Introduction to R (Part 2) - Data Visualization

Dr. Lendie Follett

2022-07-07

Working with packages

For the next exercises we need to install a *package*. A *package* is a set of functions created by the wider R community that you can add on to make base R even more powerful. We will start our journey with the `ggplot2` package.

Install the package:

```
#You only have to do this once
install.packages("ggplot2")
```

Load the package:

```
#You have to do this every time you re-open R
#and want to use the package
library(ggplot2)
```

Data frames

- The most common way we work with data in R is through data frames
- In this class, we will get familiar with data frames through graphics
- R has some pre-loaded data frames that we can get started with
- In the **console**, type `?diamonds` to learn about this particular data set. Read the description of each of the variables

Initial Exploration of Data Frames

For a data frame named d, we can (and should!) use:

- d: print data
- `head(d, n = 10)`: print first 10 rows
- `tail(d, n = 5)`: print last 5 rows
- `summary(d)`: print numerical summary of each column in d
- `str(d)`: print the structure of the data
- `dim(d)`: print the dimensions (number of rows, columns)

These are essential to get to know your data.

Extracting bits from a data frame

Indexing

- In R, it is easy to pick and choose rows and columns to view (or manipulate...)
- First note that a data frame has 2 *dimensions*: (1) rows and (2) columns
- We can use what is called *positional indexing* to extract the elements we want from both dimensions

The syntax of positional indexing looks like:

```
d[row_position, column_position]
#or
d[row_position, column_name]
```

Extracting bits from a data frame

Indexing

- In R, it is easy to pick and choose rows and columns to view (or manipulate...)
- First note that a data frame has 2 *dimensions*: (1) rows and (2) columns
- We can use what is called *positional indexing* to extract the elements we want from both dimensions

The syntax of positional indexing looks like:

```
d[2, ] #extract second row
```

column_name1	column_name2	column_name3
X	X	X
X	X	X
X	X	X
X	X	X
X	X	X

Extracting bits from a data frame

Indexing

- In R, it is easy to pick and choose rows and columns to view (or manipulate...)
- First note that a data frame has 2 *dimensions*: (1) rows and (2) columns
- We can use what is called *positional indexing* to extract the elements we want from both dimensions

The syntax of positional indexing looks like:

```
d[,3 ] #extract third column
```

column_name1	column_name2	column_name3
X	X	X
X	X	X
X	X	X
X	X	X
X	X	X

Extracting bits from a data frame

Indexing

- In R, it is easy to pick and choose rows and columns to view (or manipulate...)
- First note that a data frame has 2 *dimensions*: (1) rows and (2) columns
- We can use what is called *positional indexing* to extract the elements we want from both dimensions

The syntax of positional indexing looks like:

```
d[c(2:3),] #extract multiple rows
```

column_name1	column_name2	column_name3
X	X	X
X	X	X
X	X	X
X	X	X
X	X	X

Extracting bits from a data frame

Indexing

- In R, it is easy to pick and choose rows and columns to view (or manipulate...)
- First note that a data frame has 2 *dimensions*: (1) rows and (2) columns
- We can use what is called *positional indexing* to extract the elements we want from both dimensions

Say we have data frame d. The syntax of positional indexing looks like:

```
d[2:3,"column_name1" ] #extract the second and third rows of "column_name1"
```

column_name1	column_name2	column_name3
X	X	X
X	X	X
X	X	X
X	X	X
X	X	X

Extracting bits from a data frame

- An alternative (for columns, anyway) to positional indexing is the $ syntax
- d$variable_name basically means I want to extract variable_name which can be found in object d.

```
d$variable_name
d$variable_name[row_position]
```

```
d$column_name1 #extract the column named "column_name1"
```

column_name1	column_name2	column_name3
X	X	X
X	X	X
X	X	X
X	X	X
X	X	X

Data Visualization - common univariate plot types (refresher)

