Data Visualization with ggplot2 (Part 2)

Case Study: California Health Interview Survey Dataset

Seun Odeyemi

2019-03-26

Contents

Load Libraries

```
library(readr)
library(dplyr)
library(ggplot2)
library(tidyr)
library(skimr)
library(knitr)
library(kableExtra)
library(RColorBrewer)
library(grid)
library(ggthemes)
library(forcats)
library(GGally)
library(here)
```

CHIS - Descriptive Statistics

Let's wrap up this course with a case study using a publicly available survey data. We will begin by understanding how to create meaningful plots to describe data distributions and then we will proceed by learning how to add statistics and generalizing our plotting functions to make plots that are both explanatory, exploratory, easy to use. The dataset we will use is from the California Health Interview Survey (CHIS): CHIS is the largest state health survey in the United States. It is a valuable source of data on California and it includes a wide variety of variables such as age, weight, eating habits, ethnicity, and all manner of personal health and economic measurements such as diabetes and poverty level.

The dataset we'll use is from the 2009 adult survey. The original dataset contains over 47000 observations and over 500 variables. To make our analysis a bit more focused, we are going to use a filtered version that only includes the 10 variables of particular interest to our analysis.

The variables are listed in the table below:

Variable	Description
RBMI	BMI Category Description
BMI_P	BMI_value
RACEHPR2	Race
SRSEX	Sex
SRAGE_P	Age
MARIT2	Marital Status
AB1	General Health Condition
ASTCUR	Current Asthma Status
AB51	Type 1 or Type II Diabetes
POVLL	Poverty Level

Table 1: A table showing the variables and their descriptions of the CHIS dataset. For more information visit UCLA's Health Policy Institute website.

Import CHIS Dataset from Stata Data File

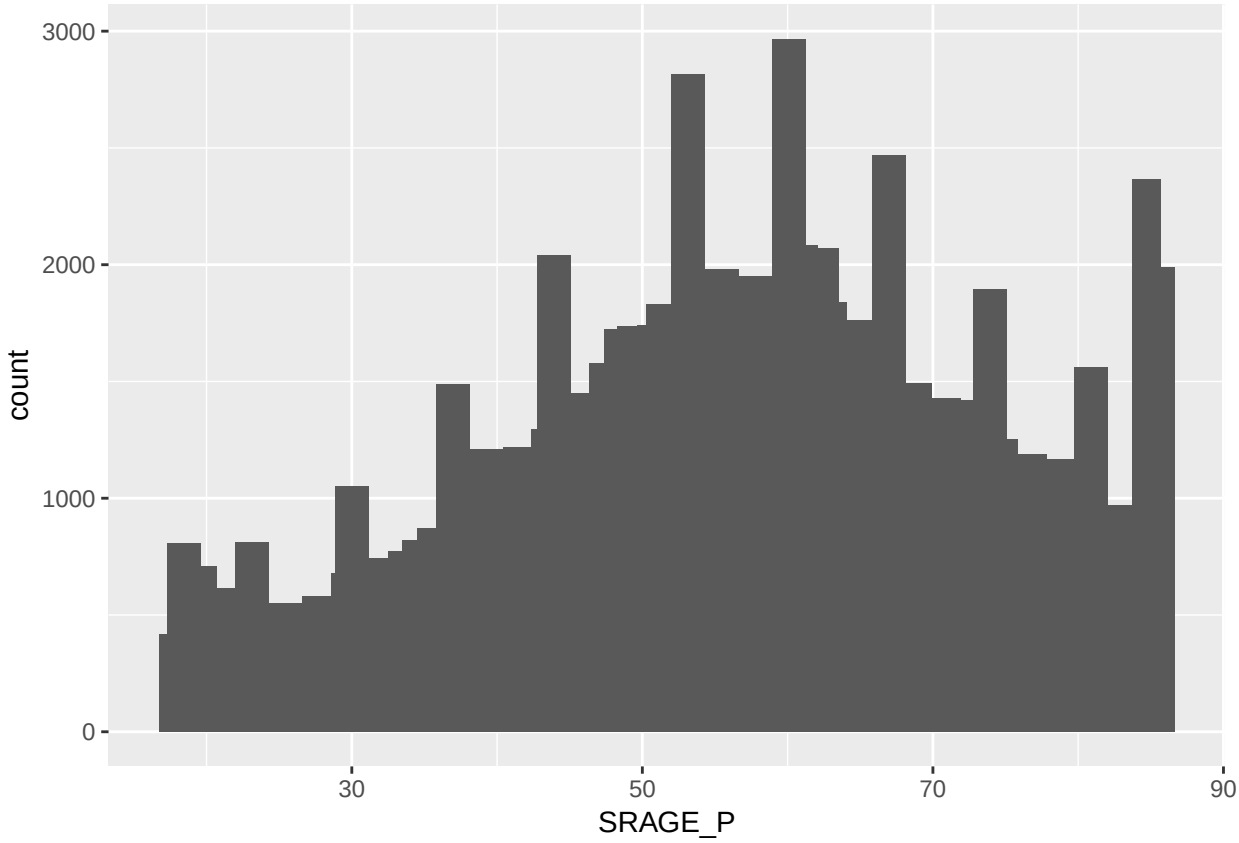
```
# Load the foreign library to access the read.dta function
# read.dta allows us to import Stata data files into R

library(foreign)

# Read in CHIS dataset and convert to a tibble
chis.data <- read.dta(here("ADULT.dta")) %>% as_tibble()

# select the variables of interest
chis.data.reduced <- chis.data %>%
  select(rbmi, bmi_p, racehpr2, srsex, srage_p, marit2, ab1, astcur, ab51, povll)

# see the first five rows

head(chis.data.reduced)
#> # A tibble: 6 x 10
#>   rbmi    bmi_p racehpr2 srsex  srage_p marit2  ab1    astcur  ab51  povll
#>   <fct>   <dbl> <fct>    <fct>    <int> <fct>   <fct>  <fct>   <fct> <fct>
#> 1 OVERWE~  28.9 WHITE    MALE        32 MARRIED EXCEL~ NO CUR~ INAP~ 300% ~
#> 2 OVERWE~  26.2 WHITE    FEMALE      80 WID/SE~ EXCEL~ NO CUR~ INAP~ 300% ~
#> 3 OVERWE~  25.1 WHITE    MALE        71 MARRIED VERY ~ CURREN~ INAP~ 300% ~
#> 4 NORMAL~  25.0 WHITE    MALE        39 NEVER ~ EXCEL~ NO CUR~ INAP~ 300% ~
#> 5 OVERWE~  25.1 WHITE    MALE        75 MARRIED VERY ~ NO CUR~ INAP~ 300% ~
#> 6 OBESE ~  32.2 WHITE    FEMALE      53 MARRIED GOOD   CURREN~ INAP~ 300% ~

# filter for the 4 largest ethnicities

chis.data.reduced <- chis.data.reduced %>%
  filter(racehpr2 %in% c("LATINO", "ASIAN", "AFRICAN AMERICAN", "WHITE"))

# change chis.data.reduced name to adult

adult <- chis.data.reduced
```

```r
# convert variable names to upper case

library(janitor)

adult <- adult %>% clean_names(case = "all_caps")
```

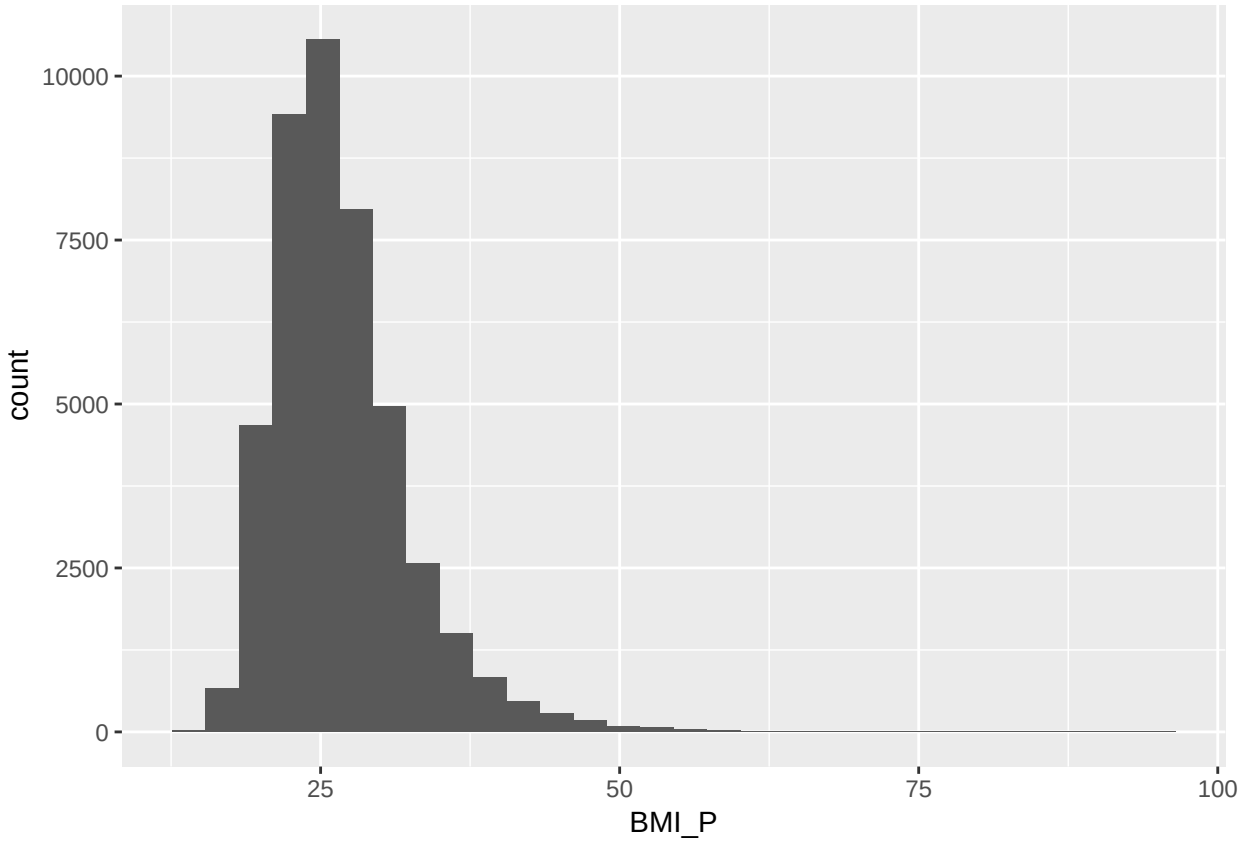
Some Exploratory Plots

Let's investigate the relationship between BMI and Age. BMI is the Body Mass Index and it is calculated as $weight/height^2$. The obvious first visualization for this variables is to look at the distribution of the univariate data, Age.

