Data Visualization

R Training for NISR

Luiza Andrade, Leonardo Viotti & Rob Marty

Aug. 2018

Outline

Introduction

In this session, you'll learn how to use R to produce insightful, meaningful and (hopefully) nice-looking graphs. In particular, you'll use a package called `ggplot2`.

Similarly, to the previous session, you can find some references at the end of this presentation that include a more comprehensive discussions on data visualization.

Introduction

Before we start, let's make sure we're all set:

1. Make sure you the packages `ggplot2` and `plotly` are installed and loaded.
2. Load the `lwh_clean.csv` data set that were created in the last session.
   - Run the Master script to load all the necessary packages and set file paths.
   - Remeber to disable the `INSTALL_PACKAGES` switch in the Master if you already installed them. This will save you a lot of time.

Introduction

```r
# Install packages
install.packages(c("ggplot2","plotly"),
                 dependencies = TRUE)

# Load packages
library(ggplot2)
library(plotly)

# Load CSV data
projectFolder  <- "YOUR/FOLDER/PATH"
finalData <- file.path(projectFolder,
                       "DataWork","DataSets","Final")
lwh <- read.csv(file.path(finalData,"lwh_clean.csv"),
                header = TRUE)
```

Introduction

There are usually two distinct uses for plots:

1. **Exploratory analysis:** Quickly visualize your data in an insightful way
   - We'll do this using R's base plot, that allows you to quickly create some basic plots
2. **Publication/Reporting:** Make pretty graphs for a presentation, a report, a dashboard. or to just show your boss something other than the laziest graph you could come up with
   - We'll do this using `ggplot`, a flexible and powerful tool to plot beautiful graphs
   - `ggplot`'s syntax is more complicated, but it's easier to make your graphs look good with it

Outline

Exploratory graphs - Base plot

For exploratory plots, we're going to use Base plot. It is easy to use and can produce useful graphs with very few lines of code.

Exploratory graphs - Base plot

Exercise 1:

Plot the `lwh_simp` data set you constructed in the previous session using the `plot()` function.

1. If you don't have the code from the previous session. Here it is:

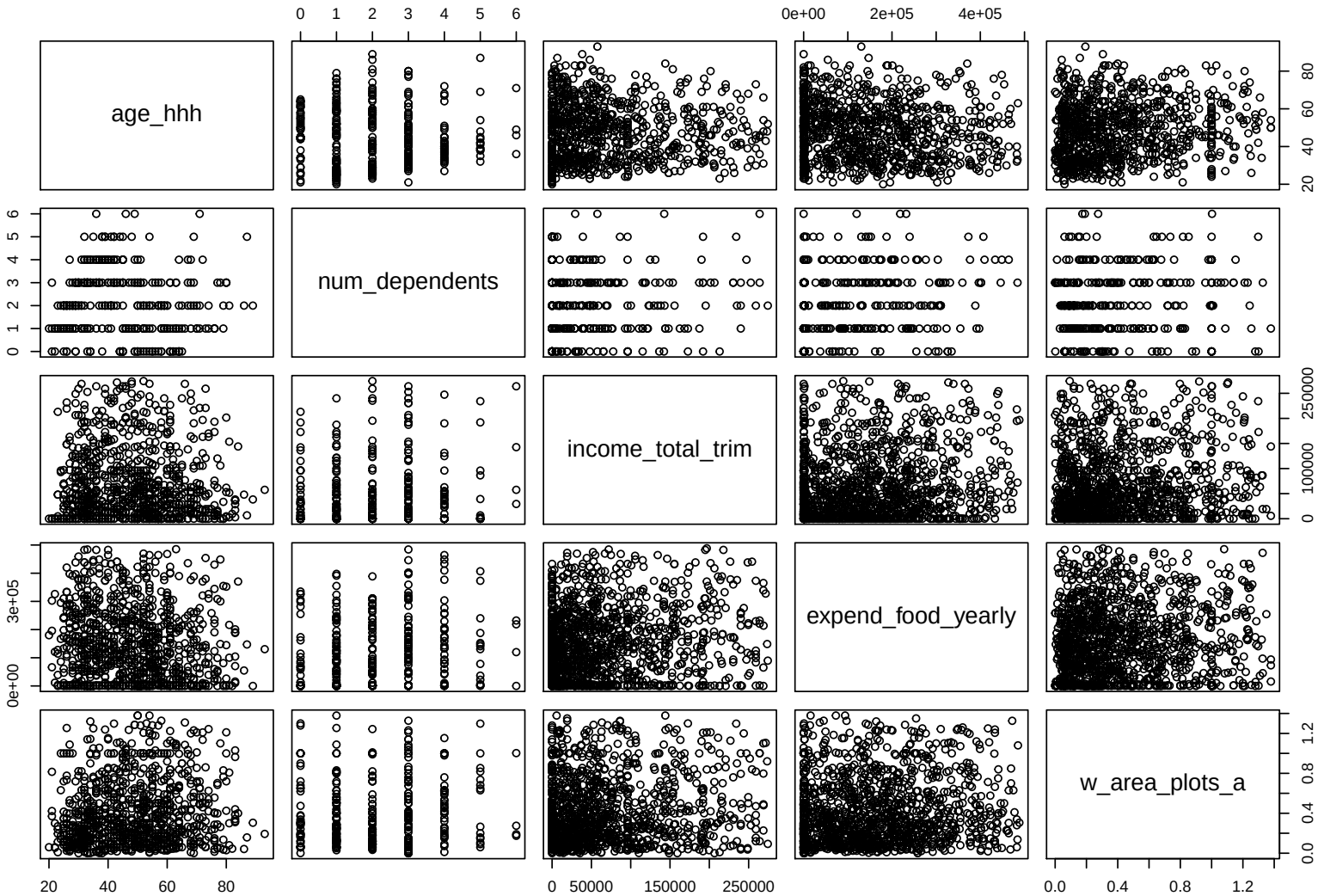
```
covariates <- c("age_hhh",
                "num_dependents",
                "income_total_trim",
                "expend_food_yearly",
                "w_area_plots_a")

lwh_simp <- lwh[,covariates]
```

