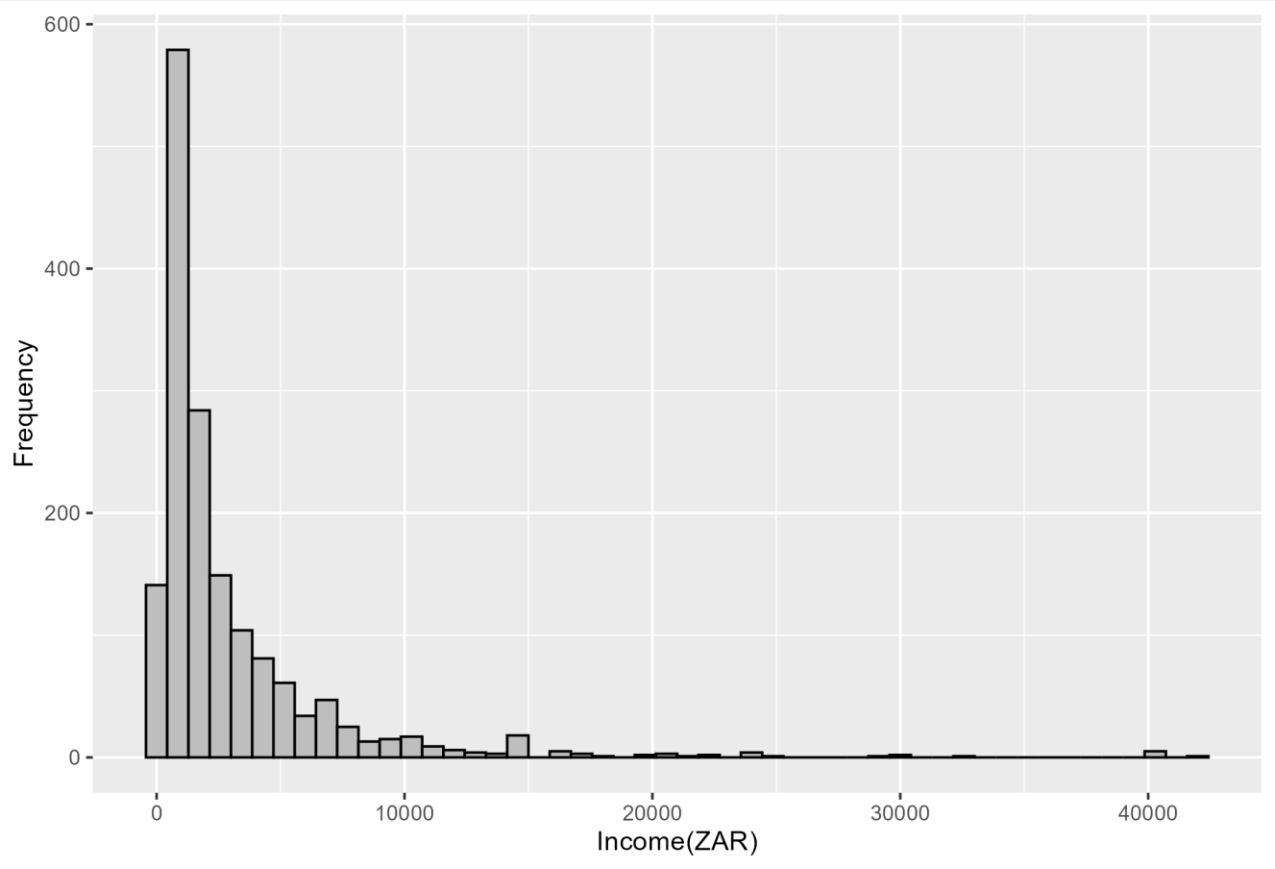
Histogram: Change bar color

```
hist_income2<-ggplot(data_nidsw1,
aes(x=income)) +
  geom_histogram(color="black",
fill="grey")+
 labs(x= "Income(ZAR)",
y="Frequency")
 print(hist_income2)
ggsave("../outputs/
hist_income2.png")
```