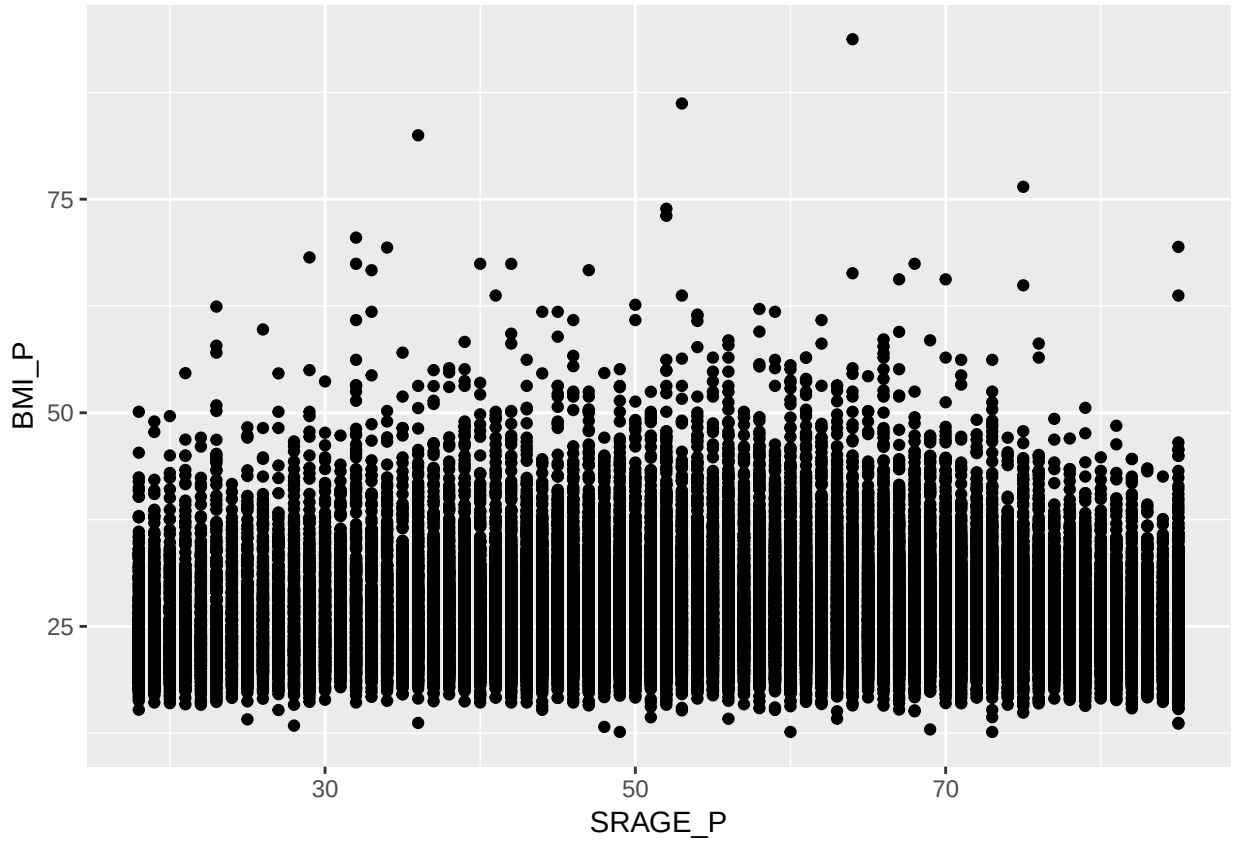
```r
# Histogram of age
ggplot(adult, aes(x = SRAGE_P)) +
  geom_histogram() +
  stat_bin(bins = 35)
#> `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



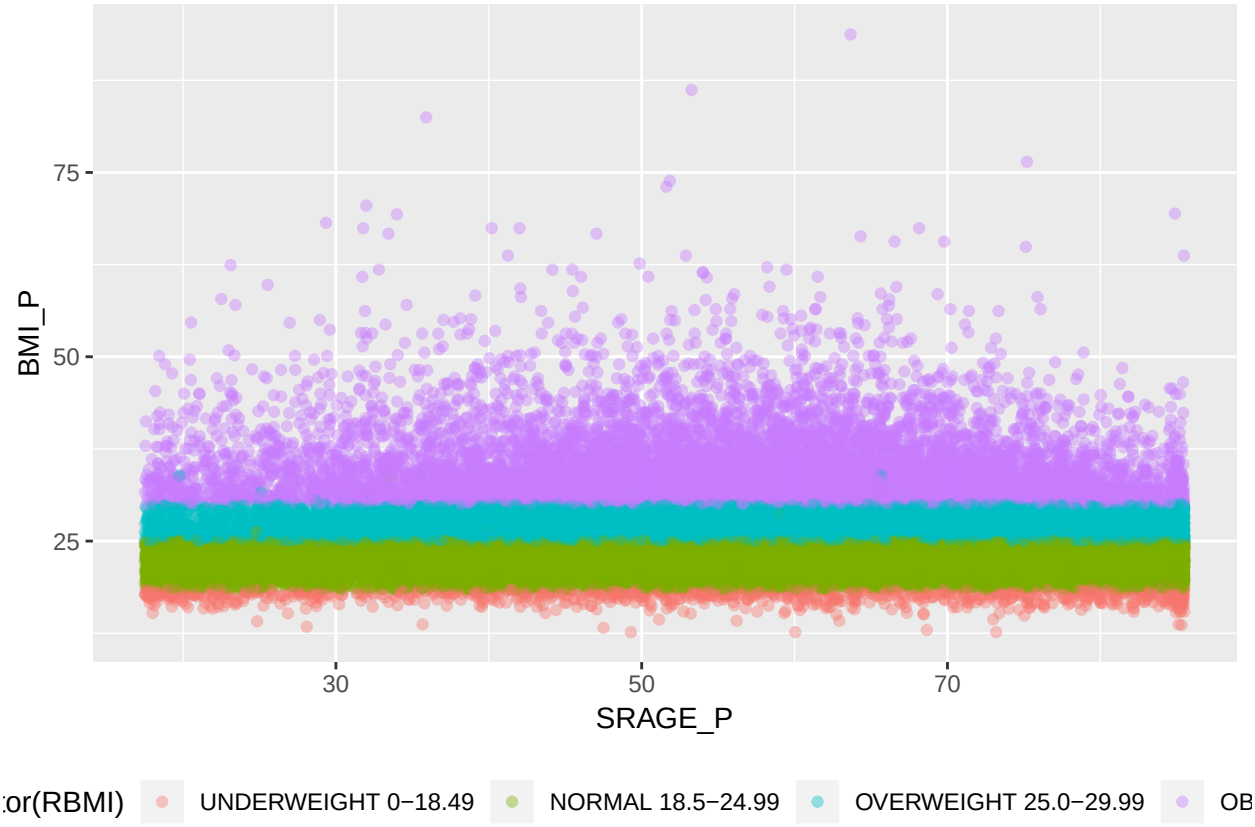

```r
diff(range(adult$SRAGE_P)) / 30
#> [1] 2.233333

# Histogram of bmi
ggplot(adult, aes(x = BMI_P)) +
  geom_histogram()
#> `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



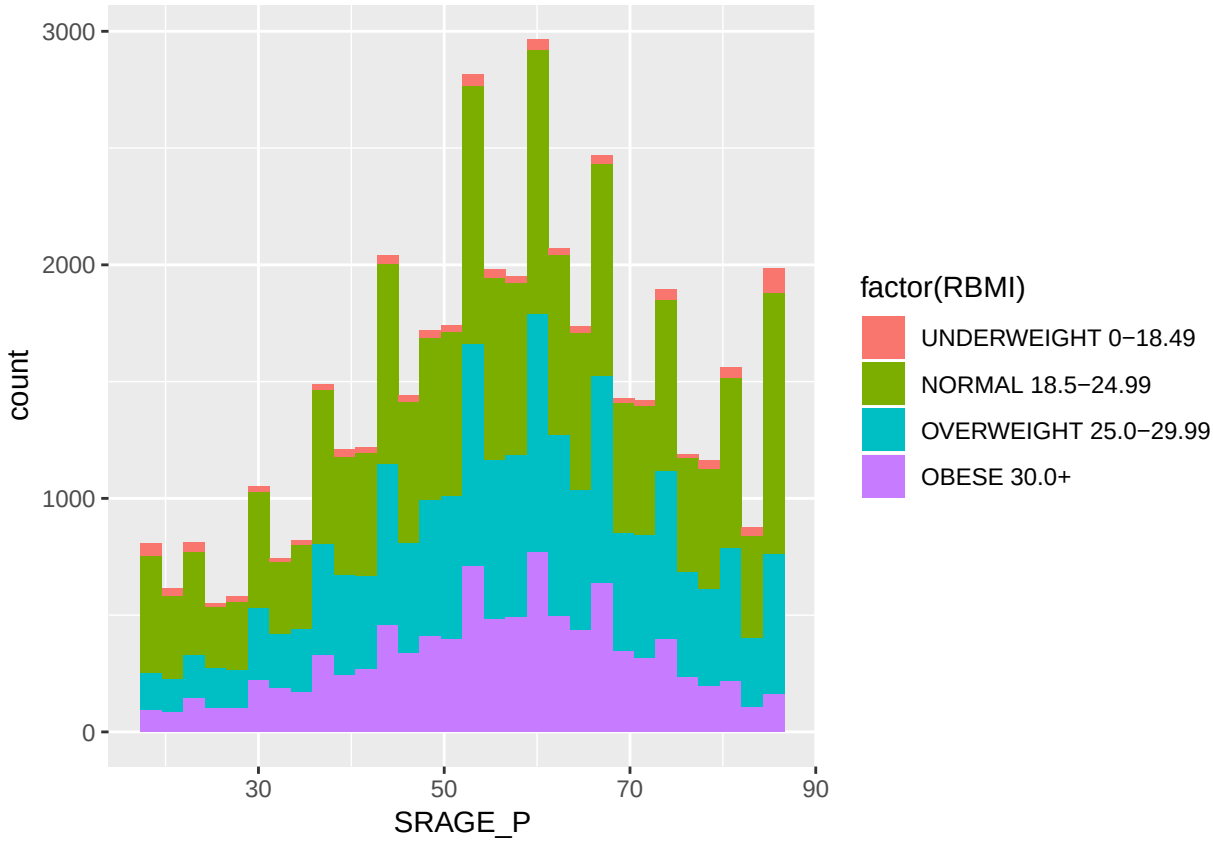

```
# Cleaning code left out
ggplot(adult, aes(x = SRAGE_P, y = BMI_P)) +
  geom_point()
```



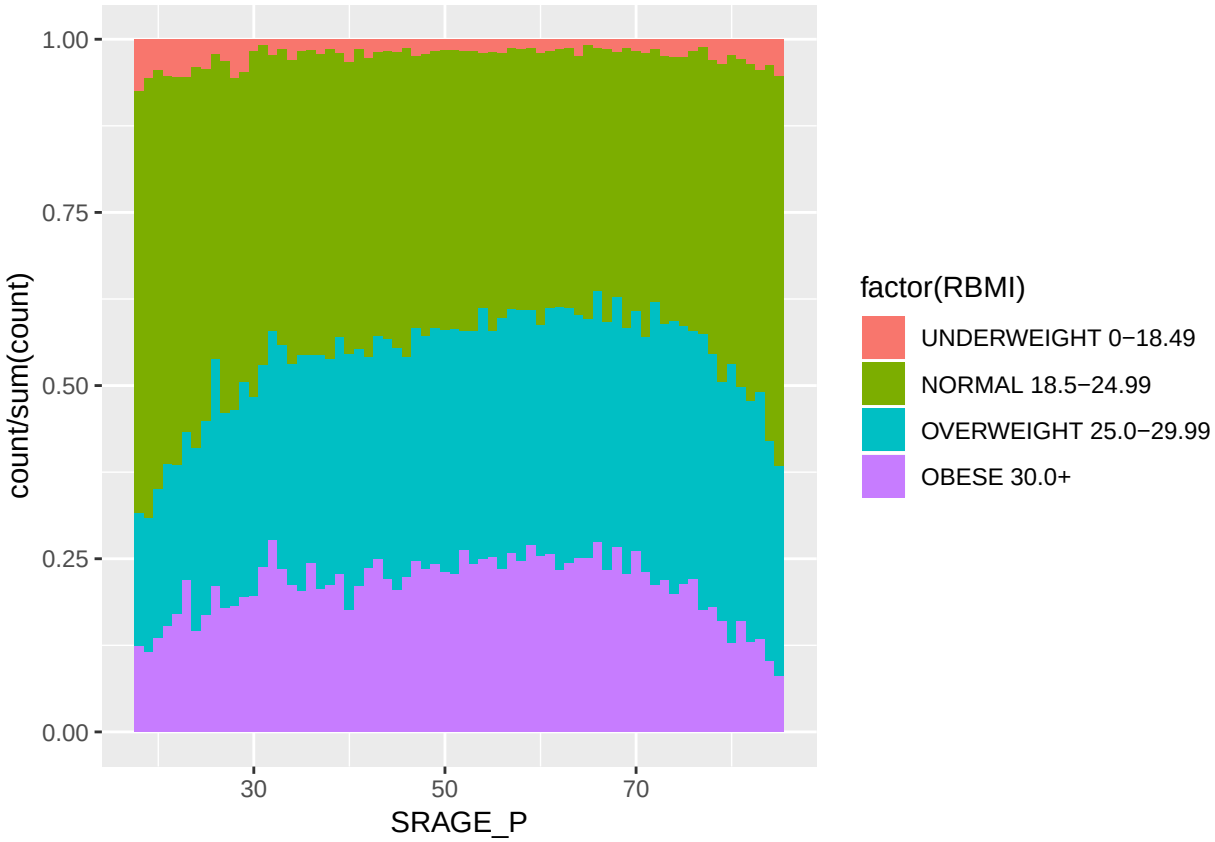

```
# BMI & age

ggplot(adult, aes(x = SRAGE_P, y = BMI_P, col = factor(RBMI))) +
  geom_point(alpha = 0.4, position = position_jitter(width = 0.5)) +
  theme(legend.position = "bottom")
```




```
# Histogram of BMI & age

ggplot(adult, aes (x = SRAGE_P, fill= factor(RBMI))) +
  geom_histogram()
#> `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
# Count histogram of BMI & age

ggplot(adult, aes (x = SRAGE_P, fill= factor(RBMI))) +
  geom_histogram(aes(y = ..count../sum(..count..)),
                 binwidth = 1, position = "fill") # left out
```



We have periodic peaks throughout our distribution. We may be tempted to naively conclude that these peaks correspond to the ages 29, 39, 49 because of a curious and as-yet unexplained biological phenomenon where people seem to stop getting older at those ages. That may be possible, but it is more likely that this is an artifact of our visualization method since the binning statistics has produced bins of width **2.2**. That doesn't seem very useful for age so we need to clean that up. For `BMI`, it turns out that we have an expected positive skew to the dataset. But we may be tempted to remove extreme values since extreme weights are likely to cause spurious results later on. We will explore `BMI` and `age` more in the exercises. For now, let's assume we have nice clean dataset and we want to proceed to compare `BMI` and `age`.