2. Now, pass the name of the data set as the only argument for the plot function

Exploratory graphs - Base plot

```
plot(lwh_simp)
```



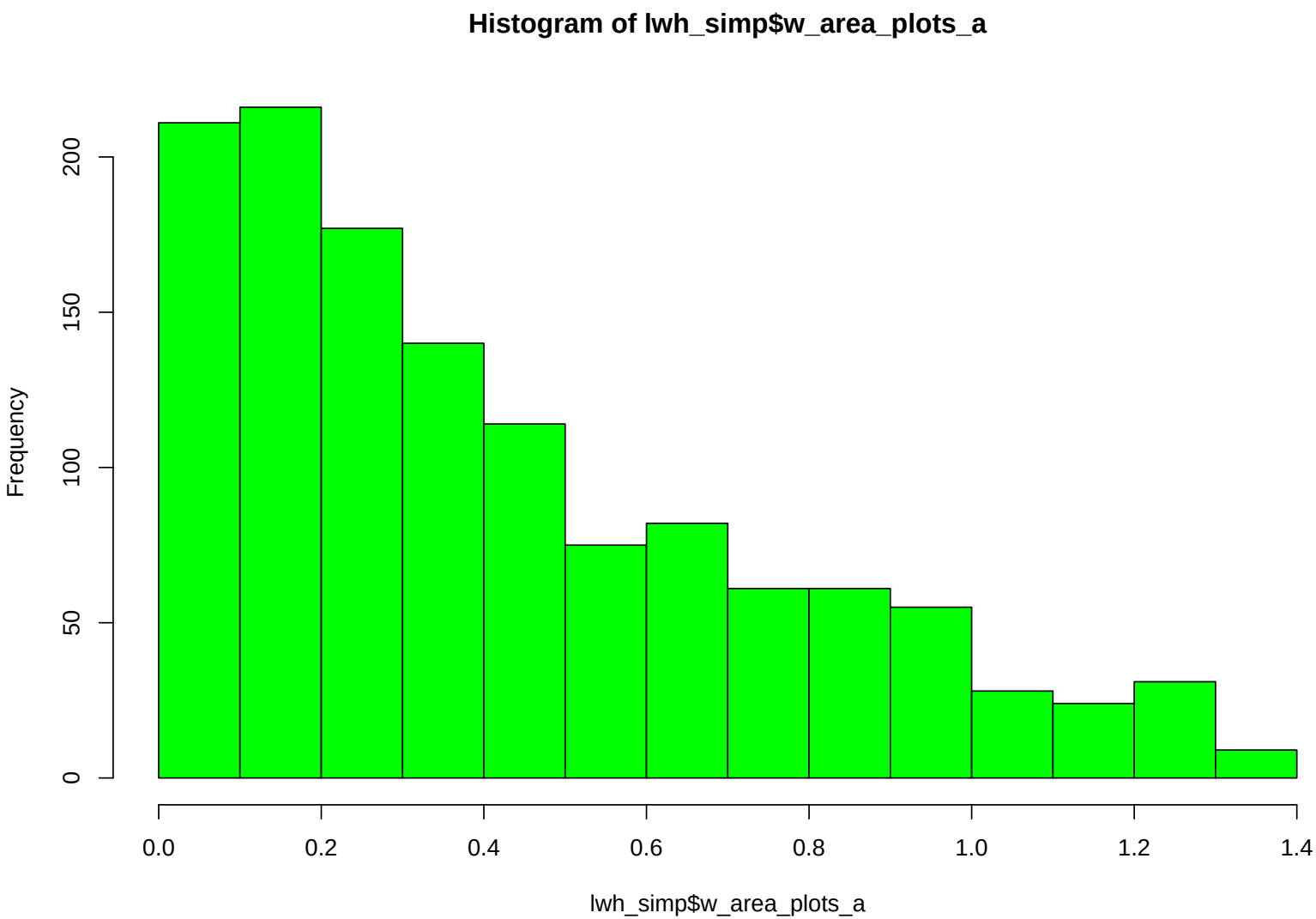
Exploratory graphs - Base plot

Exercise 2

1. Create a histogram of variable `w_area_plots_a` in data frame `lwh_simp`. Use the `hist()` function for that.
2. Try to set the color as `"green"`. Use the help if you are not sure how to do it.

Exploratory graphs - Base plot

```
hist(lwh_simp$w_area_plots_a, col = "green")
```



Outline

ggplot - Introduction

The ggplot2 package is an implementation of the theoretical framework for the construction of quantitative graphs developed in The Grammar of Graphics, a book by Leland Wilkinson.