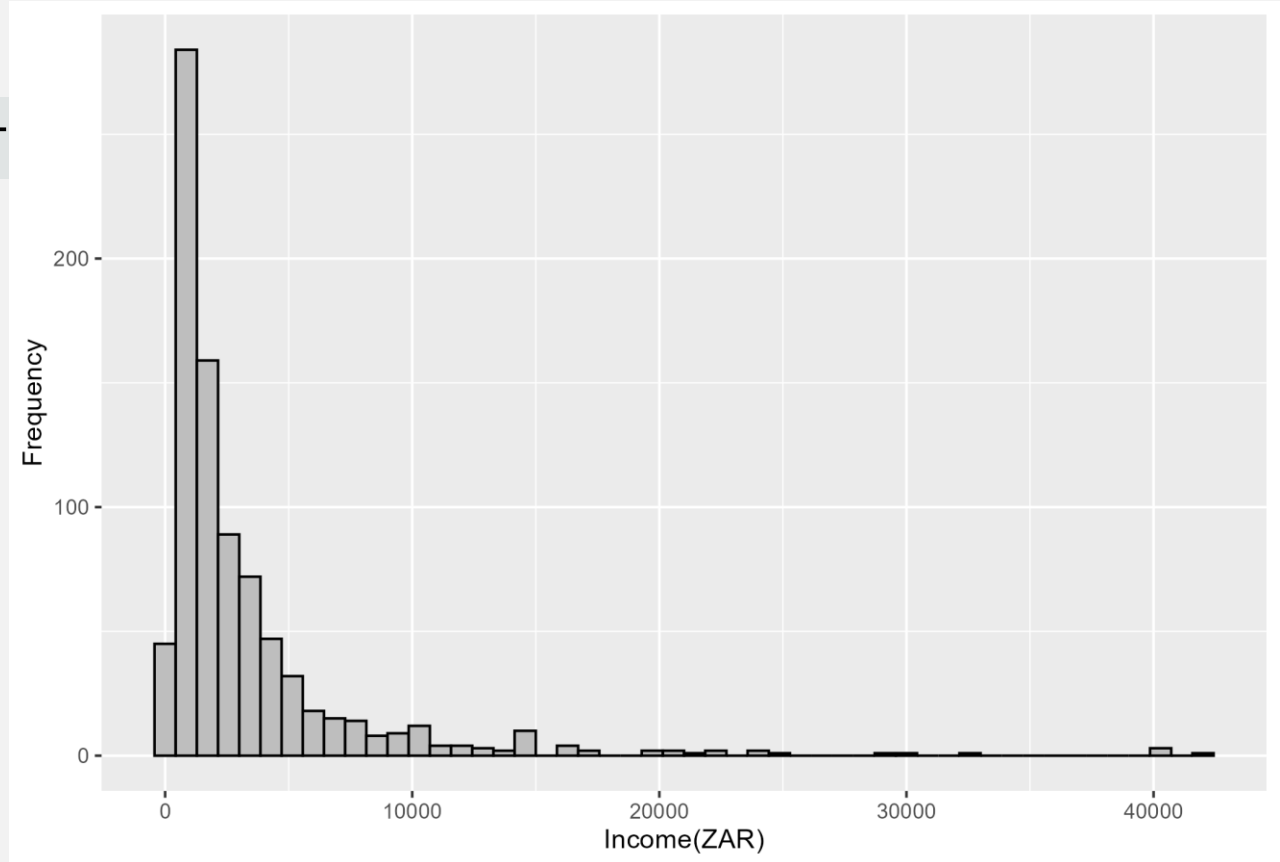
Histogram: Change the number of bins

```
hist_income3<-ggplot(data_nidsw1,
aes(x=income)) +
 geom_histogram(bins=50,
color="black", fill="grey")+
 labs(x= "Income(ZAR)",
y="Frequency")
 print(hist_income3)
ggsave("../outputs/
hist_income3.png")
```



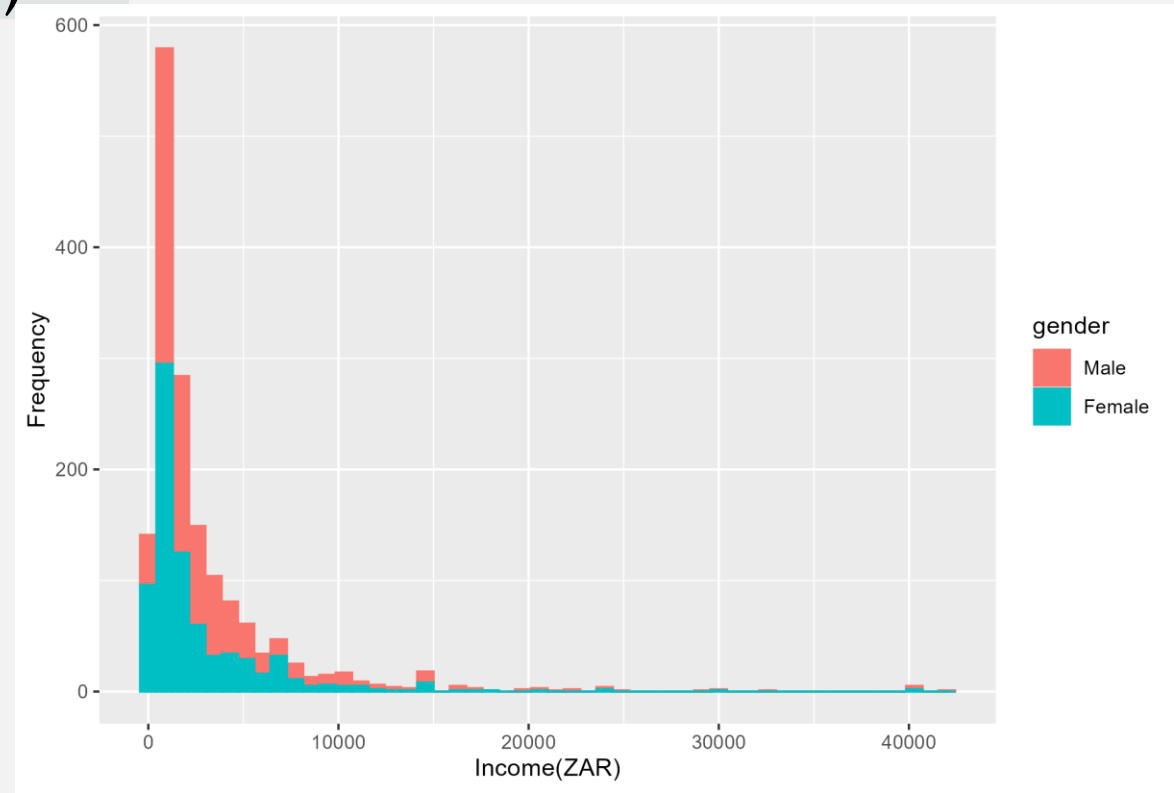
Histogram: Change only part of the data (Male)

```
hist_income4<-data_nidsw1 %>%
  filter(gender=="Male") %>%
  ggplot(aes(x=income)) +
  geom_histogram(bins=50,
color="black", fill="grey")+
 labs(x= "Income(ZAR)",
y="Frequency")
 print(hist_income4)
ggsave("../outputs/
hist_income4.png")
```