Since these are both continuous variables, you maybe be tempted to plot a scatter plot of the data. If we do that, we don't really see any interesting trends so it looks like we missed a plot so to say. Well, it turns out that `BMI` is not only continuous but also ordinal (see the table below).

Range	**Classification**
0 - 18.49	Underweight
18.5 - 24.99	Healthy-weight
25.0 - 29.99	Over-weight
30.0+	Obese

Table 2: BMI is ordinal.

This presents an obvious question about the distribution of each of these categories in each age group. If we color our scatterplot according to BMI category, we don't really get a useful picture because the range of each group is different and it is difficult to know how many of each group is present in each age group. The solution is to plot an histogram. Now, we are getting somewhere but we are still dealing with absolute counts. There is still a few steps to take before we can start answering meaningful questions. For example,

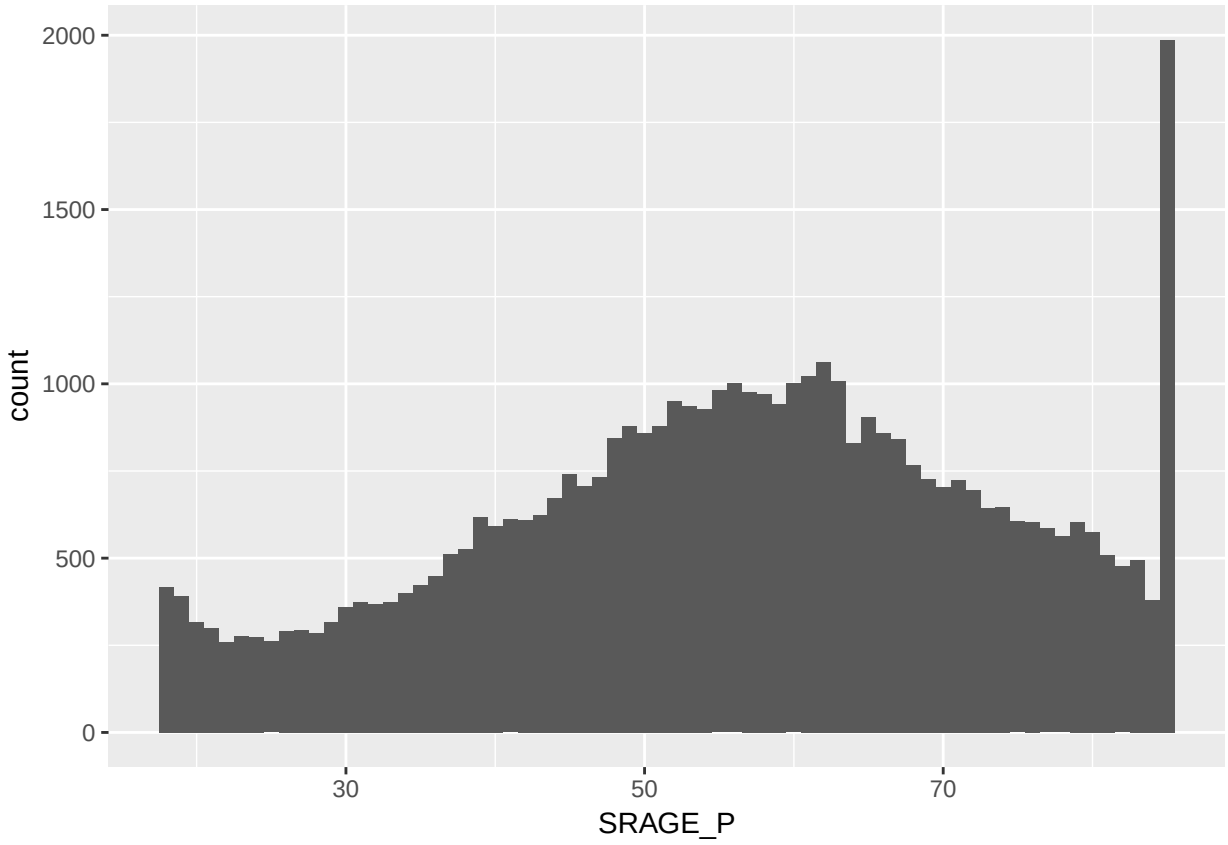
how does the proportion of each BMI category change across the age groups? To get this, we need to plot the proportion and not the absolute count of the histogram. This is the plot which will address questions of proportion and the version we will make in the exercises is colored with the appropriate color scale and have labeled the plot correctly. Do you think there is an under-representation of obesity at the higher age range? Could there be an over-representation of young under weight individuals? To answer these questions we will also want to know something about how many indivuals are in each group and also apply some statistics that will tell us about groups that are over or under represented.

Exploring Data

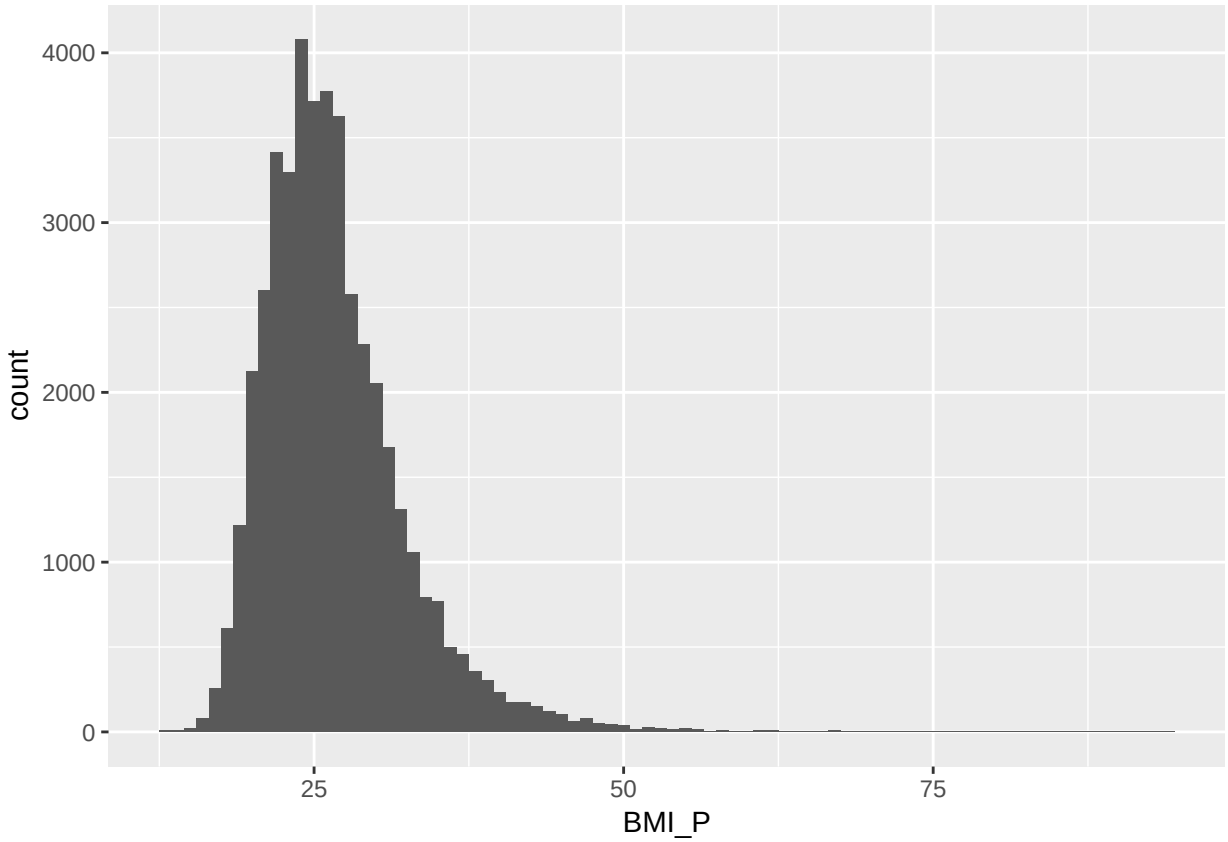
In this chapter we're going to continuously build on our plotting functions and understanding to produce a `mosaic plot` (aka **Marimekko** plot). This is a visual representation of a contingency table, comparing two categorical variables. Essentially, our question is which groups are over or under represented in our dataset. To visualize this we'll color groups according to their Pearson residuals from a chi-squared test. At the end of it all we'll wrap up our script into a flexible function so that we can look at other variables.

```
# Explore the dataset with summary and str
# str(adult)
summary(adult)
#>                     RBMI           BMI_P
#>  NOT ASCERTAINED      :    0   Min.   :12.65
#>  UNDERWEIGHT 0-18.49  : 1000   1st Qu.:22.77
#>  NORMAL 18.5-24.99    :18767   Median :25.72
#>  OVERWEIGHT 25.0-29.99:14984   Mean   :26.64
#>  OBESE 30.0+          : 9595   3rd Qu.:29.32
#>                                Max.   :93.72
#>
#>                            RACEHPR2        SRSEX          SRAGE_P
#>  LATINO                        : 5753   MALE  :18131   Min.   :18.00
#>  PACIFIC ISLANDER              :    0   FEMALE:26215   1st Qu.:44.00
#>  AMERICAN INDIAN/ALASKAN NATIVE:    0                  Median :57.00
#>  ASIAN                         : 4874                  Mean   :56.14
#>  AFRICAN AMERICAN              : 1950                  3rd Qu.:69.00
#>  WHITE                         :31769                  Max.   :85.00
#>  OTHER SINGLE/MULTIPLE RACE    :    0
#>                MARIT2                      AB1
#>  NOT ASCERTAINED  :    0   VERY GOOD      :14634
#>  MARRIED          :23352   GOOD           :12621
#>  LIVING W/ PARTNER: 2112   EXCELLENT      : 8594
#>  WID/SEP/DIV      :12519   FAIR           : 6245
#>  NEVER MARRIED    : 6363   POOR           : 2252
#>                            NOT ASCERTAINED:    0
#>                            (Other)        :    0
#>                ASTCUR                    AB51
#>  NOT ASCERTAINED  :    0   NOT ASCERTAINED:    0
#>  CURRENT ASTHMA   : 3751   DON'T KNOW     :    0
#>  NO CURRENT ASTHMA:40595   REFUSED        :    0
#>                            INAPPLICABLE   :39948
#>                            TYPE I         :  455
#>                            TYPE 2         : 3858
#>                            ANOTHER TYPE   :   85
#>                 POVLL
#>  0-99% FPL         : 5033
#>  100-199% FPL      : 7251
#>  200-299% FPL      : 6062
```

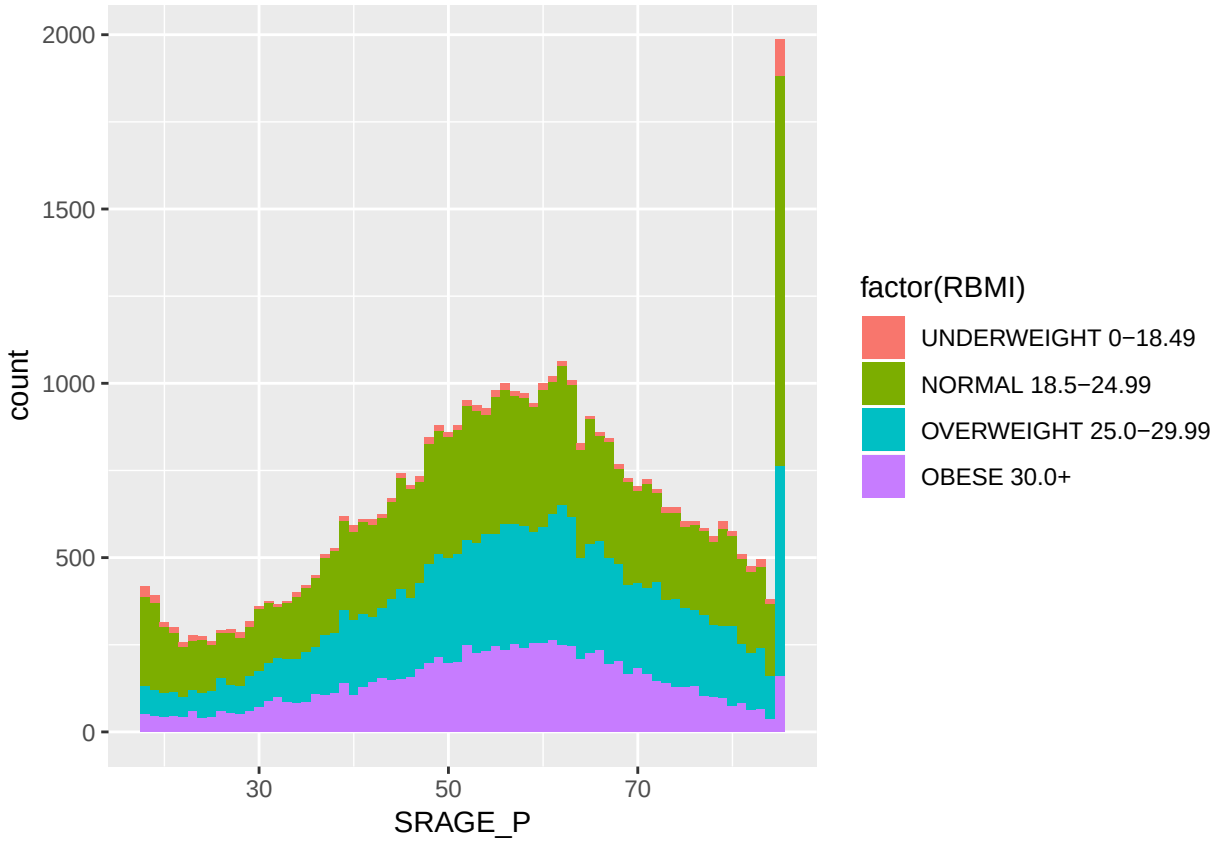
```
#>  300% FPL AND ABOVE:26000
#>
#>
#>

# Age histogram
ggplot(adult, aes(x = SRAGE_P)) +
  geom_histogram(binwidth = 1)
```




```
# BMI vggplot(adult, aes(x = SRAGE_P)) +
ggplot(adult, aes(x = BMI_P)) +
  geom_histogram(binwidth = 1)
```




```
# Age colored by BMI, binwidth = 1
ggplot(adult, aes(x = SRAGE_P, fill = factor(RBMI))) +
  geom_histogram(binwidth = 1)
```



Yes, it looks like everyone 85 and above has been categorized as 85 years old.