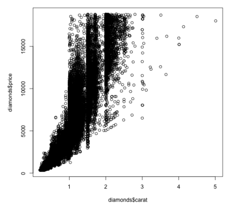
- bar chart: a chart displaying groups of data with bars having a length proportional to the value corresponding to each group.
- histogram: a type of bar chart used to show the frequency, or relative frequency, with which data points take on values. It shows the shape of the distribution of continuous data points.
- box plot (a.k.a box and whisker plot): a graphical representation of the quantiles and outliers of a continuous variable.
- violin plot: like a box plot, but more detailed (later)

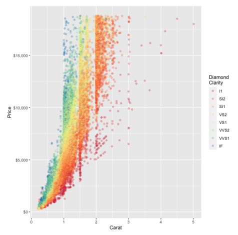
Data Visualization - common multivariate plot types

- multivariate plot: displaying two or more columns/variables at once, usually to illustrate the relationship between them
- Many univariate plot types can be extended to represent multivariate distributions using additional aesthetics such as *color* and *facets*.
- The most common way to display a two dimensional relationship is with a *scatter plot*
- scatter plot: a plot displaying the joint distribution of two continuous variables by mapping the value of one variable to the x-axis and the value of the other variable to the y-axis.

Why do we use ggplot2? Why not base-R?

- Industry standards and expectations
- Flexibility
- Also,





I mean, seriously.

Building a ggplot

- To create a plot, we start with the function `ggplot()` - this creates a coordinate system that can be built upon.
- The function `ggplot()` takes in the data set of interest as the first argument. For example,

```
ggplot(data = diamonds)
```

```
#not very interesting
```

Building a ggplot

- A plot is created by adding:
  1. **one or more layers to ggplot**. Examples of layers:
     - `geom_point()` adds a layer of points to plot (scatter plot)
     - `geom_bar()` adds a layer of bars (bar chart)
     - `geom_boxplot()` adds a layer of box-and-whisker plots (boxplots)
     - MANY more....
  2. **scale (optional)**: fine tuning how to map data to aesthetics. For example,
     - `scale_colour_brewer()` has great colorblind-friendly palettes for categorical variables
     - `scale_colour_distiller()`
     - many more
  3. **coordinate system (optional)**: normally Cartesian, but polar and flipped, and others are also options.
  4. **facet (optional)**: useful for incorporating additional categorical variables.
  5. **theme (optional)**: fonts, font sizes, text angles, colors, etc.

A simple template for a ggplot

- Text between $<$ and $>$ are to be replaced with a data set name, a geom function, and a collection of mappings, in the order of appearance

```
ggplot(data = <data>) +
  <geom_function>(mapping = aes(<mappings>))
```

- *Important*: The mapping of aesthetics (x location, y location, color, etc...) to variables in the data set occurs within `aes()`

Let's get started.

What clarities of diamonds are the most common? The least?

[*Reword*: What is the distribution of the variable `clarity`?]

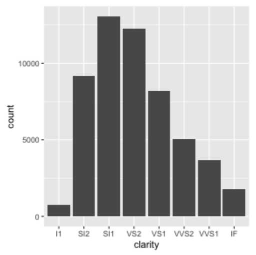
This is a question requiring a *univariate* plot of a *categorical* variable.

We can use a bar chart, and variations of it, to display this.

Visualizing the distribution of clarity

We've already specified diamonds as our data set name. Add the desired geom function and a collection of mappings within aes().

```
ggplot(data = diamonds) +
  geom_bar(aes(x = clarity))
```

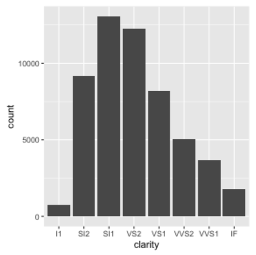


- Specifically, we're saying we want a barchart to count the number of times each clarity level shows up in the dataset.