It is a powerful and easy to use tool (once you understand its logic) that produces complex and multifaceted plots.

ggplot - Introduction

There are a few reasons why we use ggplot:

- It is generally easier to customize and tailor plots to your needs.
- It works great with almost any kind of plot, including maps, with little variation in syntax. The notable exception is 3D plotting, which can be done, but it's not as straightforward.
- It looks good.

ggplot - Grammar

Here is a list of ggplot2 most commonly used "grammar" elements:

Element	Description
Data	The dataset(s) being used.
Aesthetics	How your data is maped. For example what goes in the X and Y axis, size, shape and color.
Geometries	The visual elements used to represent the data (e.g. lines, dots, bars, etc.)
Themes	All non-data formating.

ggplot - Data structure

As almost everything in R, ggplot is very flexible and can be used in different ways. It is easier to work, however, if your data is in a specific format.

- Data has to be a data frame.
- Long format, especially categorical values.
  - It's better to have one variable with all the categories than a set of dummies.
- Labelled factors.

This is very similar to Stata, with the advantage of not needing to preserve and restore anything.

ggplot - Data structure

Here's an example of how an aggregated (collapsed) data set should look like:

```
##    year gender income_total_trim
## 1  2012 Female          34331.47
## 2  2013 Female          57123.69
## 3  2014 Female          48560.15
## 4  2016 Female          53980.17
## 5  2018 Female          70238.30
## 6  2012   Male          40222.72
## 7  2013   Male          77518.86
## 8  2014   Male          78886.26
## 9  2016   Male          85901.84
## 10 2018   Male         100867.57
```

ggplot - Introduction

Exercise 4

First, let's create a very simple plot with the total income on the Y axis and The plot area on the X axis.

1. Use the ggplot function to print your plot
   - Since it's the first one, Here's the code:

```
ggplot(data = lwh,
       aes(y = w_area_plots_a,
           x = income_total_trim))
```

2. Now, open yout Plots window. What does it show?

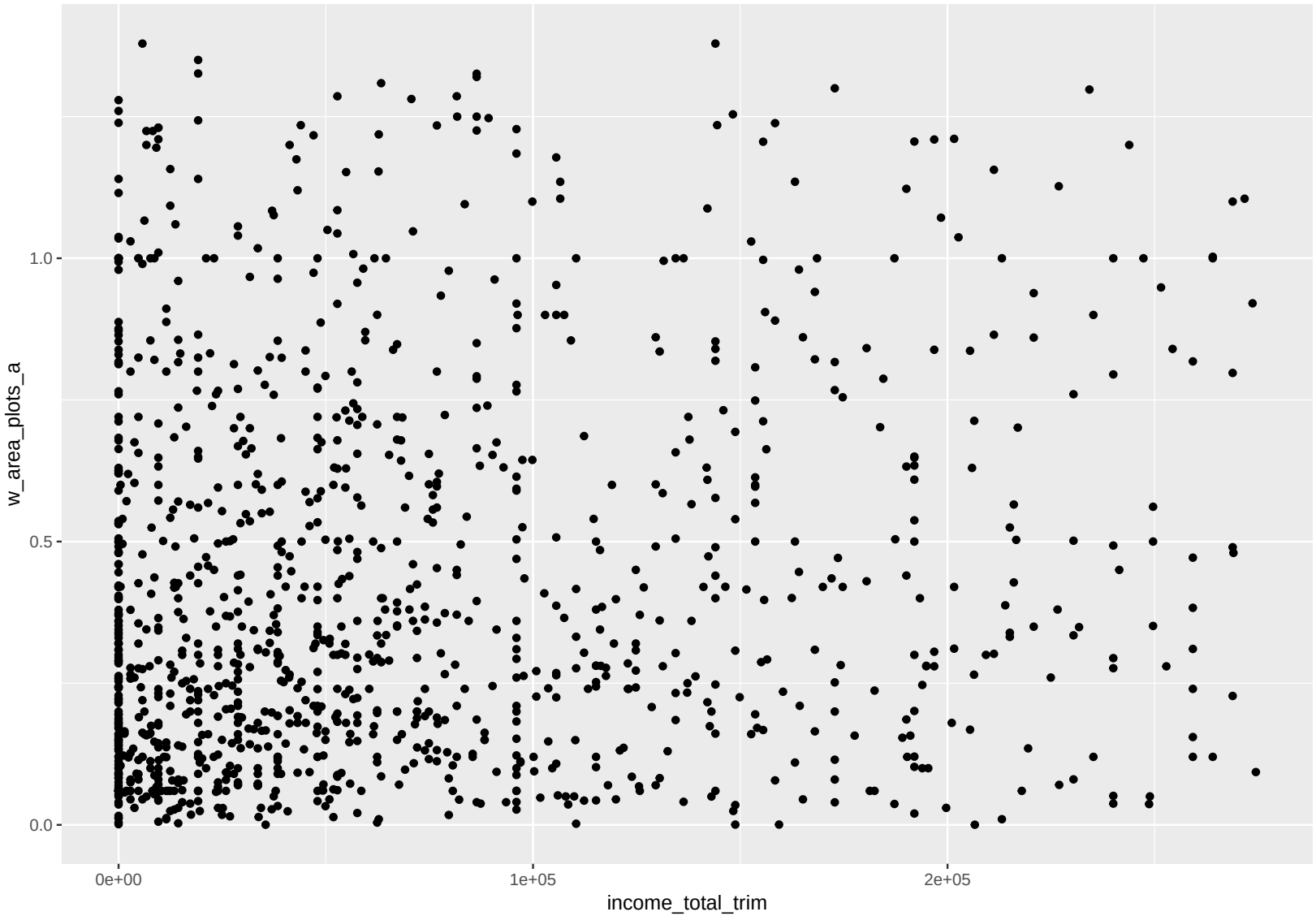
ggplot - Introduction

We mapped our variables, but without a geometry ggplot doesn't know how to represent the data.