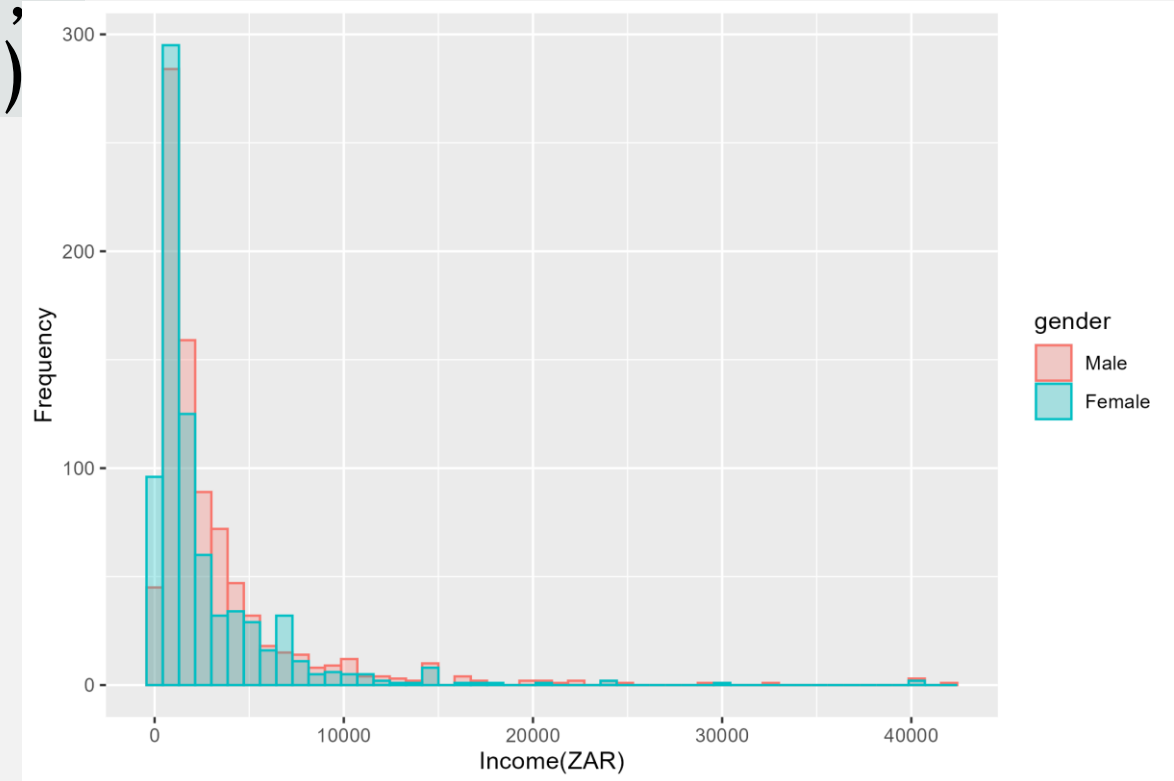
Histogram: Stack multiple groups (Male/Female)

```
hist_income5<-ggplot(data_nidsw1,
aes(x=income)) +
 geom_histogram(bins=50,
aes(color=gender, fill=gender))+
labs(x= "Income(ZAR)",
y="Frequency")
 print(hist_income5)
ggsave("../outputs/
hist_income5.png")
```



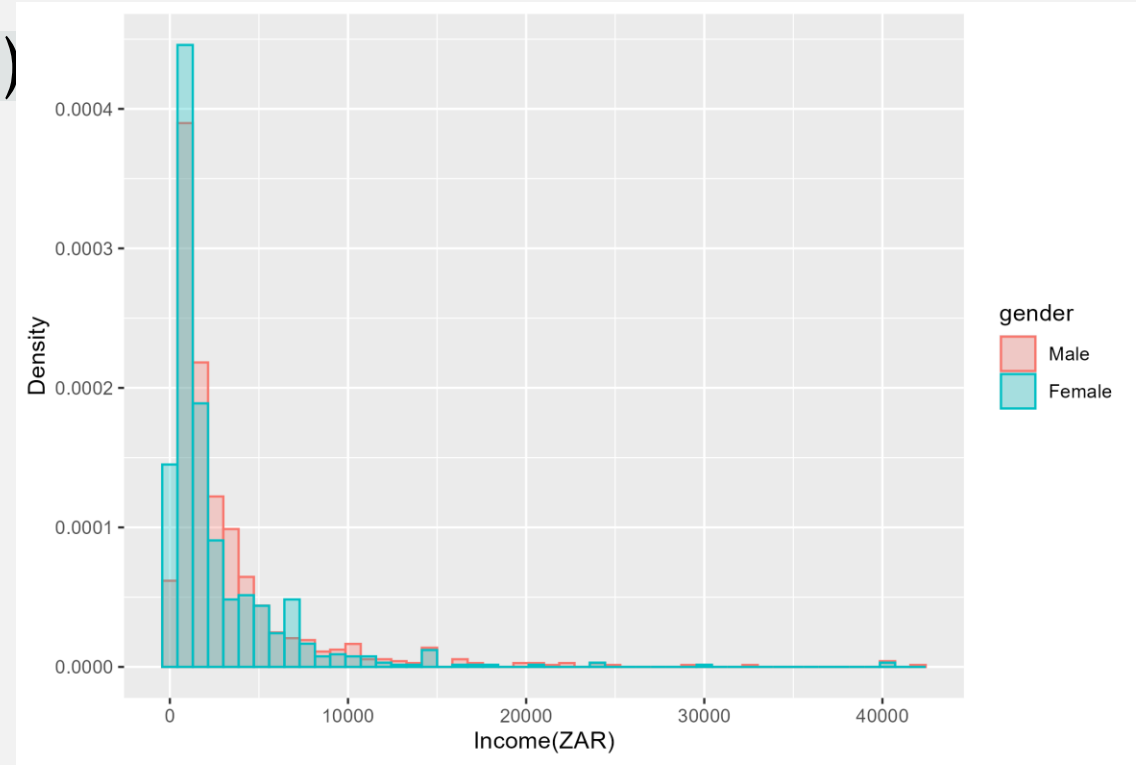
Histogram: Overlay multiple groups (Male/Female)

```
hist_income5rev<-
ggplot(data_nidsw1,
aes(x=income)) +
 geom_histogram(bins=50,
aes(color=gender, fill=gender),
position="identity", alpha=0.3)
+labs(x= "Income(ZAR)",
y="Frequency")
 print(hist_income5rev)
ggsave("../outputs/
hist_income5rev.png")
```



