Learn R!

Complete Beginner's Workshop

R LADIES Vancouver

Sat April **14**

1 pm | Room 121 LSK Building
UBC Campus

ECOSCOPE
routing the microcosmos

LADIES
Vancouver

A local chapter of R-Ladies, a global organization promoting gender diversity in the R community.

@RLadiesVan

Events meetup/R-Ladies-Vancouver

vancouver@rladies.org

R-Ladies mission

More women/non-binary

- coders
- developers
- speakers
- leaders

More awesome people developing R packages and being part of the R community.

Hello!

Your chapter’s current organizers

Anna

Özüm

Marion

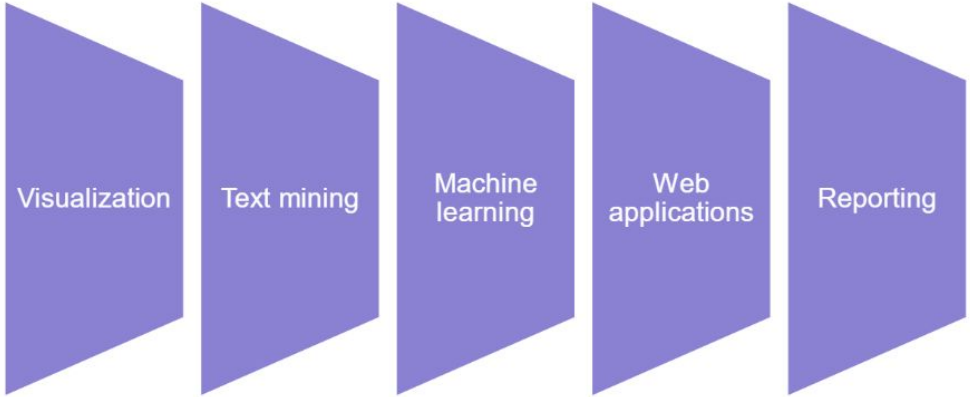
What is R and RStudio?

R has diverse application

It is one of the most popular languages in Data science

R is a free software environment for statistical computing, graphics, machine learning, etc.
It consists of various packages which are focused for different objectives and use cases.

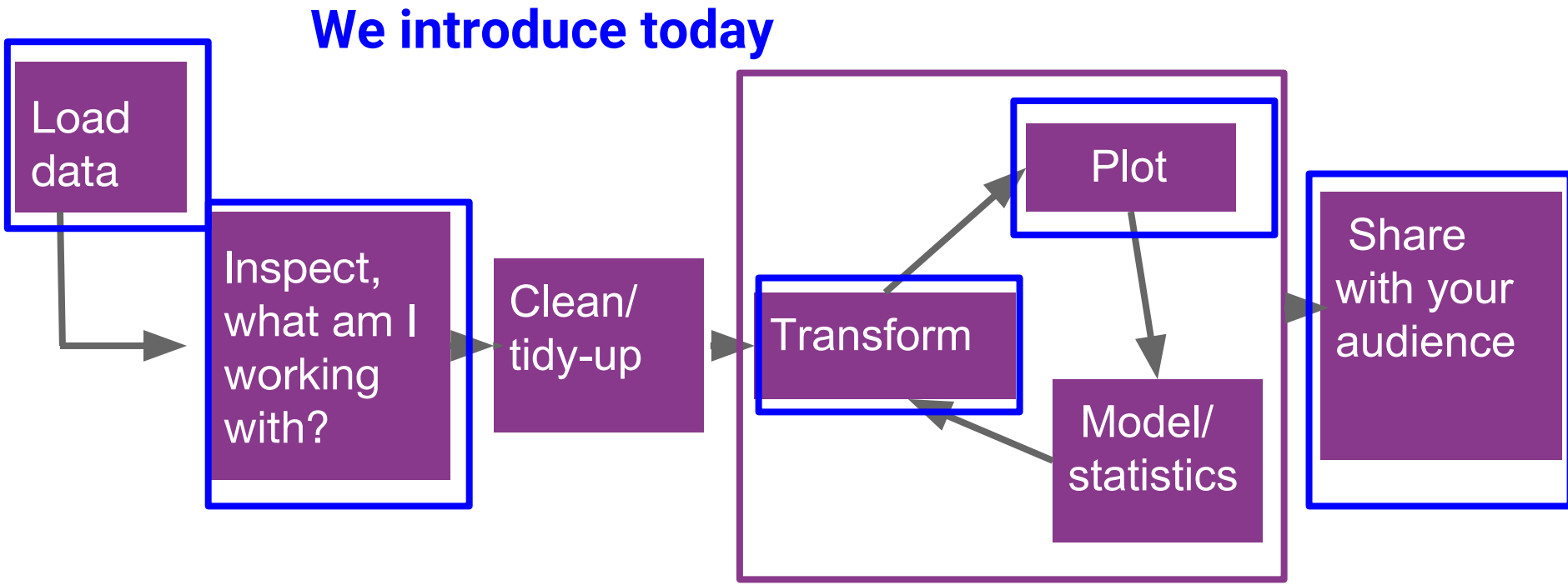
Some example of R usages



From R-Ladies Milan

Ok, now let’s learn some R!