Visualizing the distribution of clarity

We've already specified diamonds as our data set name. Add the desired geom function and a collection of mappings within aes()".

```
ggplot(data = diamonds) +
  geom_bar(aes(x = clarity))
```

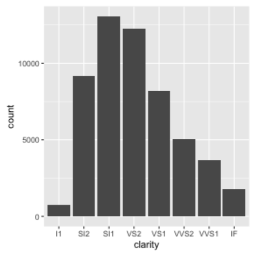


- Specifically, we're saying we want the categorical variable `clarity` to be on the x axis.

Visualizing the distribution of clarity

We've already specified diamonds as our data set name. Add the desired geom_function and a collection of mappings within aes().

```
ggplot(data = diamonds) +
  geom_bar(aes(x = clarity))
```

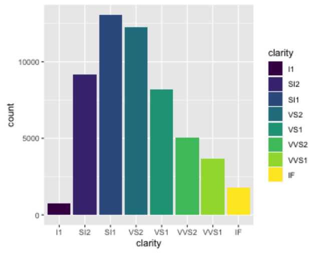


- Note that any mappings having to do with data MUST occur within aes()

Visualizing the distribution of clarity

Further, we could also add some color using the fill aesthetic.

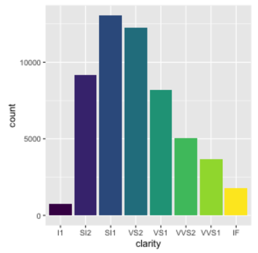
```
ggplot(data = diamonds) +
  geom_bar(aes(x = clarity, fill = clarity))
```



Visualizing the distribution of clarity

Note the legend on the right hand side. We can add a theme to remove it *if it contains unnecessary/redundant information.*

```
ggplot(data = diamonds) +
  geom_bar(aes(x = clarity, fill = clarity)) +
  theme(legend.position = "none")
```



Your Turn

1. What if we map a different categorical variable to the fill aesthetic? Fill by the `cut` variable.
2. What does this plot tell us?
3. Is the legend helpful now?

Next question, please

Before we get into what affects the price of diamonds, let's examine a numeric column. What is the distribution of the carat of diamonds?

Again, this will be a univariate plot.

Carat is continuous, so we won't do a bar plot. We could try a histogram!

Visualizing carat

Start out as you normally would:

```
ggplot(data = diamonds)
```

Visualizing carat

Add the appropriate geom function.

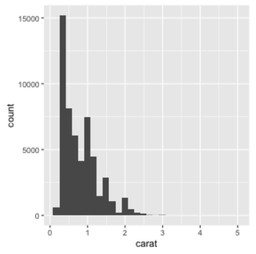
```
ggplot(data = diamonds) +
  geom_histogram()
```

Visualizing carat

Fill in only the necessary aesthetics to start. As with `geom_bar`, the only necessary aesthetic is `x=`.

```
ggplot(data = diamonds) +
  geom_histogram(aes(x = carat))
```

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



- What's that message talking about?

Visualizing carat

The height of the bars in a histogram represent the number of rows of data that fall into the respective *bins*. The *binwidth* can affect what we learn from a plot.

```
ggplot(data = diamonds) +
  geom_histogram(aes(x = carat)) #default behavior
ggplot(data = diamonds) +
  geom_histogram(aes(x = carat), binwidth = .5) #(0,0.5], (0.5, 1], etc...
ggplot(data = diamonds) +
  geom_histogram(aes(x = carat), binwidth = .05)#(0,0.05], (0.05, 0.1], etc...
ggplot(data = diamonds) +
  geom_histogram(aes(x = carat), binwidth = .01)

#binwidth is an argument of the function geom_histogram()
#it doesn't involve our data so it goes OUTSIDE of aes()
```

The big question:

What affects the price of a diamond? We have carat, color, clarity, and cut.

Let's start by examining the relationship between carat (numeric) and price (also numeric).

This suggests a scatterplot.