Histogram: Probability density

```
hist_income6<-data_nidsw1 %>%
  ggplot(aes(x=income, y=..density..))
+
 geom_histogram(bins=50,
aes(color=gender, fill=gender), ,
position="identity", alpha=0.3)
+ labs(x= "Income(ZAR)", y="Density")
+ scale_y_continuous(labels =
function(x) format(x, scientific =
FALSE))
 print(hist_income6)
ggsave("../outputs/
hist_income6.png")
```



Color codes

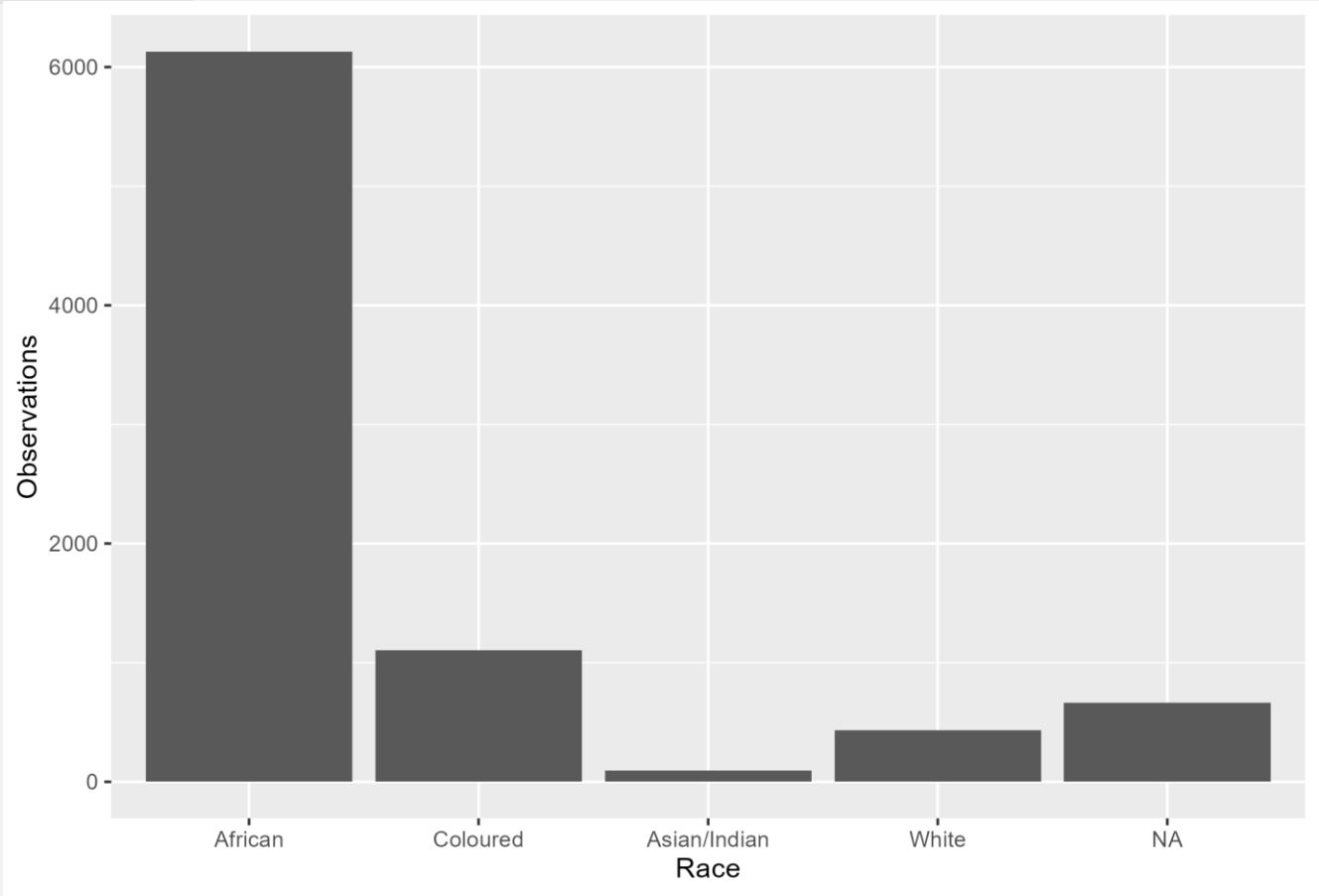
- When specifying colors manually, use color codes using hexadecimal
  - E.g. #00AFBB, "#E7B800
- There are many web pages that convert actual colors to color codes, so it is good to use them to change colors
  E.g. https://www.rapidtables.com/web/color/RGB_Color.html