Life stages of data analysis



Adapted from R for Data Science by Hadley Wickham
http://r4ds.had.co.nz/introduction.html

Today's workshop

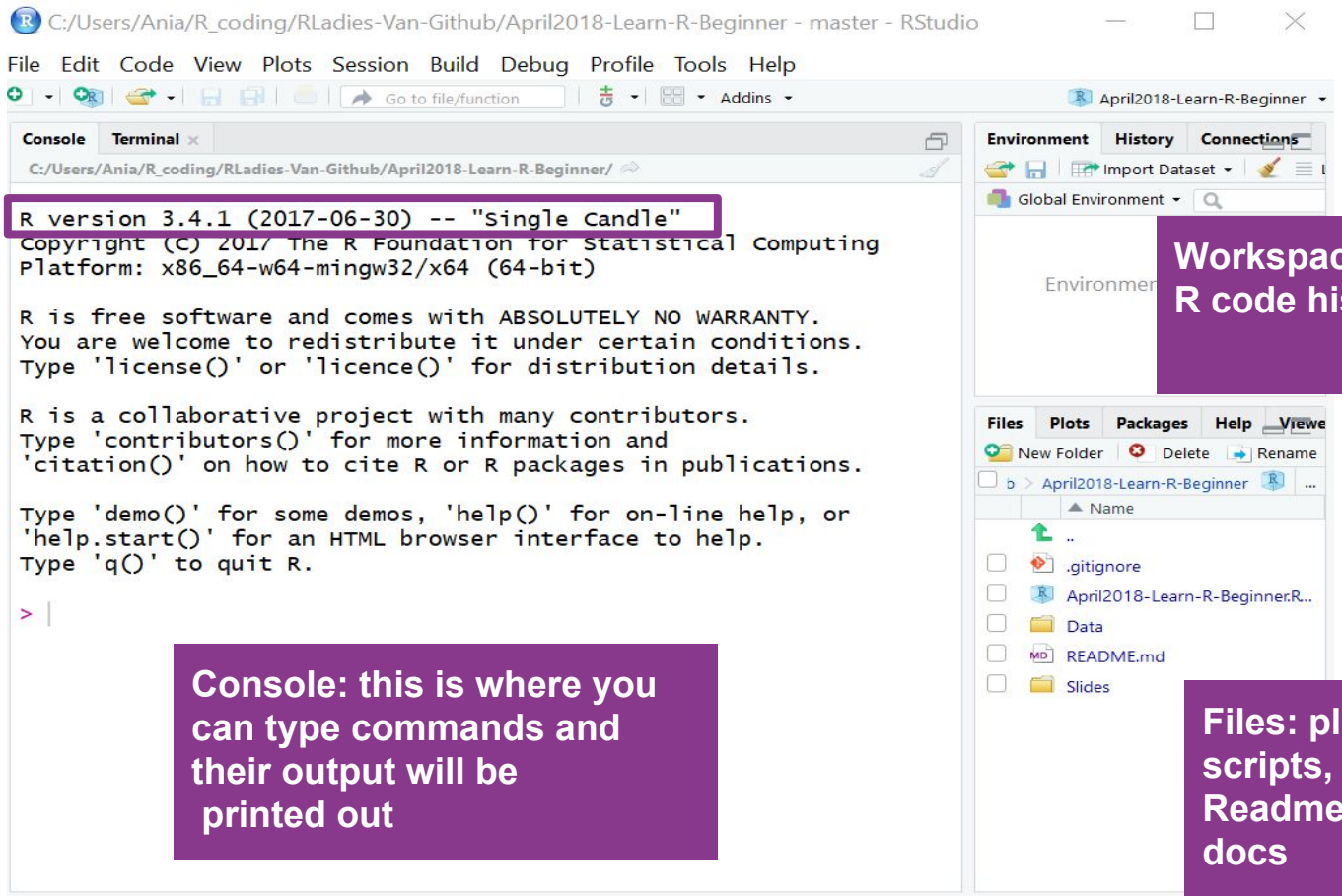
1. RStudio orientation
2. Basics of coding in R
3. Working with functions and packages
4. Loading and inspecting the Titanic dataset
5. Basics of DATA WRANGLING with the DPLYR package
6. Basics of DATA PLOTTING with the GGPLOT2 package

Workshop format

- Say hello to your neighbour!
- After each concept, we all do a challenge together

1. RStudio orientation

RStudio launched



Workspace
R code history

Console: this is where you can type commands and their output will be printed out

Files: plots, scripts, folders, Readme files, docs

Where is my R work going to exist?