Data Cleaning

Now that we have an idea about our data, let's clean it up.

```
# Keep adults younger than or equal to 84
adult <- adult[adult$SRAGE_P <= 84, ]

# Keep adults with BMI at least 16 and less than 52
adult <- adult[adult$BMI_P >= 16 & adult$BMI_P < 52, ]

# Relabel the race variable
adult$RACEHPR2 <- factor(adult$RACEHPR2,
                         labels = c("Latino", "Asian", "African American", "White"))

# Relabel the BMI categories variable
adult$RBMI <- factor(adult$RBMI,
                     labels = c("Under-weight", "Normal-weight", "Over-weight", "Obese"))
```

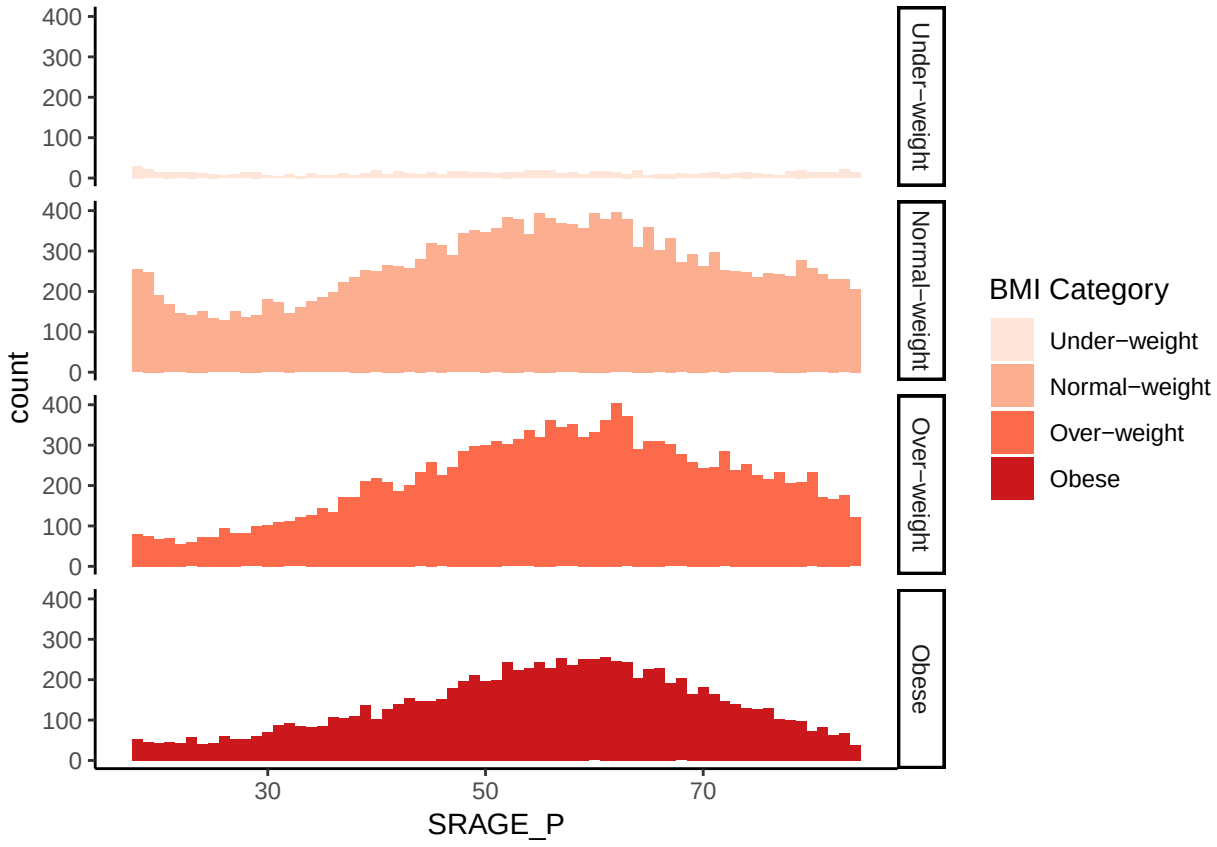
Multiple Histogram

When we introduced histograms we focused on univariate data, which is exactly what we've been doing here. However, when we want to explore distributions further there is much more we can do. For example, there are density plots, which you'll explore in the next course. For now, we'll look deeper at frequency histograms and begin developing our mosaic plots.

The adult dataset, which is cleaned up by now, is available in the workspace for you.

Two layers have been pre-defined for you: BMI_fill is a scale layer which we can add to a ggplot() command using +: ggplot(...) + BMI_fill. fix_strips is a theme() layer to make nice facet titles.

```
# The color scale used in the plot
BMI_fill <- scale_fill_brewer("BMI Category", palette = "Reds")

# Theme to fix category display in faceted plot
fix_strips <- theme(strip.text.y =
                      element_text(angle = 0, hjust = 0, vjust = 0.1, size = 14),
                    strip.background = element_blank(),
                    legend.position = "none")

# Histogram, add BMI_fill and customizations
ggplot(adult, aes (x = SRAGE_P, fill= RBMI)) +
  geom_histogram(binwidth = 1) +
  BMI_fill +
  fix_strips +
  facet_grid(RBMI ~ .) +
  theme_classic()
```



Great! This looks good already.

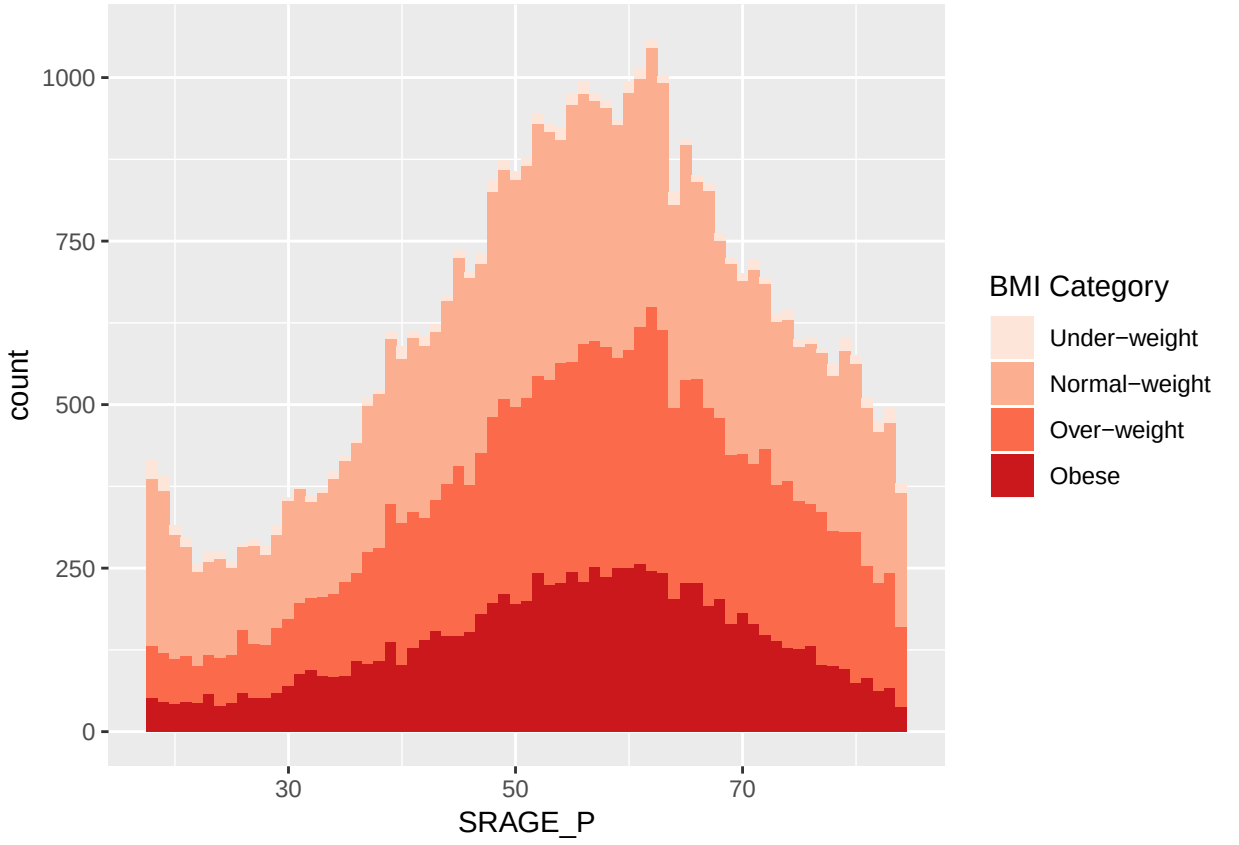
Alternatives

In the previous exercise we looked at different ways of showing the absolute count of multiple histograms. This is fine, but density would be a more useful measure if we wanted to see how the frequency of one variable

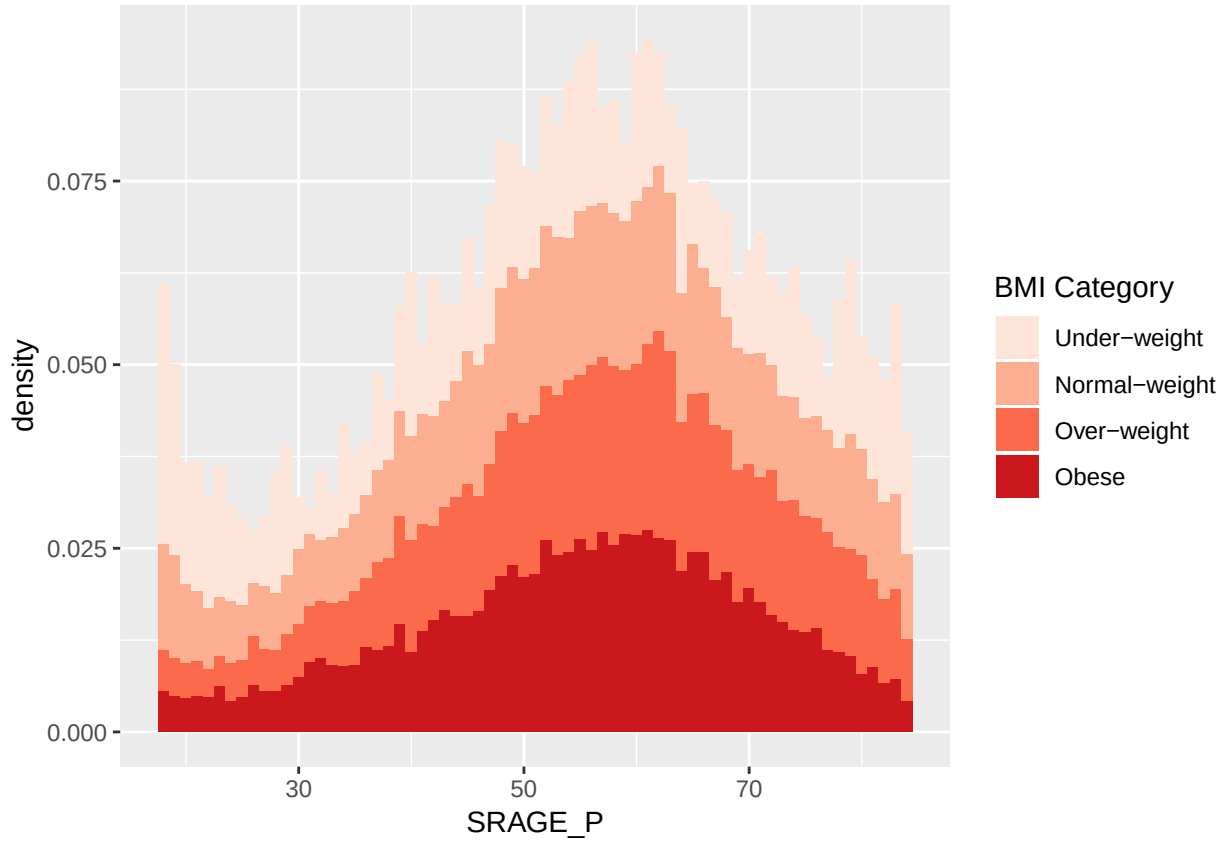
changes across another. However, there are some difficulties here, so let's take a closer look at different plots.