Bar graph

- Similar to histogram, but the horizontal axis is a categorical variable
- Can be used even if there is no order on the x-axis
- Useful for comparison between categories

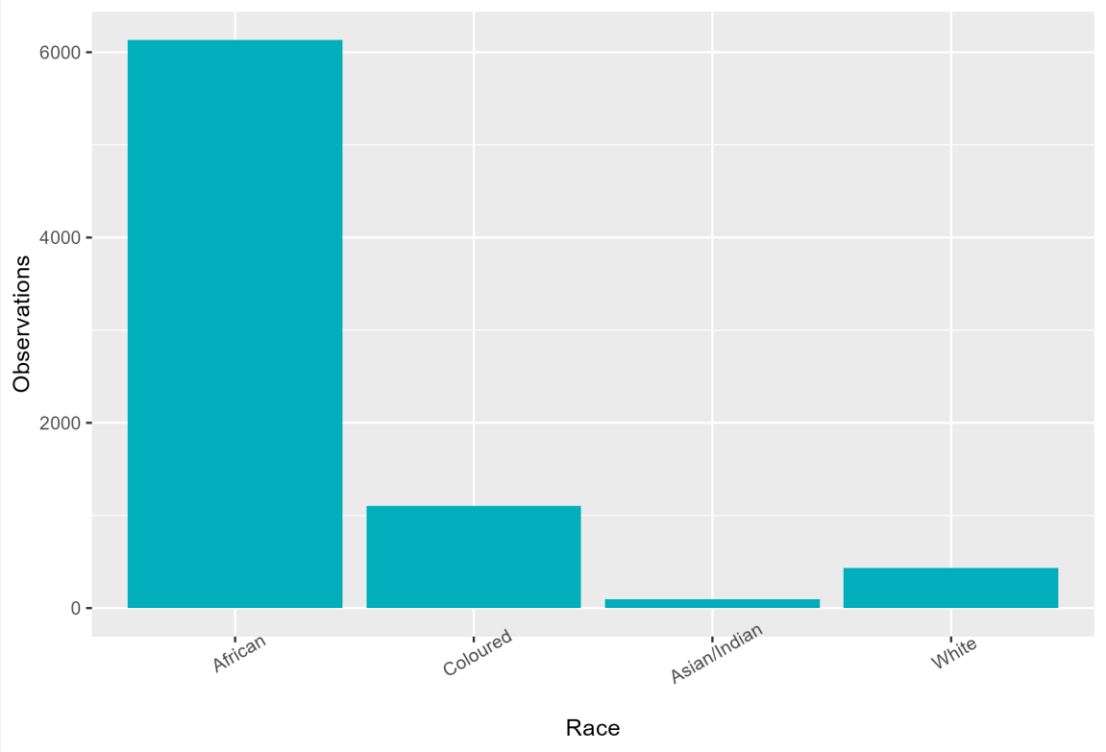
Bar graph: Example

```
bar_race1<-ggplot(data_nidsw1,
aes(x=popgroup)) +
  geom_bar(stat="count")+
  labs(x="RRace",
y="Observations")
print(bar_race1)
ggsave("../outputs/
bar_race1.png")
```