What is a working directory?

- A `location` on your computer
- Below is a `path` to where R will put anything I generate with my R script.
- You can find out where that is for you by typing `getwd()` `in the console` or noting the path in the console bar

Best to keep everything related
to your analysis in the same **LOCATION**
How? with R Projects!

great tip

Steps

1. Decide where on your computer you want your future R work to live (somewhere sensible)

2. File> New project>New working>R project

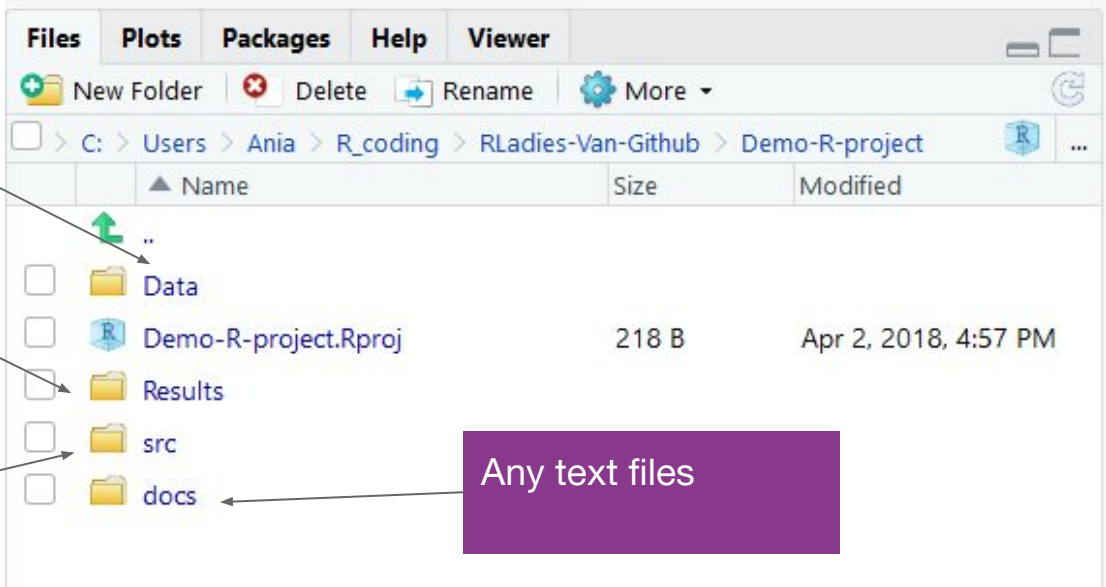
Best to keep everything in that **LOCATION** organized

How?

data files e.g. csv & metadata

Data you “cleaned” for stat. Analysis etc and final data

Source code -> put your scripts here



Any text files

More on this from
https://swcarpentry.github.io/r-novice-gapminder/02-project-intro/

This is a good place to start, you might want to adapt to suit your needs.

Challenge 1!

1. Create an **Rproject** called “R-beginner-workshop". Click File>New Project>New directory>Empty project
2. Create a folder called "**data**"
3. Put the **titanic.csv** into the folder data
4. Create a folder called "scr”

2.
R coding basics

Calculating with R

You can clear your console with a command `ctr + L`

We can use R to do simple arithmetic:

- division `/`
- multiple `*`
- addition `+`
- exponent `^`
- subtraction `-`

For instance, type the following in your console:

```
5+5
```

```
## [1] 10
```

Building a code

To create a script:

1. Click File
2. Click New File
3. Click R script
4. Give your script a name

> Challenge

- Create a script called “R-beginner-scripts” in your current project and put it in the `scr` folder

Creating objects in R workspace

To store results or data we need to assign it a name, using assignment operator `<-`

```
# now I tell R to store my height

height_cm <- 172
```

Now, position your cursor at the line of the above command and press `CTRL + ENTER` (Windows) or `COMMAND + ENTER` (Mac). In the environment (top right) you should see a variable `height_cm` and its value appear

Challenge!

1. Clear your workspace
2. Create a variable called **mass_kg** with a value of **50**
3. Convert mass_kg to mass in **pounds** (multiply by 2.2) and store the answer in **mass_lbs**

#rcatladies

3.

Functions and Packages

What are functions?

- Fundamental building blocks of R
- Built-in functions use input arguments to return a value or an output
- You can build your own functions for your specific needs!

`log()` is an example of a function to compute a logarithm.