Visualizing what drives price

Start out as you normally would:

```
ggplot(data = diamonds)
```

We want a scatter plot, which can be obtained using `geom_point()`

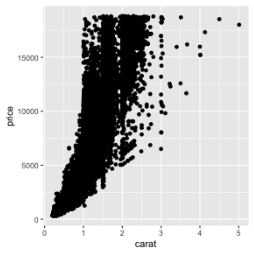
```
ggplot(data = diamonds) +
geom_point(aes())
```

Here, we have 2 mappings to take care of since a scatterplot requires something on the x-axis and on the y-axis!

```
ggplot(data = diamonds) +
geom_point(aes(x = carat, y = price))
```

Visualizing what drives price

```
ggplot(data = diamonds) +
geom_point(aes(x = carat, y = price))
```

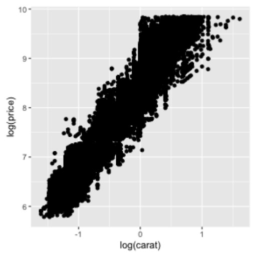


Interpreting Scatterplots

- Form:
- Direction:
- Strength:
- Deviations from this pattern?:

What changes if we take the log?

```
ggplot(data = diamonds) +
geom_point(aes(x = log(carat), y = log(price)
```



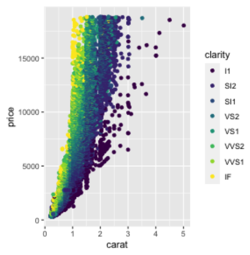
Interpreting Scatterplots

- Form:
- Direction:
- Strength:
- Deviations from this pattern?:

Visualizing what drives price

- We've assessed the relationship between carat and price. We can use other mappings to incorporate other variables.
    - color
    - size
    - shape

```
ggplot(data = diamonds) +
geom_point(aes(x = carat, y = price, color = clarity))
```



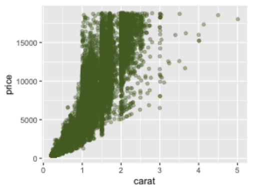
Mapping aesthetics to data vs setting aesthetics

Specifying color and other aesthetics is done within aes() when we map to data. However, what if we just wanted to make an olive-colored scatterplot?

```
ggplot(data = diamonds) +
geom_point(aes(x = carat, y = price), color = "darkolivegreen")
```

And what if we wanted to make the points more transparent?

```
ggplot(data = diamonds) +
geom_point(aes(x = carat, y = price), color = "darkolivegreen", alpha = I(.4))
```

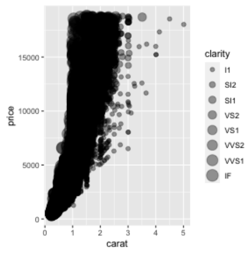


Mappings being set independent of your data set go outside of `aes()`.

Visualizing what drives price

- We've assessed the relationship between carat and price. We can use other mappings to incorporate other variables.
    - color
    - size
    - shape

```
ggplot(data = diamonds) +
geom_point(aes(x = carat, y = price, size = clarity), alpha = I(0.4))
```



Visualizing what drives price

- We've assessed the relationship between carat and price. We can use other mappings to incorporate other variables.
    - color
    - size
    - shape

```
ggplot(data = diamonds) +
geom_point(aes(x = carat, y = price, shape = clarity), alpha = I(0.4))
```

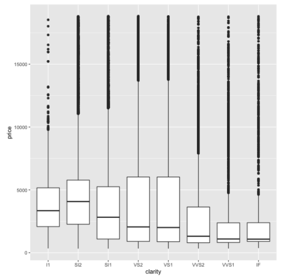
```
## Warning: Using shapes for an ordinal variable is not advised

## Warning: The shape palette can deal with a maximum of 6 discrete values because
## more than 6 becomes difficult to discriminate; you have 8. Consider
## specifying shapes manually if you must have them.

## Warning: Removed 5445 rows containing missing values (geom_point).
```

Often, the data will determining which of the mappings are best at displaying the relationships in the data. There are several signs that this particular iteration of the plot may not be ideal.