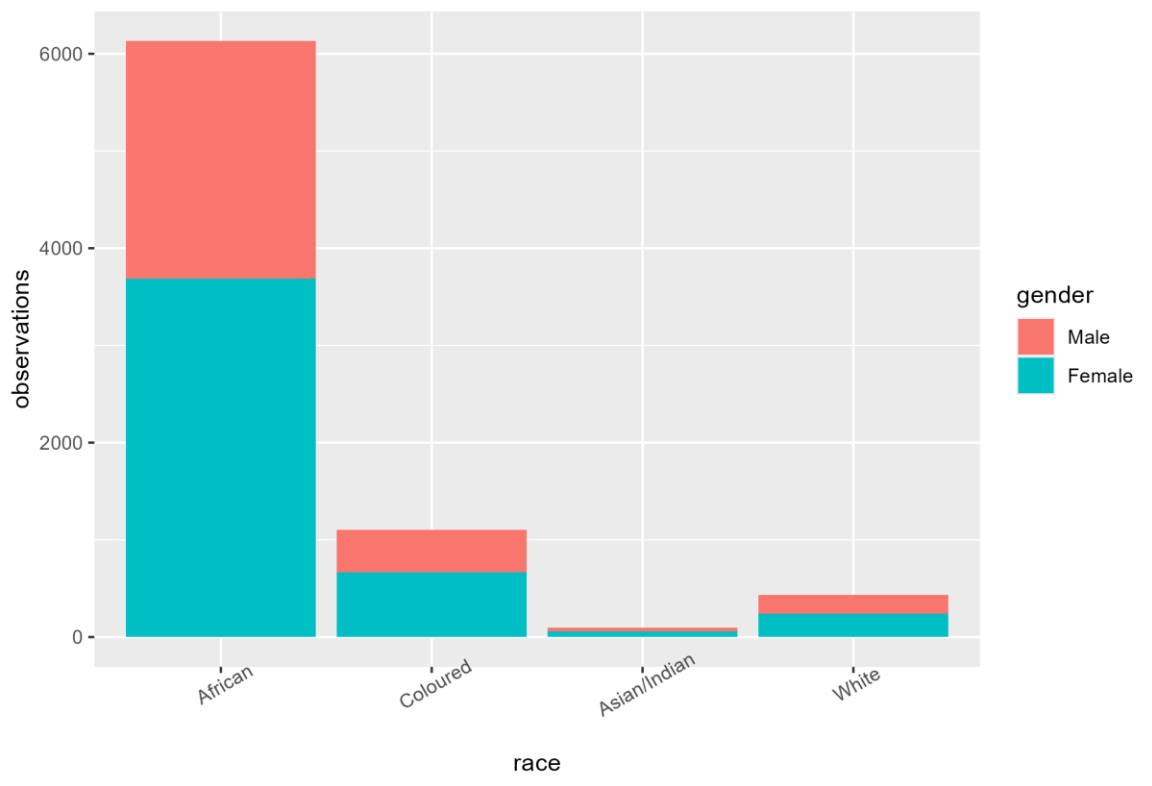
Bar graph: Some variants in design

```
bar_race2 <- data_nidsw1 |>
dplyr::filter(!is.na(popgroup)) |>
  ggplot(data_nidsw1,
aes(x=popgroup)) +
  geom_bar(stat="count",
fill="#00AFBB")+
  labs(x="Race", y="Observations")+
 theme(
 axis.text.x = element_text(angle =
30),
 axis.text.y = element_text(angle =
0)   )
print(bar_race2)
ggsave("../outputs/bar_race2.png")
```



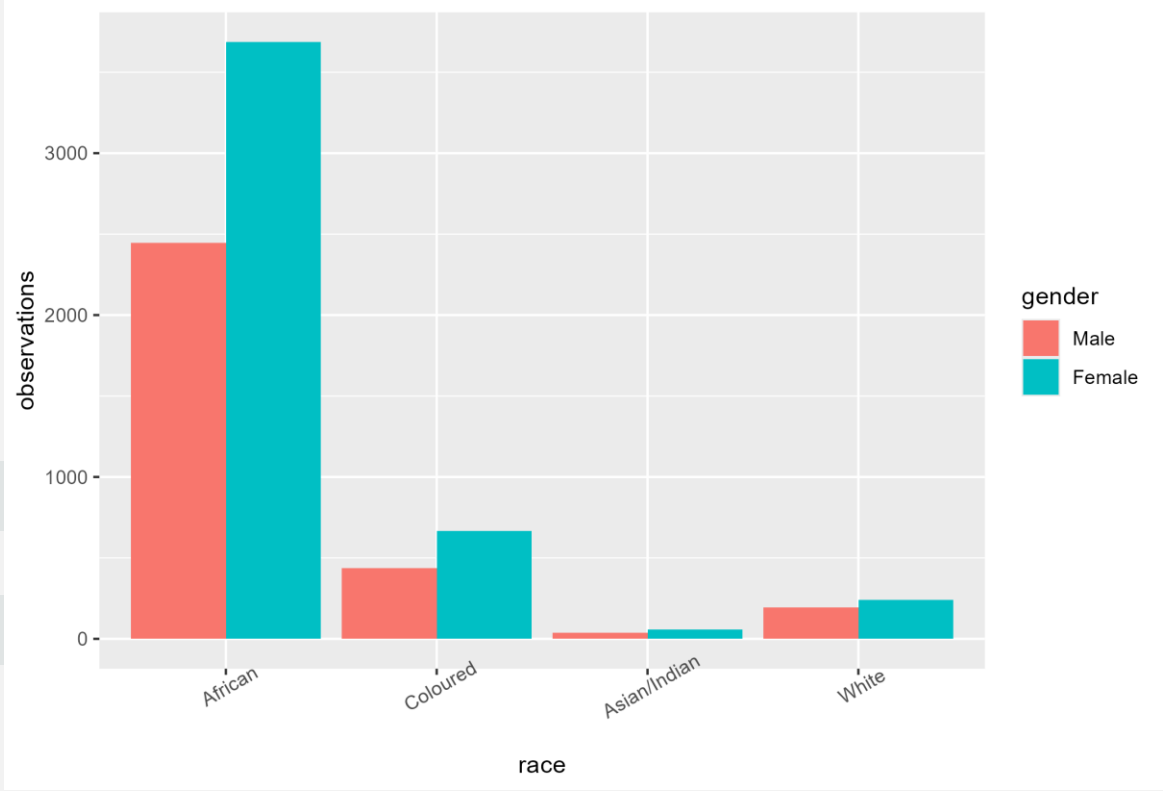
Bar graph: Display breakdown by gender (1)

```
bar_race3<- data_nidsw1 |>
dplyr::filter(!is.na(popgroup)) |>
  ggplot(aes(x=popgroup,
fill=gender)) +
  geom_bar(stat="count",
position="stack")+
  labs(x="race", y="observations")+
  theme(
 axis.text.x = element_text(angle =
30),
 axis.text.y = element_text(angle =
0)   )
print(bar_race3)
ggsave("../output/bar_race3.png")
```