Let's add a geometry

```
ggplot(data = lwh,
       aes(y = w_area_plots_a,
           x = income_total_trim)) +
  geom_point()
```

ggplot - Introduction



Outline

ggplot - Aesthetics

In ggplot , the aesthetic element defines how data is represented in the aesthetics (e.g. color, shape and size) of geometric objects (e.g. points, lines and bars).

Exercise 7: Part 1

1. Create a data frame containing only 2018 observations of `lwh` in the Rwamangana 35 site.

   - TIP: This time we want to keep only observations (lines) that meet a certain condition. Use the left argument of the brackets operators. Like this:

```
lwh[CONDITION,]
```

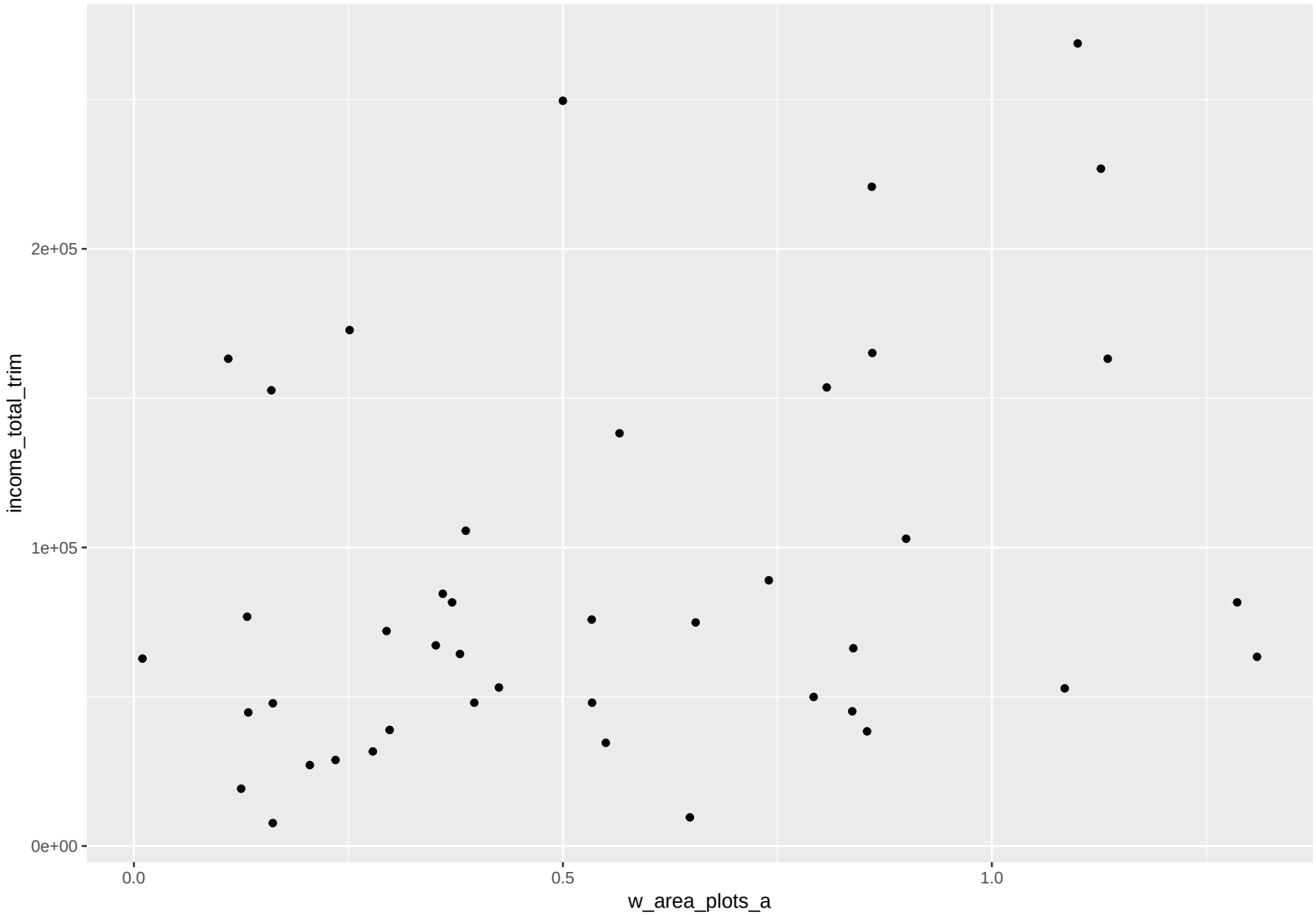
2. Use `ggplot()` to scatter plot area in season A and total annual income

ggplot - Aesthetics

```r
# Subset lwh to 2018 and Rwamangana 35
lwh_s <- lwh[lwh$year == 2018 & lwh$site_code == "Rwamangana 35", ]

# Plot
ggplot(lwh_s, aes(x = w_area_plots_a,
                  y = income_total_trim)) +
    geom_point()
```

ggplot - Aesthetics



ggplot - Aesthetics

Now add the women's dietary diversity score[1] to `aes()`.