The clean adult dataset is available, as is the BMI_fill color palette. The first plot simply shows a histogram of counts, without facets, without modified themes. It's denoted `Plot 1`.

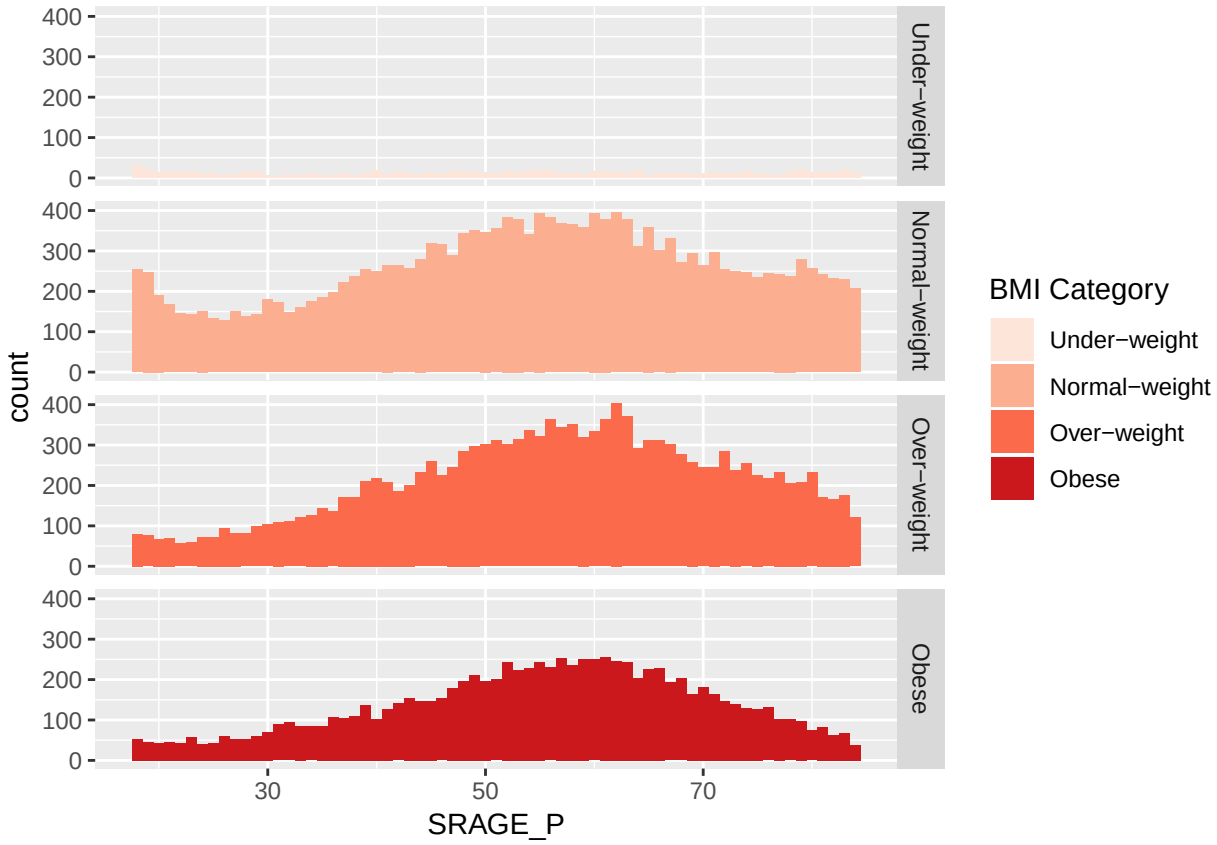
```
# Plot 1 - Count histogram
ggplot(adult, aes (x = SRAGE_P, fill= factor(RBMI))) +
  geom_histogram(binwidth = 1) +
  BMI_fill
```



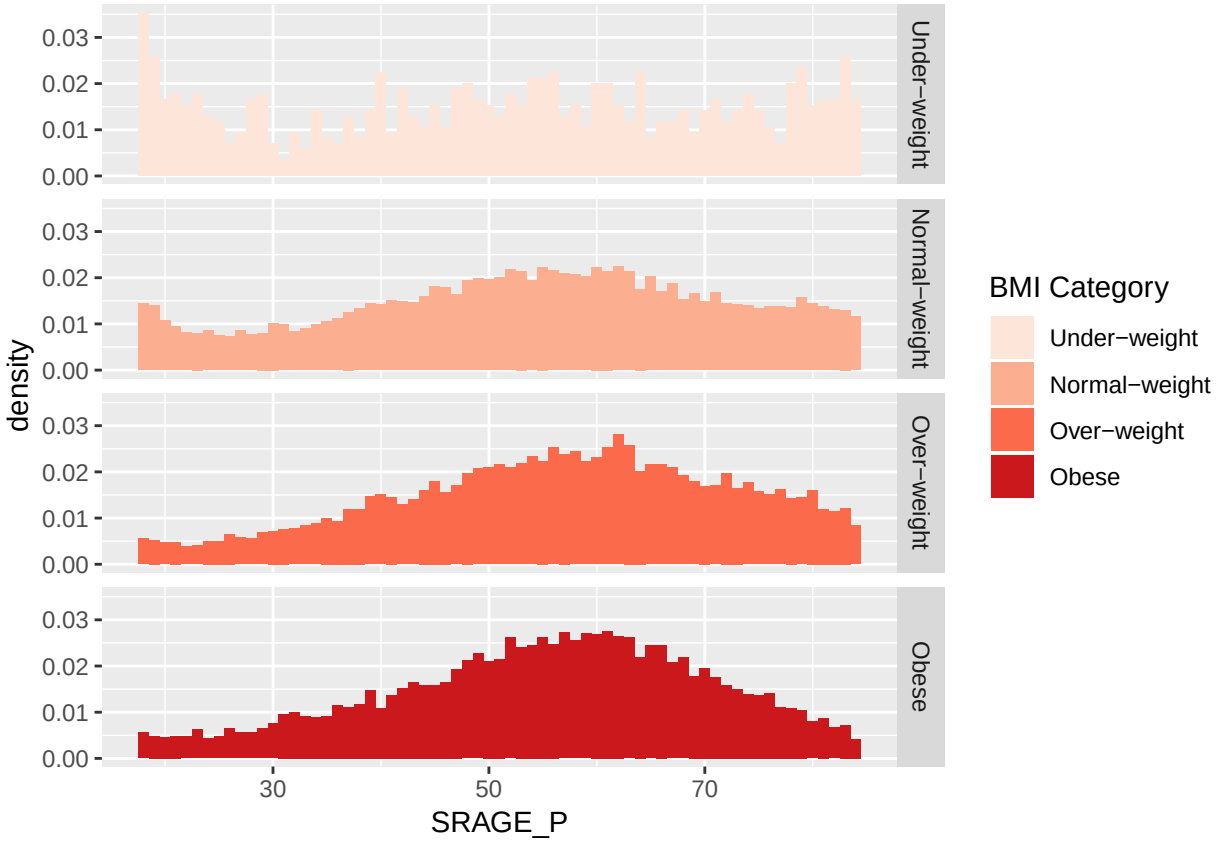

```
# Plot 2 - Density histogram
ggplot(adult, aes (x = SRAGE_P, fill= factor(RBMI))) +
  geom_histogram(aes(y = ..density..), binwidth = 1) +
  BMI_fill
```



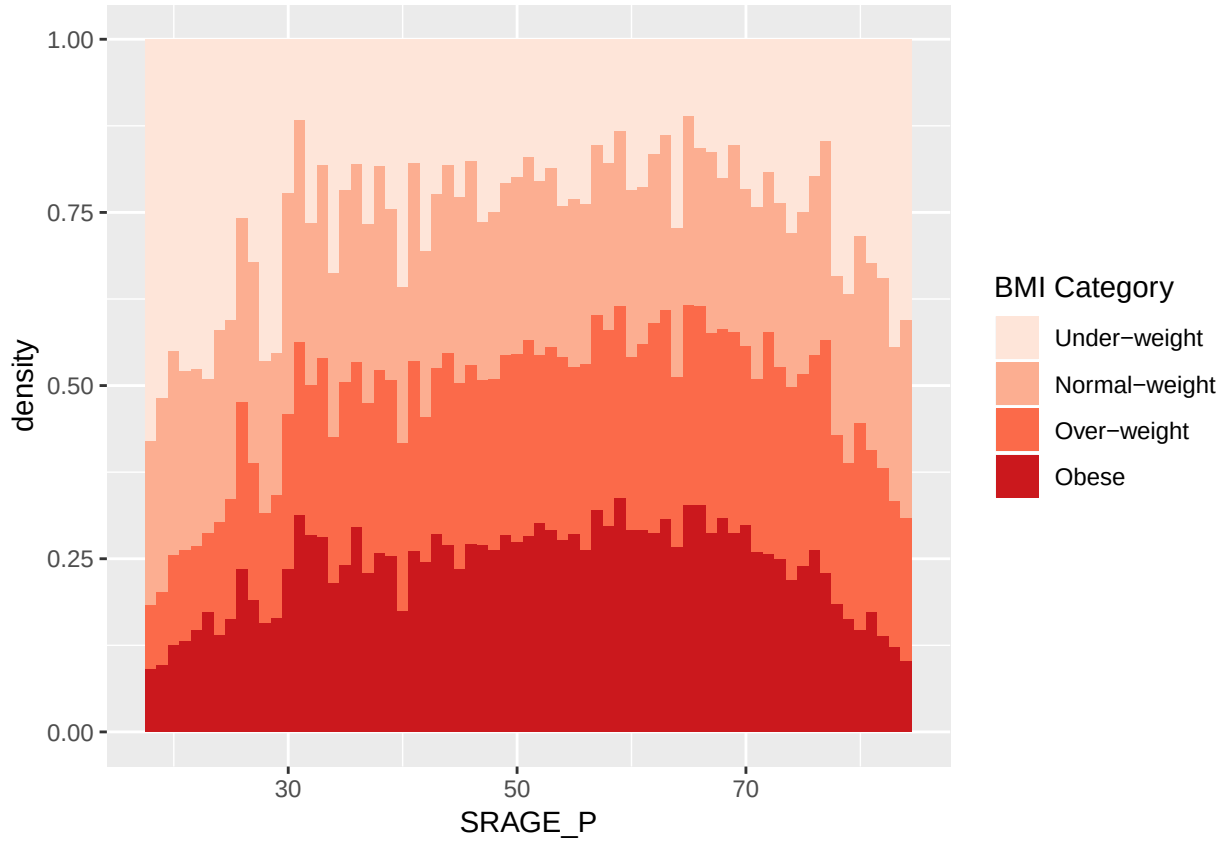

```
# Plot 3 - Faceted count histogram
ggplot(adult, aes (x = SRAGE_P, fill= factor(RBMI))) +
  geom_histogram(binwidth = 1) +
  BMI_fill +
  facet_grid(RBMI ~ .)
```