Bar graph: Display breakdown by gender (2)

```
bar_race4<- data_nidsw1 |>
dplyr::filter(!is.na(popgroup)) |>
  ggplot(aes(x=popgroup,
fill=gender)) +
  geom_bar(stat="count",
position="dodge")+
  labs(x="race", y="observations")+
  theme(
 axis.text.x = element_text(angle =
30),
 axis.text.y = element_text(angle =
0)   )
print(bar_race4)
ggsave("../output/bar_race4.png")
```

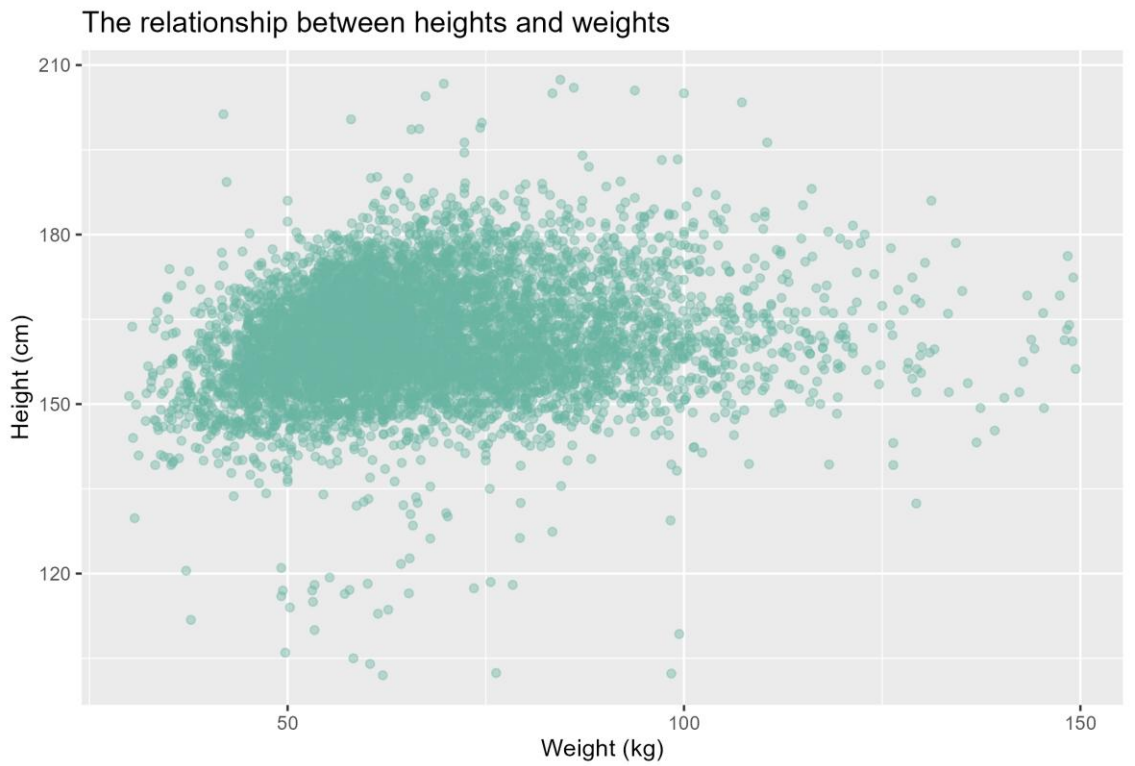


Scatter plot

- Take quantitative variables on the x-axis and y-axis to grasp the relationship between the two variables
- The first step to do to understand the association between two variables

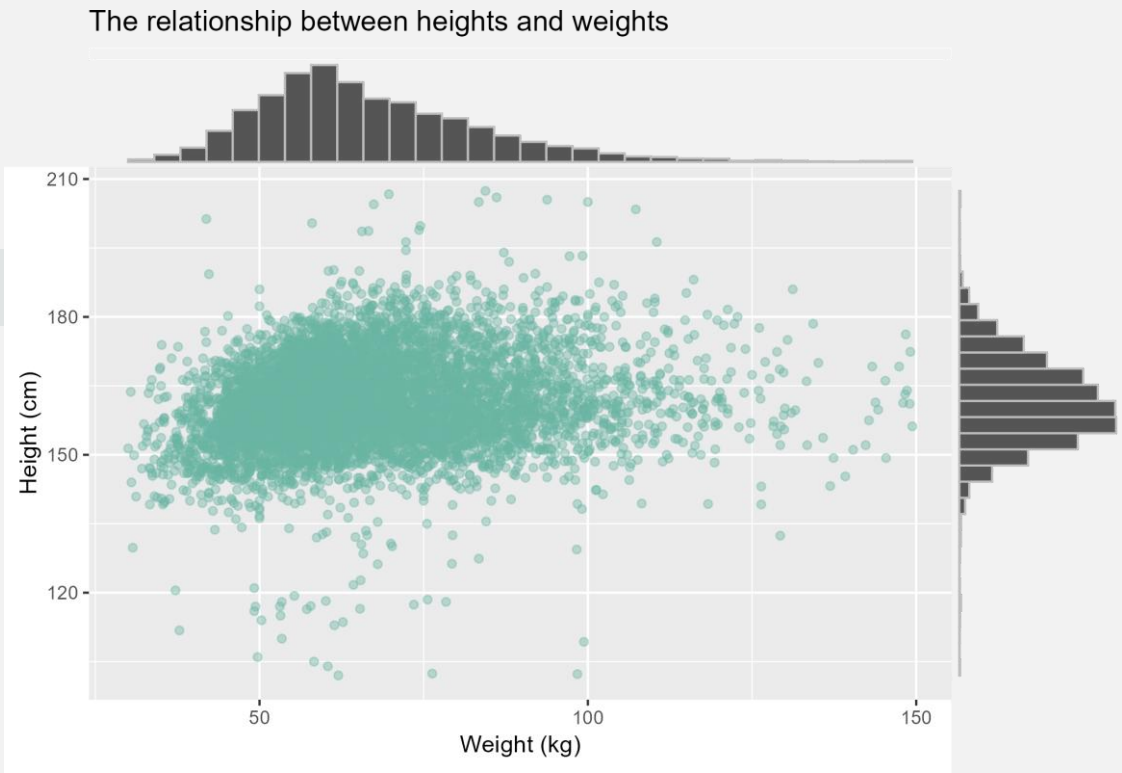
Scatter plot: Example

```
scatter_height_weight_1<-
ggplot(data=data_nidsw1,
mapping=aes(x=weight,
y=height))+
  geom_point(color="#69b3a2",
alpha=0.4)+
  labs(x="Weight (kg)",
y="Height (cm)", title="The
relationship between…")
print(scatter_height_weight1)
ggsave("../outputs/scatter_hw1.
png")
```