Exercise 7: Part 2

1. Use the `wdds_score` varaible as color in `aes()`.
   - TIP: this is is a categorical variable, but it is stored in numeric format. Use the `factor()` function. Like this:

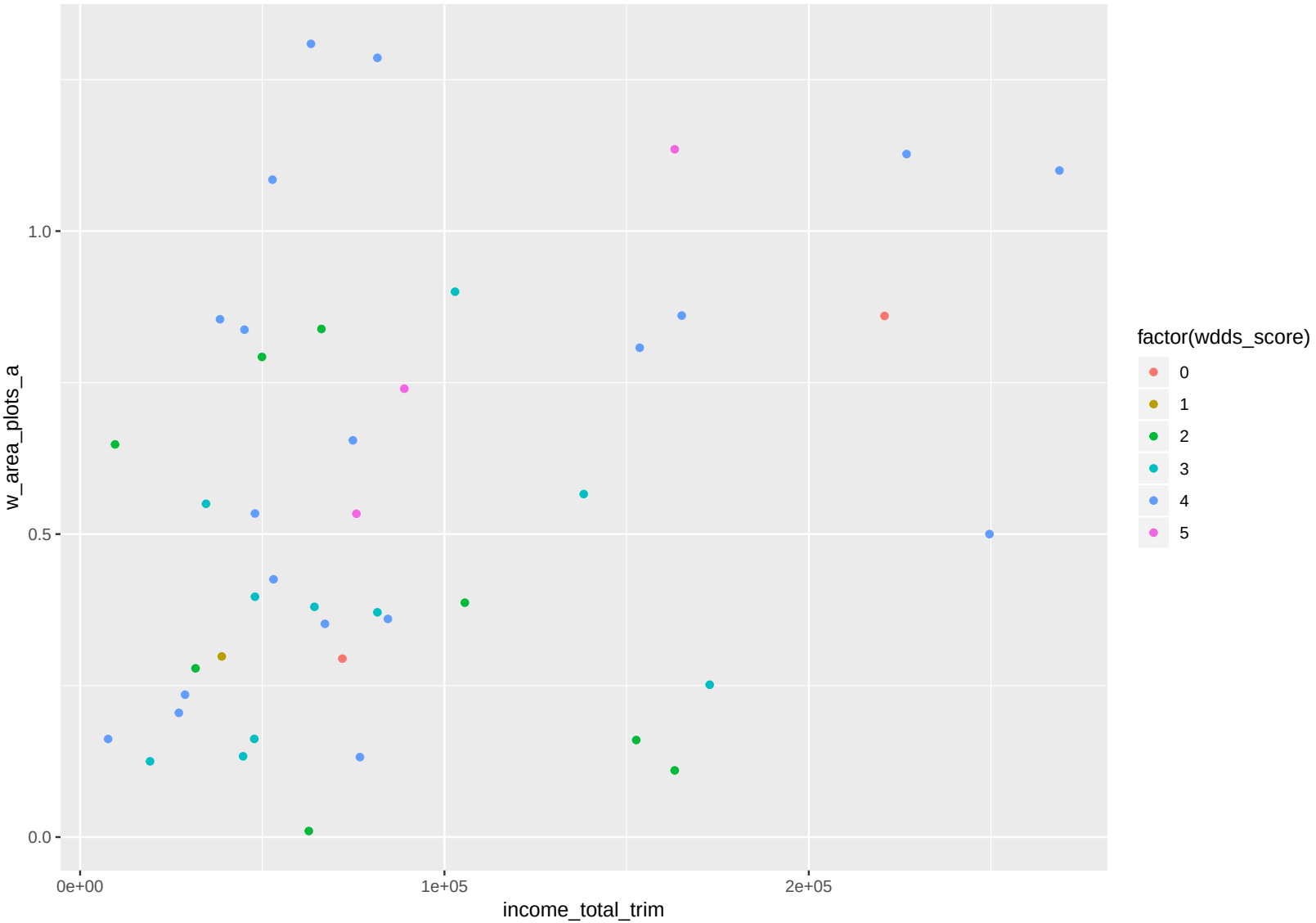
```
color = factor(wdds_score)
```

[1] Women's dietary diversity score is based on the questions asked specifically to an adult female respondent about her food consumption. We ask if she ate 16 food items categorized into 9 groups. Each group get 1 if any of the food belonging to the group is consumed. The final score is a simply sum of the categories.

ggplot - Aesthetics

```
# Plot code
ggplot(data = lwh_s,
       aes(y = w_area_plots_a,
           x = income_total_trim,
           color = factor(wdds_score))) +
    geom_point()
```