Visualizing what drives price

- What about the relationship between clarity (categorical) and price?
- In SAS, we did this kind of thing with box plots.

```
ggplot(data = diamonds) +
geom_boxplot(aes(x = clarity, y = price))
```



Visualizing what drives price

- What about the relationship between clarity (categorical) and price?
- In SAS, we did this kind of thing with box plots.

```
ggplot(data = diamonds) +
geom_boxplot(aes(x = clarity, y = price), colour = clarity)
```

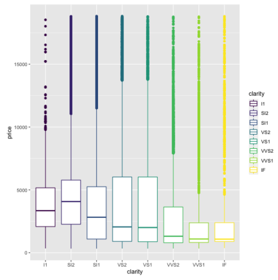
```
## Error in layer(data = data, mapping = mapping, stat = stat, geom = GeomBoxplot, : object 'clarity' not
```

What happened?

Visualizing what drives price

- What about the relationship between clarity (categorical) and price?
- In SAS, we did this kind of thing with box plots.

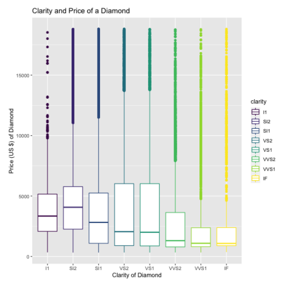
```
ggplot(data = diamonds) +
geom_boxplot(aes(x = clarity, y = price, colour = clarity))
```



Visualizing what drives price

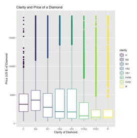
- What about the relationship between clarity (categorical) and price?
- In SAS, we did this kind of thing with box plots.

```
ggplot(data = diamonds) +
geom_boxplot(aes(x = clarity, y = price, colour = clarity)) +
  ggtitle("Clarity and Price of a Diamond") +
  labs(x = "Clarity of Diamond", y = "Price (US $) of Diamond")
```

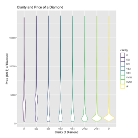


Alternative to boxplots

```
ggplot(data = diamonds) +
 geom_boxplot(aes(x = clarity, y = price, col
  ggtitle("Clarity and Price of a Diamond") +
  labs(x = "Clarity of Diamond", y = "Price (
```



```
ggplot(data = diamonds) +
  geom_violin(aes(x = clarity, y = price, col
  ggtitle("Clarity and Price of a Diamond") +
  labs(x = "Clarity of Diamond", y = "Price (
```



Some useful and fun soures

We're not done with visualization. Not even close. But in the meantime, here are some sources you might enjoy:

- Print this: https://raw.githubusercontent.com/rstudio/cheatsheets/main/data-visualization.pdf
  - You're welcome.
- For reference: https://ggplot2.tidyverse.org/reference/
- For inspiration: https://www.r-graph-gallery.com/

Your Turn

Open a new script. Start your script with `library(ggplot2)`. Save it as `t2_yt_mpg.R`

We're ultimately interested in what affects the highway gas mileage of a vehicle.

1. Type `?mpg` into your console and read about this data set. Also use `head()` and `summary()` to get to know the data better.
2. Create a plot displaying the distribution of the type of transmission of the vehicles.
   - Add color to this plot.
   - Add professional-looking x and y-axis labels.
   - Add a title.
3. Create a histogram displaying the distribution of engine displacement.
   - Polish this plot with labels and a title.
   - Did you look at different binwidths? Should you?
4. Create a scatter plot to examine the relationship between engine displacement and highway gas mileage.
   - Explain to your uncle (say) the relationship between engine displacement and highway gas mileage.
   - How could you incorporate the type of transmission into this plot? Try a few options, and choose which you feel most clearly displays the data patterns.
   - Set the alpha transparency mapping to 0.3. Why are some points darker/less transparent than others?
   - Add axis labels and a title.
5. Choose another categorical variable and create a quality visualization of the relationship between it and **city** gas mileage.