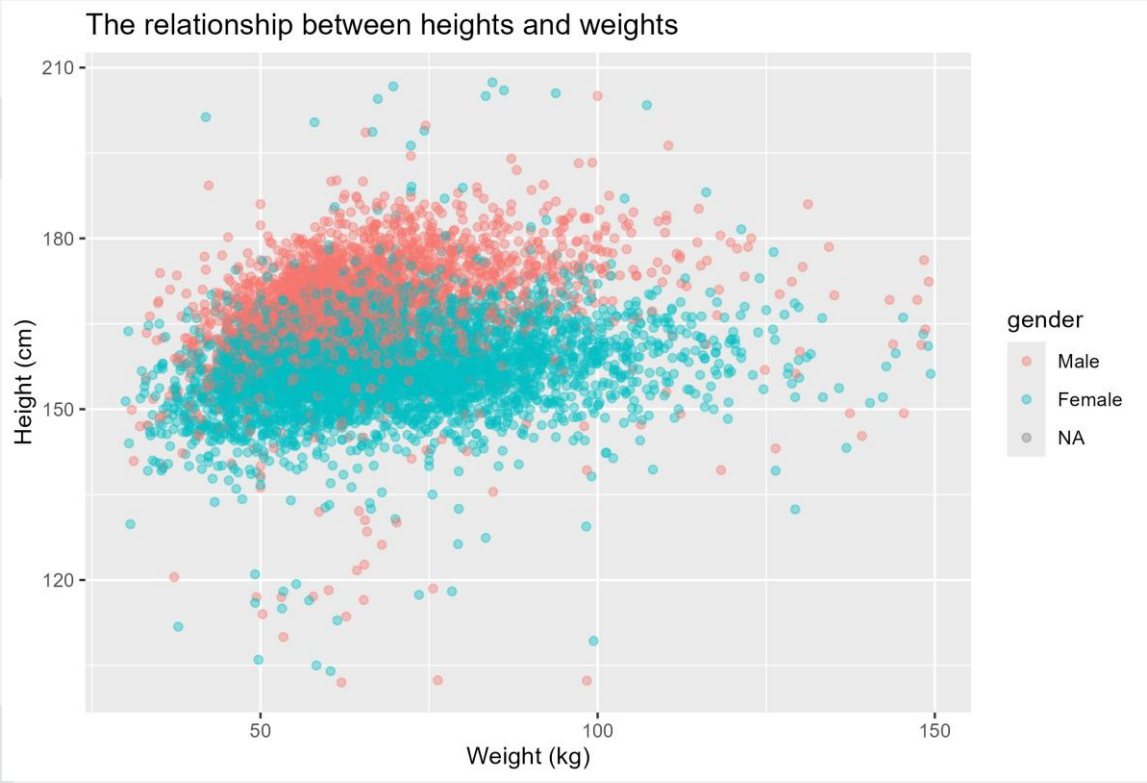
Scatter plot: With histograms

```
scatter_height_weight_1<-
ggplot(data=data_nidsw1,
mapping=aes(x=weight, y=height))+
  geom_point(color="#69b3a2",
alpha=0.4)+
  labs(x="Weight (kg)", y="Height
(cm)", title="The relationship
between…")
scatter_height_weight_2 <-
ggExtra::ggMarginal(scatter_heigh
t_weight_1, type="histogram",
color="grey")
print(scatter_height_weight2)
ggsave("../outputs/scatter_hw2.pn
g", scatter_height_weight_2)
```



Scatter plot: By group

```
scatter_height_weight3<-
ggplot(data=data_nidsw1,
mapping=aes(x=weight, y=height,
color=gender, fill=gender))+
 geom_point(alpha=0.4)+
  labs(x="Weight (kg)",
y="Height (cm)", title="The
relationship between…")
print(scatter_height_weight3)
ggsave("../outputs/scatter_hw3.
png")
```



Exercise

- Visualize the data to show the income difference between two racial groups, African and White
- Visualize the data to show the relationship between education and income