ggplot - Aesthetics



ggplot - Aesthetics

Now we will combine different arguments of `aes()`

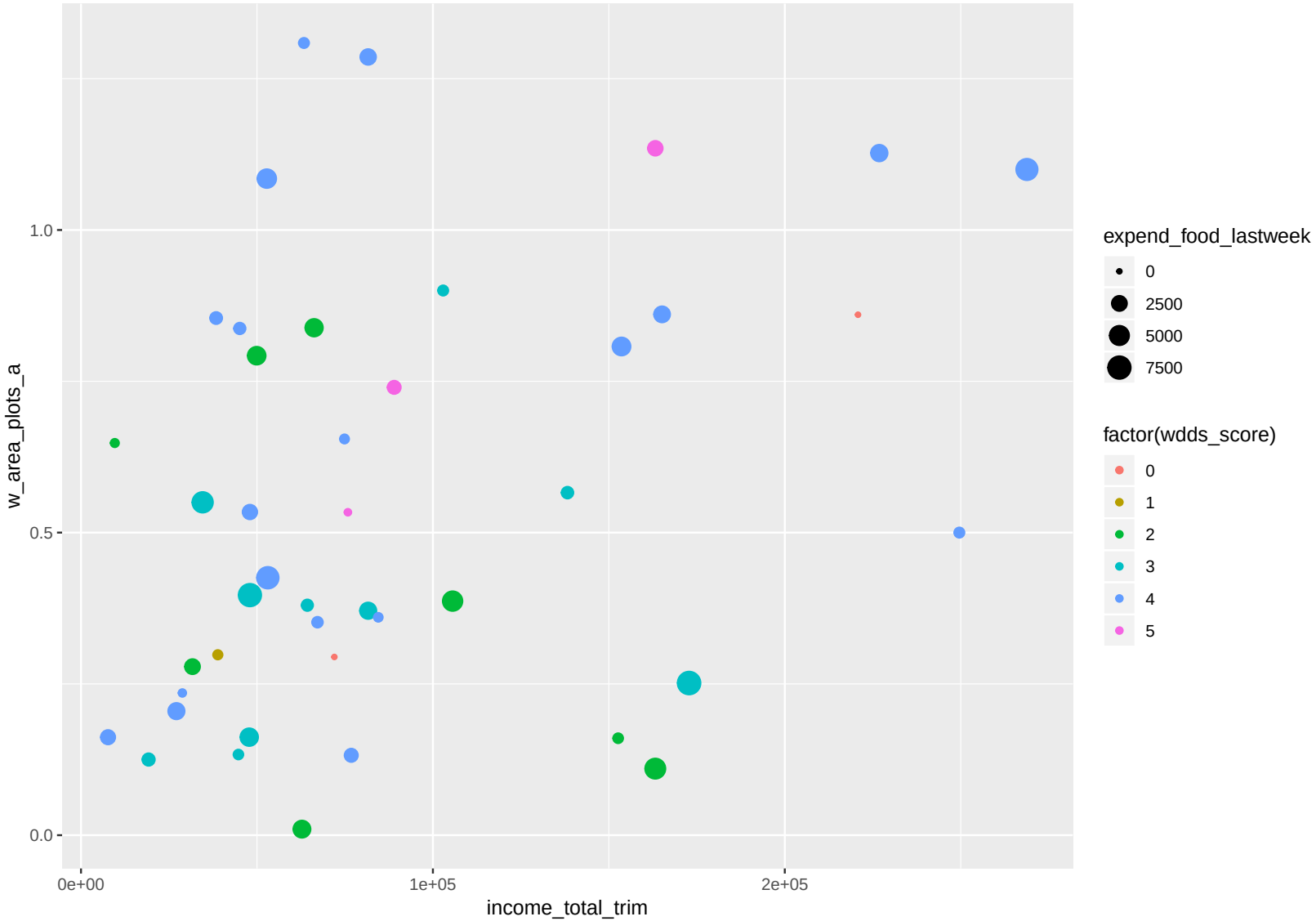
Exercise 7: Part 3

1. Add expenditure on food last week variable, `expend_food_lastweek` to the `size` argument of `aes()` in your last graph.

ggplot - Aesthetics

```
# Plot code
ggplot(data = lwh_s,
       aes(y = w_area_plots_a,
           x = income_total_trim,
           color = factor(wdds_score),
           size = expend_food_lastweek)) +
    geom_point()
```