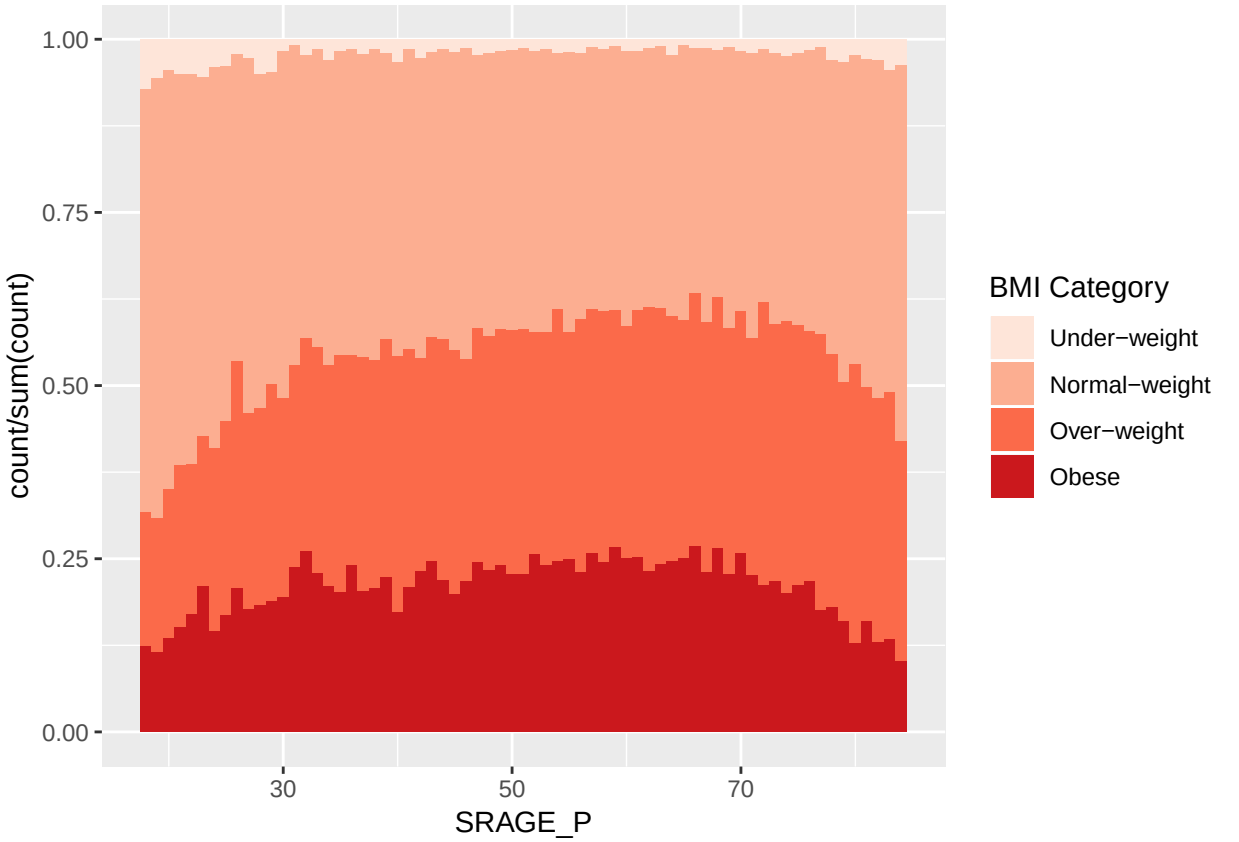

```
# Plot 4 - Faceted density histogram
ggplot(adult, aes (x = SRAGE_P, fill= factor(RBMI))) +
  geom_histogram(aes(y = ..density..), binwidth = 1) +
  BMI_fill +
  facet_grid(RBMI ~ .)
```




```
# Plot 5 - Density histogram with position = "fill"
ggplot(adult, aes (x = SRAGE_P, fill= factor(RBMI))) +
  geom_histogram(aes(y = ..density..), binwidth = 1, position = "fill") +
  BMI_fill
```




```
# Plot 6 - The accurate histogram
ggplot(adult, aes (x = SRAGE_P, fill= factor(RBMI))) +
  geom_histogram(aes(y = ..count../sum(..count..)), binwidth = 1, position = "fill") +
  BMI_fill
```



If we want to facet this plot, we'll again run into problems (give it a try), but we'll solve that in the next exercise.

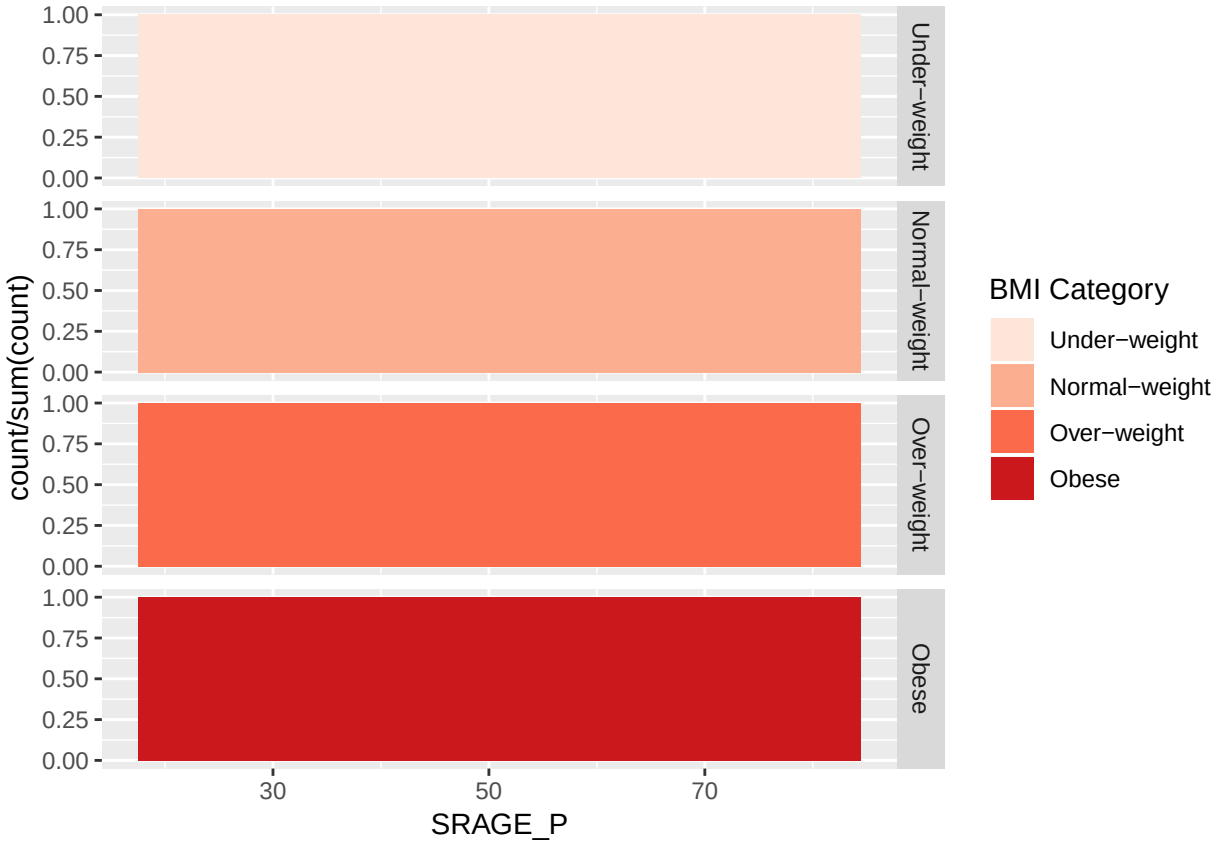
Do Things Manually

In the previous exercise we looked at how to produce a frequency histogram when we have many sub-categories. The problem here is that this can't be facetted because the calculations occur on the fly inside ggplot2.

To overcome this we're going to calculate the proportions outside ggplot2. This is the beginning of our flexible script for a mosaic plot.

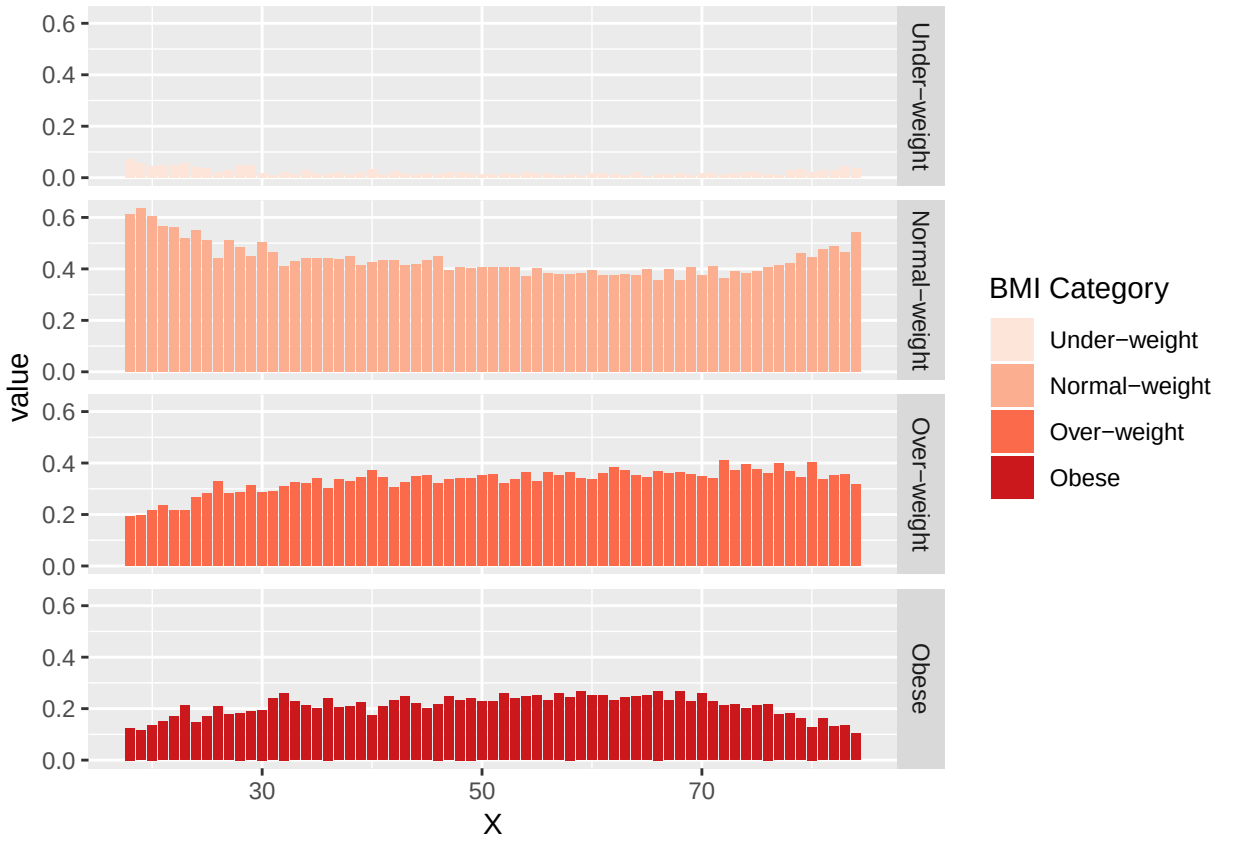
The dataset adult and the BMI_fill object from the previous exercise have been carried over for you. Code that tries to make the accurate frequency histogram facetted is available. You should understand these commands by now.

```
# An attempt to facet the accurate frequency histogram from before (failed)
ggplot(adult, aes (x = SRAGE_P, fill= factor(RBMI))) +
  geom_histogram(aes(y = ..count../sum(..count..)), binwidth = 1, position = "fill") +
  BMI_fill +
  facet_grid(RBMI ~ .)
```




```
# Create DF with table()
DF <- table(adult$RBMI, adult$SRAGE_P)

# Use apply on DF to get frequency of each group
DF_freq <- apply(DF, 2, function(x)x/sum(x))

# Load reshape2 and use melt on DF to create DF_melted
library(reshape2)
#>
#> Attaching package: 'reshape2'
#> The following object is masked from 'package:tidyr':
#>
#>     smiths

DF_melted <- melt(DF_freq)

# Note: Here we use reshape2 instead of the more current tidyr because
# reshape2::melt() allows us to work directly on a table.
# tidyr::gather() requires a data frame.

# Change names of DF_melted
names(DF_melted) <- c("FILL", "X", "value")

# Add code to make this a faceted plot
ggplot(DF_melted, aes(x = X, y = value, fill = FILL)) +
  geom_col(position = "stack") +
```

```
BMI_fill +
facet_grid(FILL ~ .) # Facets
```




```
# Note that we use geom_col() now, this is just a short-cut to geom_bar(stat = "identity")
```

Good job! This one was a bit tricky!

Mosaic Plots

In this video we will keep working on the CHIS dataset and start asking more detailed questions about our dataset. We ended the last video with a plot depicting the distribution of BMI category across ages. In our filtered dataframe we have 67 ages each represented by 4 BMI categories so in total we have 268 individual groups. Essentially, what we are interested in looking for are unsual patterns in the dataset.