ggplot - Aesthetics

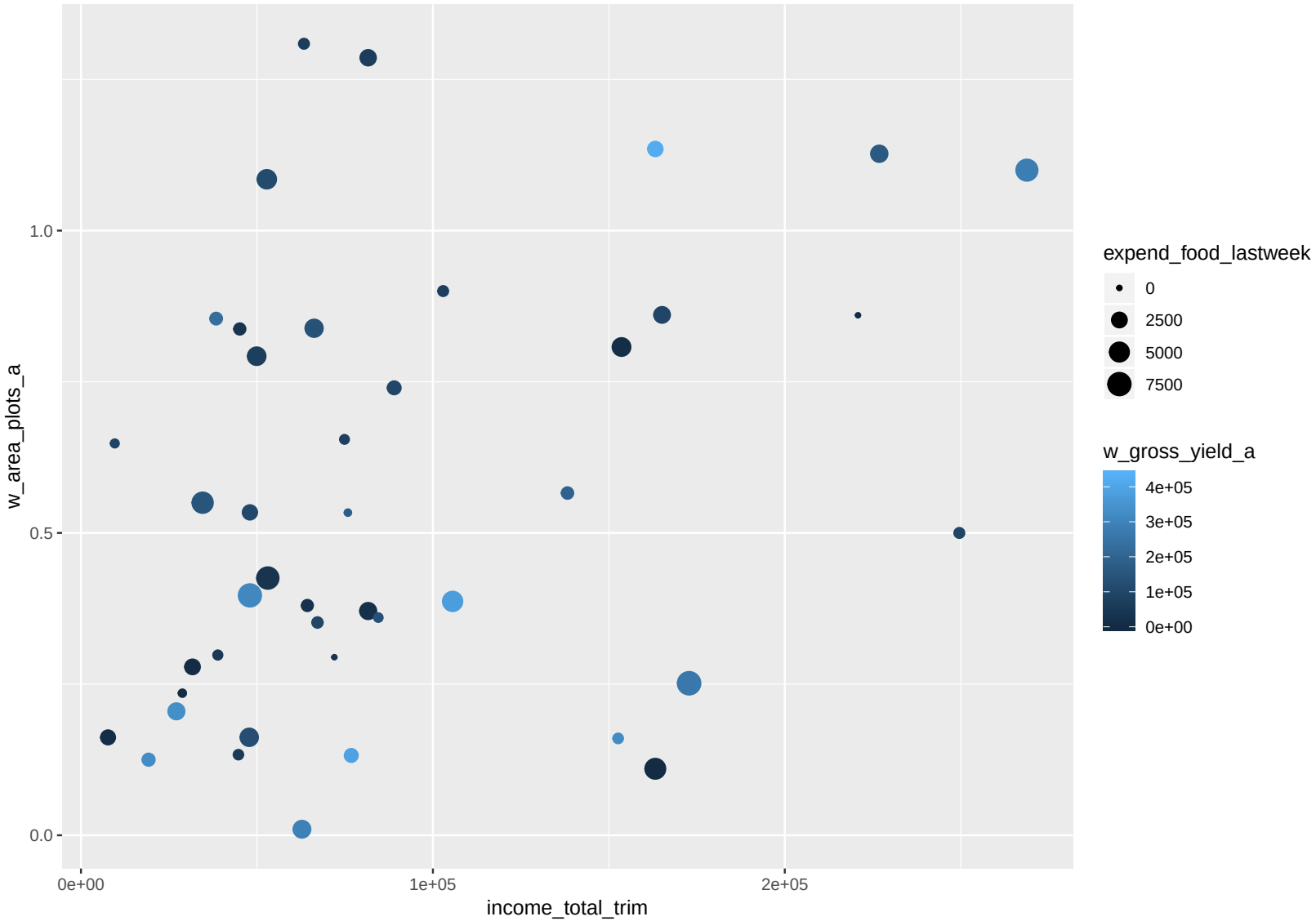


ggplot - Aesthetics

- Because we used a factor variable as the color in the aesthetic, it displayed different categories clearly
- If we had used a numeric variable instead, it would have created a gradient
- You'll not create this for time's sake, but here's how it would look:

```
# Plot code
ggplot(data = lwh_s,
       aes(y = w_area_plots_a,
           x = income_total_trim,
           color = w_gross_yield_a,
           size = expend_food_lastweek)) +
    geom_point()
```

ggplot - Aesthetics



Outline

ggplot - Titles and labels

Lastly, we can add titles, axis labels and format legends. This is done by adding additional layers to the plot.

To add a title and legend to the X and Y axis, we use the functions listed below. They all take a string scalar as argument.

- `ggtitle()` - adds a title to the plot
- `xlab()` - adds a label to X axis
- `ylab()` - adds a label to Y axis

ggplot - Titles and labels

To add legend titles, we will use two functions that control the formatting of the aesthetic. As they can take many different arguments, we use `name` argument sets legend title:

- `scale_color_discrete()` - formatting of the color aesthetic.
- `scale_size_continuous()` - formatting of the color aesthetic.