Marimekko/Mosaic Plot

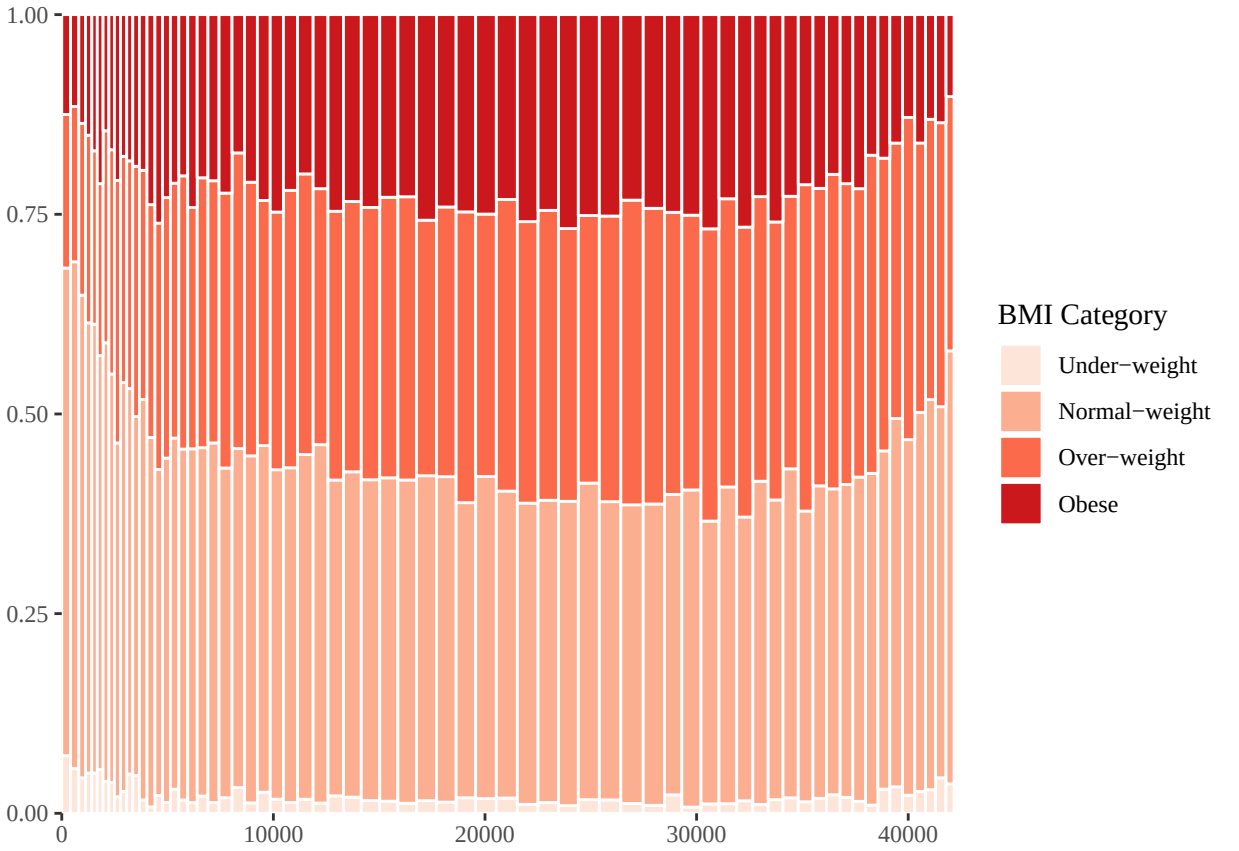
In the previous exercise we looked at different ways of showing the frequency distribution within each BMI category. This is all well and good, but the absolute number of each age group also has an influence on if we will consider something as over-represented or not. Here, we will proceed to change the widths of the bars to show us something about the n in each group.

This will get a bit more involved, because the aim is not to draw bars, but rather rectangles, for which we can control the widths. You may have already realized that bars are simply rectangles, but we don't have easy access to the `xmin` and `xmax` aesthetics, but in **`geom_rect()`** we do! Likewise, we also have access to `ymin` and `ymax`. So we're going to draw a box for every one of our 268 distinct groups of BMI category and age.

The clean adult dataset, as well as `BMI_fill`, are already available. Instead of running **`apply()`** like

in the previous exercise, the contingency table has already been transformed to a data frame using **as.data.frame.matrix()**.

```
# The initial contingency table
DF <- as.data.frame.matrix(table(adult$SRAGE_P, adult$RBMI))

# groupSum, containing the sum of each row in the DF.
# Use rowSums() to calculate this.
# groupSum represents the total number of individuals in each age group.

# Create groupSum, xmax and xmin columns
DF$groupSum <- rowSums(DF)
DF$xmax <- cumsum(DF$groupSum)
DF$xmin <- DF$xmax - DF$groupSum
# The groupSum column needs to be removed; don't remove this line
DF$groupSum <- NULL

# Copy row names to variable X
DF$X <- row.names(DF)

# Melt the dataset
library(reshape2)
DF_melted <- melt(DF, id.vars = c("X", "xmin", "xmax"), variable.name = "FILL")

# dplyr call to calculate ymin and ymax - don't change
library(dplyr)
DF_melted <- DF_melted %>%
  group_by(X) %>%
  mutate(ymax = cumsum(value/sum(value)),
         ymin = ymax - value/sum(value))

# Plot rectangles - don't change
library(ggthemes)
ggplot(DF_melted, aes(ymin = ymin,
                      ymax = ymax,
                      xmin = xmin,
                      xmax = xmax,
                      fill = FILL)) +
  geom_rect(colour = "white") +
  scale_x_continuous(expand = c(0,0)) +
  scale_y_continuous(expand = c(0,0)) +
  BMI_fill +
  theme_tufte()
```



Adding Statistics

In the previous exercise we generated a plot where each individual bar was plotted separately using rectangles (shown in the viewer). This means we have access to each piece and we can apply different fill parameters.

So let's make some new parameters. To get the Pearson residuals, we'll use the **`chisq.test()`** function.

The data frames `adult` and `DF_melted`, as well as the object `BMI_fill` that you created throughout this chapter, are all still available. The reshape2 package is already loaded.

```
# Perform chi.sq test (RBMI and SRAGE_P)
results <- chisq.test(table(adult$RBMI, adult$SRAGE_P))

# Melt results$residuals and store as resid
resid <- melt(results$residuals)

# Change names of resid
names(resid) <- c("FILL", "X", "residual")

# merge the two datasets:
DF_all <- merge(DF_melted, resid)

# Update plot command
library(ggthemes)
ggplot(DF_melted, aes(ymin = ymin,
                      ymax = ymax,
                      xmin = xmin,
```

```
                  xmax = xmax,
                  fill = FILL)) +
  geom_rect() +
  scale_fill_gradient2() +
  scale_x_continuous(expand = c(0,0)) +
  scale_y_continuous(expand = c(0,0)) +
  theme_tufte()
```

Adding Text

Since we're not coloring according to BMI, we have to add group (and x axis) labels manually. Our goal is the plot in the viewer.

For this we'll use the label aesthetic inside geom_text(). The actual labels are found in the FILL (BMI category) and X (age) columns in the DF_all data frame. (Additional attributes have been set inside geom_text() in the exercise for you).

The labels will be added to the right (BMI category) and top (age) inner edges of the plot. (We could have also added margin text, but that is a more advanced topic that we'll encounter in the third course. This will be a suitable solution for the moment.)

The first two commands show how we got the the four positions for the y axis labels. First, we got the position of the maximum xmax values, i.e. at the very right end, stored as index. We want to calculate the half difference between each pair of ymax and ymin (e.g. (ymax - ymin)/2) at these index positions, then add this value to the ymin value. These positions are stored in the variable yposn.

We'll begin with the plot thus far, stored as object p. In the sample code, %+% DF_all refreshes the plot's dataset with the extra columns.

```
# Plot so far
p

# Position for labels on y axis (don't change)
index <- DF_all$xmax == max(DF_all$xmax)
DF_all$yposn <- DF_all$ymin[index] + (DF_all$ymax[index] - DF_all$ymin[index])/2

# Plot 1: geom_text for BMI (i.e. the fill axis)
p1 <- p %+% DF_all +
  geom_text(aes(x = max(xmax),
                y = yposn,
                label = FILL),
            size = 3, hjust = 1,
            show.legend  = FALSE)
p1

# Plot 2: Position for labels on x axis
DF_all$xposn <- DF_all$xmin + (DF_all$xmax - DF_all$xmin)/2

# geom_text for ages (i.e. the x axis)
p1 %+% DF_all +
  geom_text(aes(x = xposn, label = X),
            y = 1, angle = 90,
            size = 3, hjust = 1,
            show.legend = FALSE)
```

Generalizations

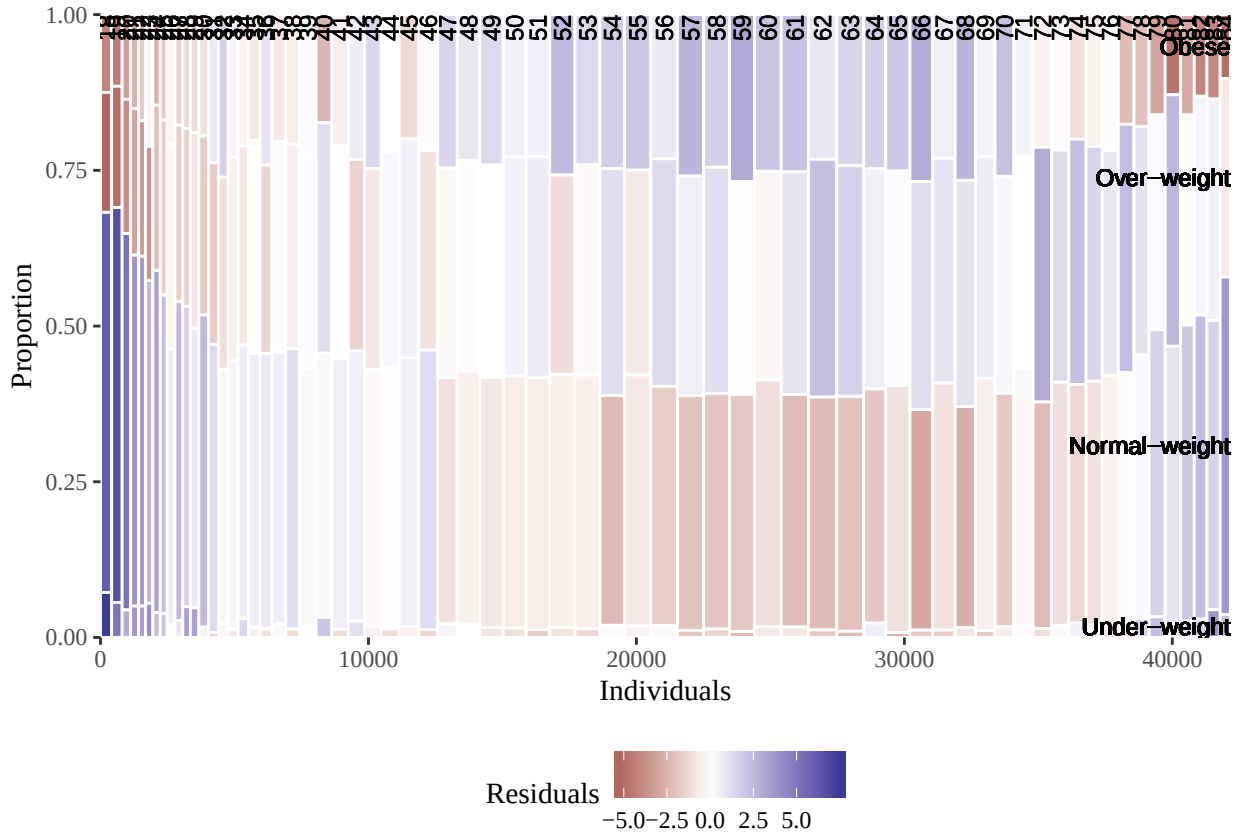
Now that you've done all the steps necessary to make our mosaic plot, you can wrap all the steps into a single function that we can use to examine any two variables of interest in our data frame (or in any other data frame for that matter). For example, we can use it to examine the Vocab data frame we saw earlier in this course.

You've seen all the code in our function, so there shouldn't be anything surprising there. Notice that the function takes multiple arguments, such as the data frame of interest and the variables that you want to create the mosaic plot for. None of the arguments have default values, so you'll have to specify all three if you want the mosaicGG() function to work.