ggplot - Titles and labels

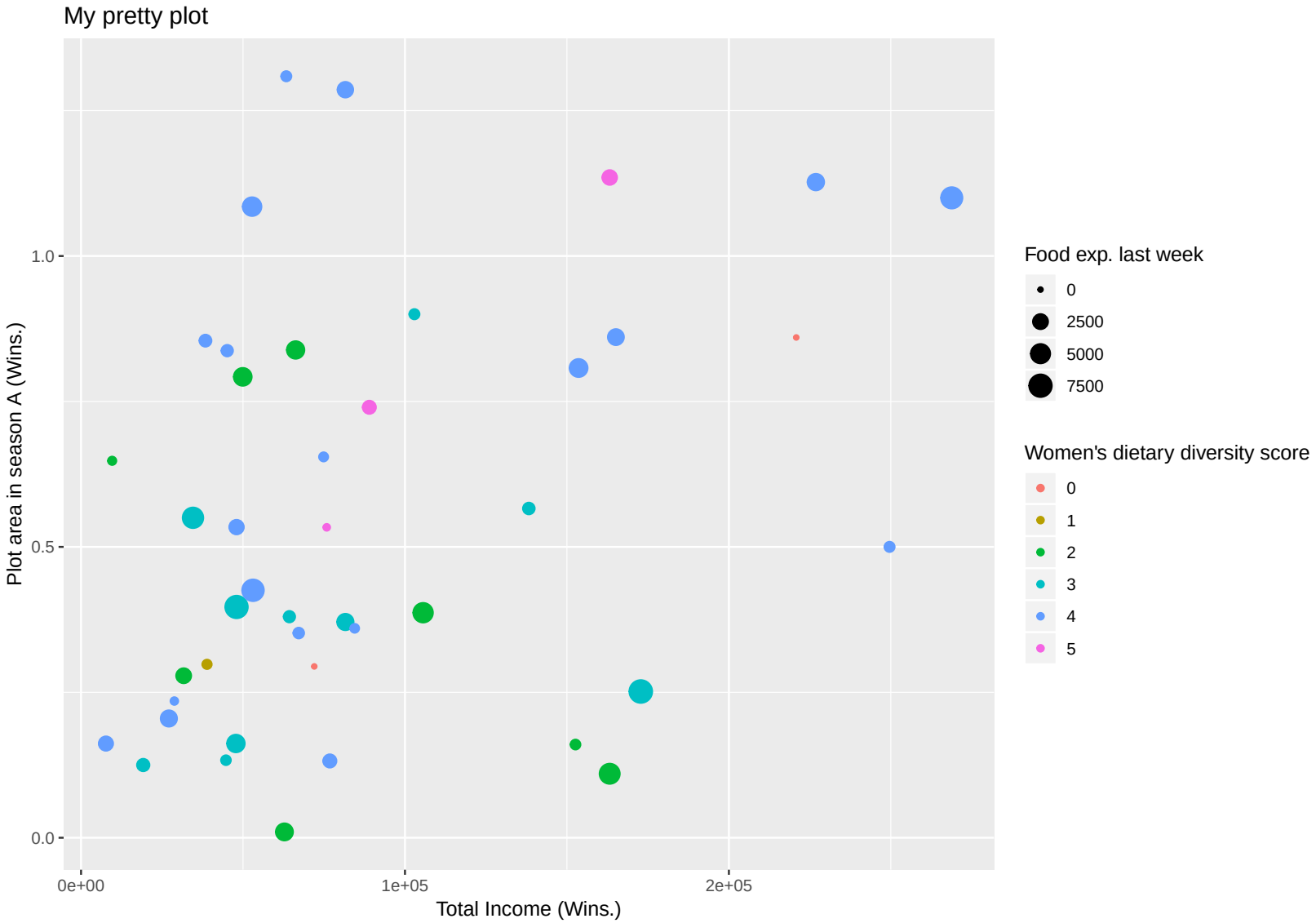
Exercise 8

1. Copy the code for the graph produced in the previous exercise.
2. Use the + symbol to add the layer.
3. `ggtitle()`, `xlab()` and `ylab()` take simple strings as inputs.
4. For `scale_color_continuous()` and `scale_size_continuous()` you need to specify `name` argument as the desired string.

ggplot - Titles and labels

```
# A properly labelled plot
ggplot(lwh_s, aes(y = w_area_plots_a,
                  x = income_total_trim,
                  size  = expend_food_lastweek,
                  color = factor(wdds_score))) +
  geom_point() +
  ggtitle("My pretty plot") +
  xlab("Total Income (Wins.)") +
  ylab("Plot area in season A (Wins.)") +
  scale_color_discrete(name = "Women's dietary diversity score") +
  scale_size_continuous(name = "Food exp. last week")
```

ggplot - Titles and labels



Outline

Saving a plot

Let's save the previous plot using the PDF graphics device

Exercise 9:

1. Open the PDF graphics device by using the `pdf()` function.
2. Set the `file` argument as your outputs path with the file name (including ".pdf")
3. Paste the code of the previous plot bellow the `pdf()` function.
4. Finally, use the `dev.off()` to close the device, i.e. tell R that you're finished plotting.

Saving a plot

```
# Open PDF graphics device
pdf(file = file.path(rawOutput, "plot1.pdf"))

 # Plot
 ggplot(data = lwh_s,
     aes(y = w_area_plots_a,
         x = income_total_trim,
         color = factor(wdds_score),
         size = expend_food_lastweek)) +
   geom_point() +
   ggtitle("My pretty plot") +
   xlab("Total Income (Wins.)") +
   ylab("Plot area in season A (Wins.)") +
   scale_color_discrete(name = "Women's dietary diversity score") +
   scale_size_continuous(name = "Food exp. last week")

# Close PDF graphics device
dev.off()
```

Outline

Interactive graph

Now we'll use ggplotly() to create an interactive graph!

Bonus exercise - ggplotly

1. Load the `plotly` package
2. Store one the plots produced in the previous `ggplot` exercises in an object.
3. Pass that object as argument for the `ggplotly()` function

Interactive graph

```
# Load package
library(plotly)

# Store graph in plot1
plot1 <- ggplot(data = lwh_s,
                aes(y = w_area_plots_a,
                    x = income_total_trim,
                    color = factor(wdds_score))) +
    geom_point()

# Use ggplotly to create an interactive plot
ggplotly(plot1)
```

Outline

References and recommendations

Websites:

- The R Graph Gallery: https://www.r-graph-gallery.com/
- Gpplot official site: http://ggplot2.tidyverse.org/

Online courses:

- Johns Hopkins Exploratory Data Analysis at Coursera: https://www.coursera.org/learn/exploratory-data-analysis

Books:

- The grammar of graphics by Leland Wilkinson.
- Beautiful Evidence by Edward Tufte.
- R Graphics cook book by Winston Chang
- R for Data Science by Hadley Wickham andGarrett Grolemund

Outline

Appendix

Slides that didn't make the cut

R common graphic devices

R has several built-in graphic devices. A graphic device is the medium where you visually represent a plot. Here are the most come:

- The standard screen device or the plot pane in RStudio.
- PDF file (file device).
- PNG (file device).
- JPEG (file device).

Differed ways of saving a plot

There are a couple of ways of saving plots in R:

- With ggplot you can use the `ggsave()` function that has a simple syntax and can do most of the dirty work for you.
- Choosing an specific graphics device (other than the default window). pdf(), png(), jpeg() and bmp() are functions that call graphic devices that save your plots in these formats. There are a few others built-in and you can check them by typing *?Devices*, and even more in CRAN.

Graphic systems

R has four main graphic systems:

- Base plot - most basic plotting system.
- Grid graphics - is a low-level graphics system mainly used by package developers.
- Lattice graphics - easy way to plot high dimensional data or many graphs at the same time.
- Ggplot2 - a flexible and powerful tool to plot beautiful graphs with intuitive syntax