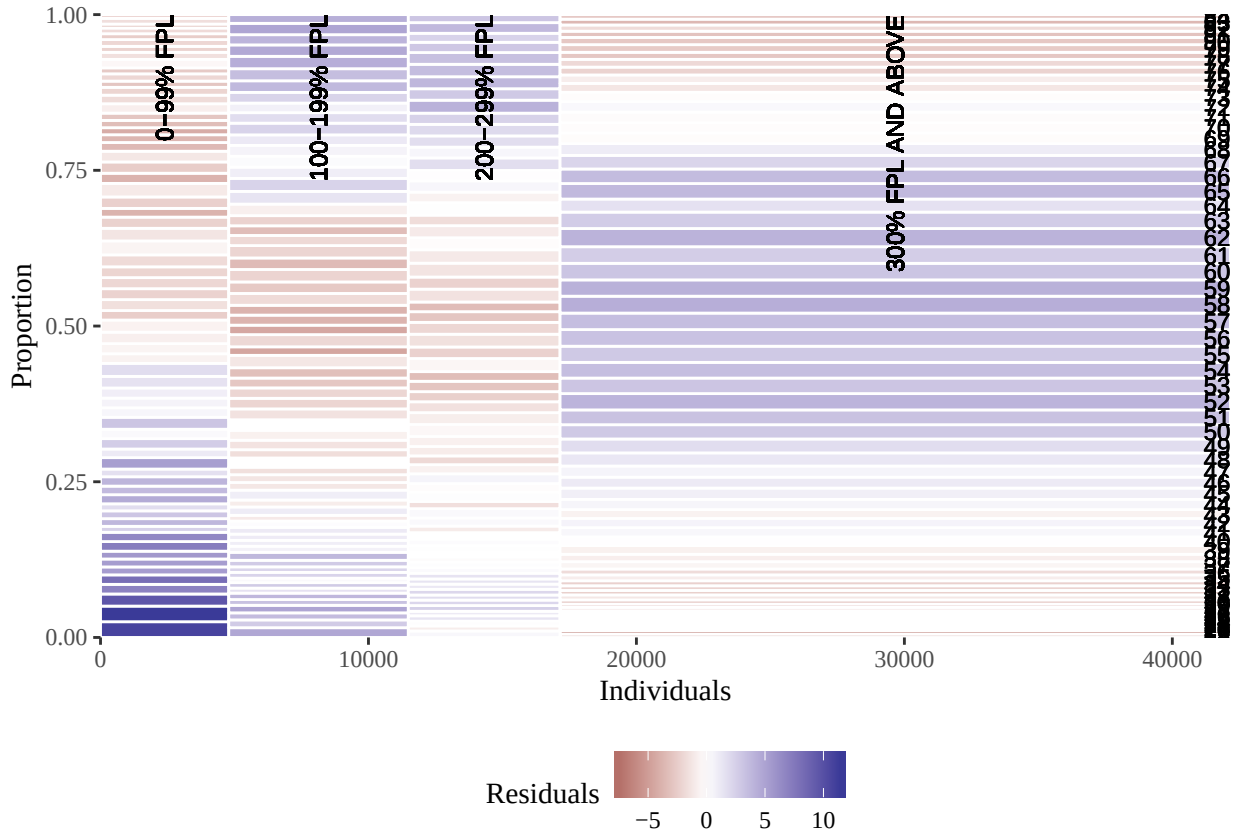
Start by going through the code and see if you understand the function's implementation.

```
mosaicGG <- function(data, X, FILL) {
  # Proportions in raw data
  DF <- as.data.frame.matrix(table(data[[X]], data[[FILL]]))
  DF$groupSum <- rowSums(DF)
  DF$xmax <- cumsum(DF$groupSum)
  DF$xmin <- DF$xmax - DF$groupSum
  DF$X <- row.names(DF)
  DF$groupSum <- NULL
  DF_melted <- melt(DF, id = c("X", "xmin", "xmax"), variable.name = "FILL")
  DF_melted <- DF_melted %>%
    group_by(X) %>%
    mutate(ymax = cumsum(value/sum(value)),
           ymin = ymax - value/sum(value))

  # Chi-sq test
  results <- chisq.test(table(data[[FILL]], data[[X]])) # fill and then x
  resid <- melt(results$residuals)
  names(resid) <- c("FILL", "X", "residual")

  # Merge data
  DF_all <- merge(DF_melted, resid)

  # Positions for labels
  DF_all$xposn <- DF_all$xmin + (DF_all$xmax - DF_all$xmin)/2
  index <- DF_all$xmax == max(DF_all$xmax)
  DF_all$yposn <- DF_all$ymin[index] + (DF_all$ymax[index] - DF_all$ymin[index])/2

  # Plot
  g <- ggplot(DF_all, aes(ymin = ymin,  ymax = ymax, xmin = xmin,
                          xmax = xmax, fill = residual)) +
  geom_rect(col = "white") +
  geom_text(aes(x = xposn, label = X),
            y = 1, size = 3, angle = 90, hjust = 1, show.legend = FALSE) +
  geom_text(aes(x = max(xmax),  y = yposn, label = FILL),
            size = 3, hjust = 1, show.legend = FALSE) +
  scale_fill_gradient2("Residuals") +
  scale_x_continuous("Individuals", expand = c(0,0)) +
  scale_y_continuous("Proportion", expand = c(0,0)) +
  theme_tufte() +
  theme(legend.position = "bottom")
  print(g)
}
```

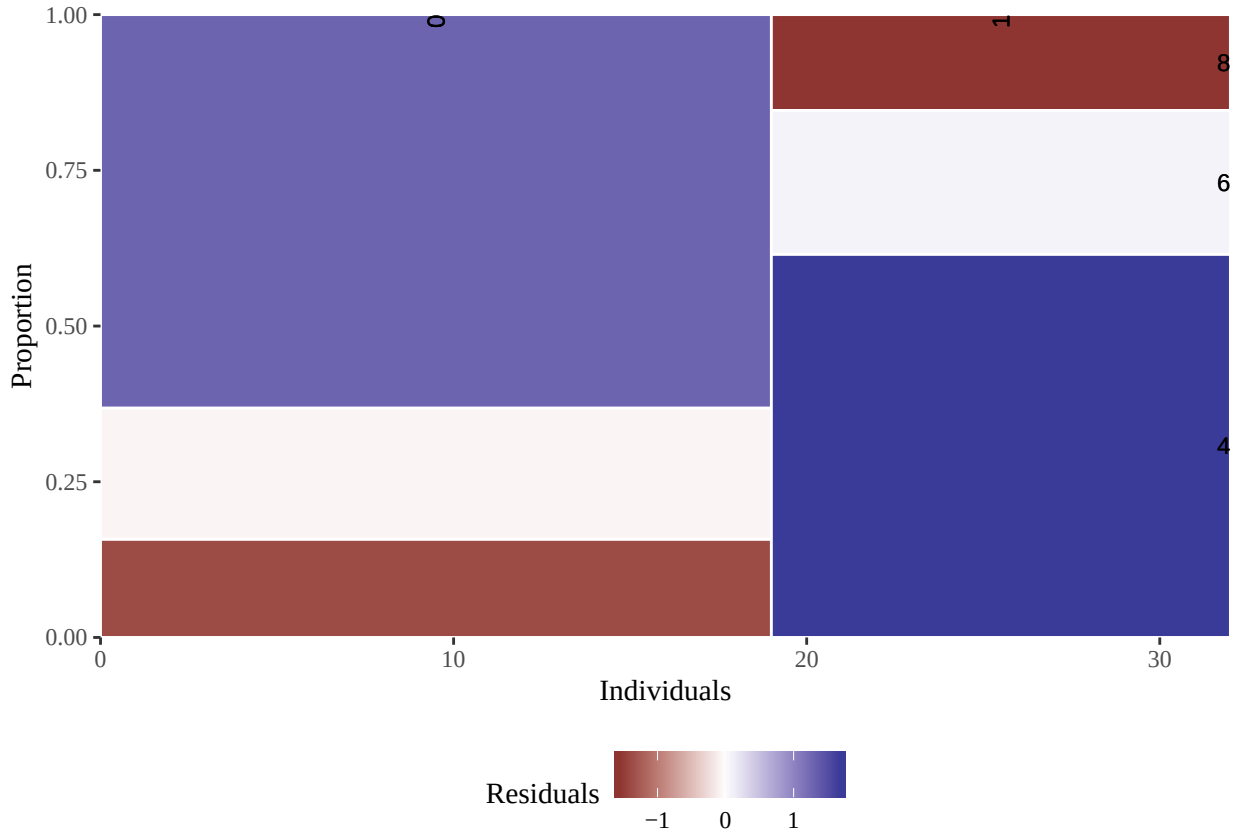
```r
# Script generalized into a function
mosaicGG
#> function(data, X, FILL) {
#>   # Proportions in raw data
#>   DF <- as.data.frame.matrix(table(data[[X]], data[[FILL]]))
#>   DF$groupSum <- rowSums(DF)
#>   DF$xmax <- cumsum(DF$groupSum)
#>   DF$xmin <- DF$xmax - DF$groupSum
#>   DF$X <- row.names(DF)
#>   DF$groupSum <- NULL
#>   DF_melted <- melt(DF, id = c("X", "xmin", "xmax"), variable.name = "FILL")
#>   DF_melted <- DF_melted %>%
#>     group_by(X) %>%
#>     mutate(ymax = cumsum(value/sum(value)),
#>            ymin = ymax - value/sum(value))
#>
#>   # Chi-sq test
#>   results <- chisq.test(table(data[[FILL]], data[[X]])) # fill and then x
#>   resid <- melt(results$residuals)
#>   names(resid) <- c("FILL", "X", "residual")
#>
#>   # Merge data
#>   DF_all <- merge(DF_melted, resid)
#>
#>   # Positions for labels
#>   DF_all$xposn <- DF_all$xmin + (DF_all$xmax - DF_all$xmin)/2
#>   index <- DF_all$xmax == max(DF_all$xmax)
#>   DF_all$yposn <- DF_all$ymin[index] + (DF_all$ymax[index] - DF_all$ymin[index])/2
#>
#>   # Plot
#>   g <- ggplot(DF_all, aes(ymin = ymin,  ymax = ymax, xmin = xmin,
#>                           xmax = xmax, fill = residual)) +
#>   geom_rect(col = "white") +
#>   geom_text(aes(x = xposn, label = X),
#>             y = 1, size = 3, angle = 90, hjust = 1, show.legend = FALSE) +
#>   geom_text(aes(x = max(xmax),  y = yposn, label = FILL),
#>             size = 3, hjust = 1, show.legend = FALSE) +
#>   scale_fill_gradient2("Residuals") +
#>   scale_x_continuous("Individuals", expand = c(0,0)) +
#>   scale_y_continuous("Proportion", expand = c(0,0)) +
#>   theme_tufte() +
#>   theme(legend.position = "bottom")
#>   print(g)
#> }

# BMI described by age (as previously seen)
mosaicGG(adult, X = "SRAGE_P", FILL = "RBMI")
```



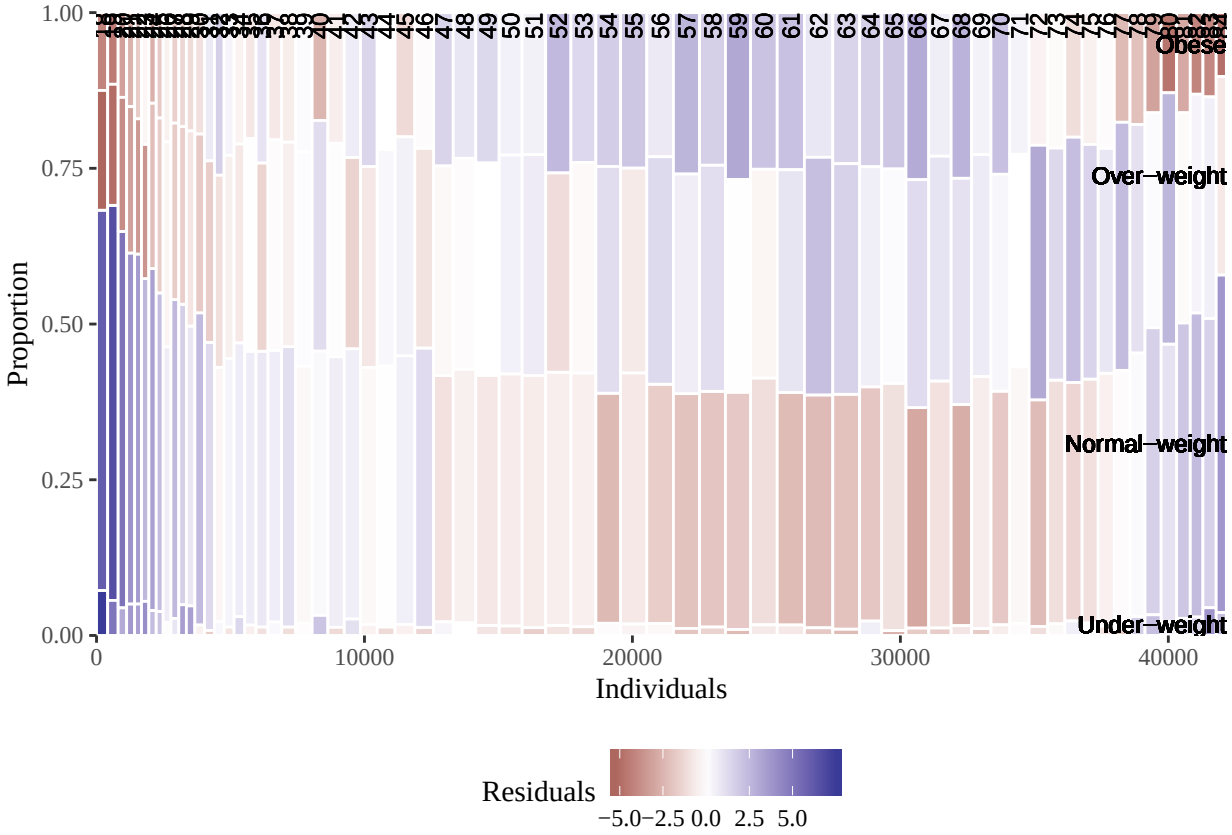

```
# Poverty described by age
mosaicGG(adult, X = "POVLL", FILL = "SRAGE_P")
```




```
# mtcars: am described by cyl
mosaicGG(mtcars, "am", "cyl")
#> Warning in chisq.test(table(data[[FILL]], data[[X]])): Chi-squared
#> approximation may be incorrect
```




```
# Vocab: vocabulary described by education
library(carData)
mosaicGG(data = adult, X = "SRAGE_P", FILL = "RBMI")
```


1.00
0.75
0.50
0.25
0.00
Proportion
0
10000
20000
30000
40000
Individuals
Obese
Over−weight
Normal−weight
Under−weight
Residuals
−5.0 −2.5 0.0 2.5 5.0