You can also get information on how to use a function in RStudio by typing

```
?log()
```

So is **getwd()** we used earlier!

What are packages?

- Fundamental units of reproducible R code; include R functions, the documentation that describes how to use them and sample data (from http://r-pkgs.had.co.nz)

- Need to be **installed** first (once only). There are 2 ways: 1) **install.packages("package-name")** or from RStudio (right-bottom corner)

- Need to be loaded for every instance of R session using **library(package-name)** [note no "" this time]

Challenge !

1. Install packages **dplyr** and **ggplot2** using **install.packages()**

2. Load both packages (top of your script)

```
library("ggplot2")
library("dplyr")
```

```
## Warning: package 'dplyr' was built under R version 3.4.2
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

4.

Loading + inspecting titanic.csv

Loading data from csv into R

```
library("dplyr")
library("ggplot2")

titanic <- read.csv("Data/titanic.csv")

# tibble: a refined print method that shows only first 10 rows and all columns fit the screen

titanic <- as_tibble(titanic)
```

The above read.csv() function creates an object in the R environment that has:

- variables of a data set as columns
- observations as rows
- Rectangular

DATAFRAME
TIBBLE

Inspecting a dataframe

- `str()`
- `glimpse()`
- `head()`
- `nrow()`
- `dim()`
- `names()`
- `levels(dataframe$variable)`

#rcatladies

Challenge![together]

Inspect the titanic dataset using functions

- How many observations and variables does this dataset have?

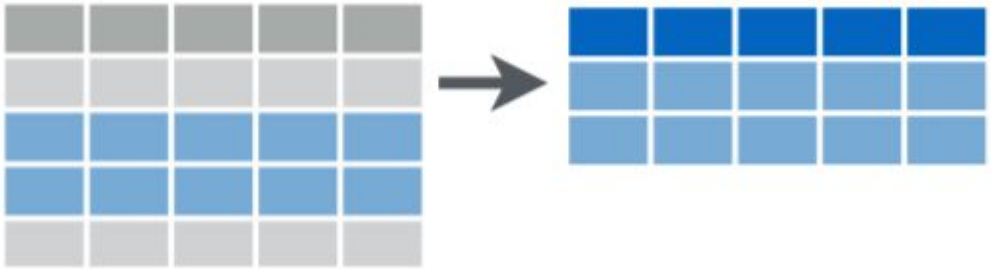
- What categories can be found in variable **Embarked**? Hint! Use levels()

5. Dplyr basics

https://dplyr.tidyverse.org/

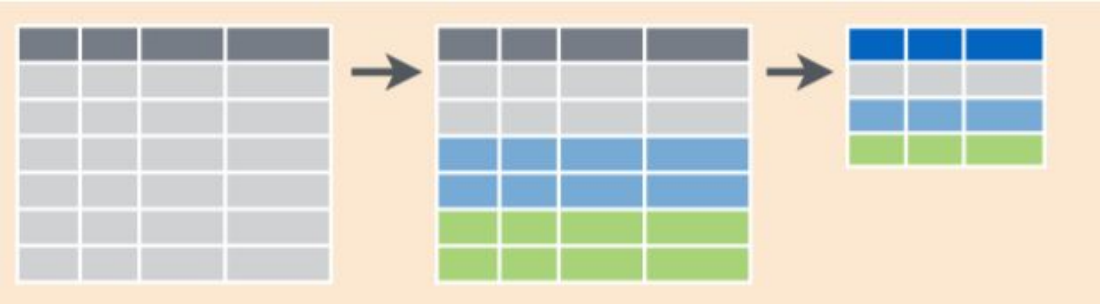
Picking variables (columns) with `select()`

Subsetting observations (rows) with `filter()`



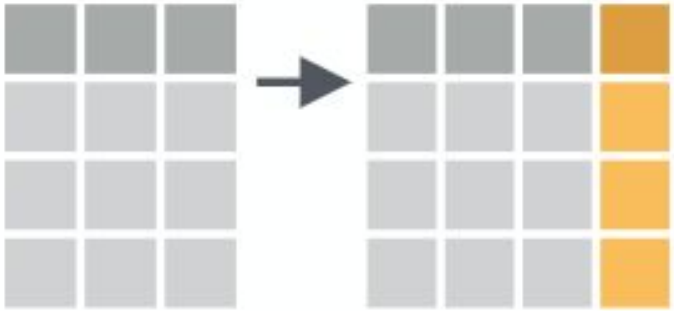
Summarizing data

`group_by()`



`summarize()`

Create new variables with `mutate()`



Challenge 5!

#rcatladies

6. Ggplot2 basics

GGPLOT2 package

Implements the grammar of graphics, a system for building graphs:

- **mapping** of data
- to **aesthetic** attributes (color, size, xy-position)
- using **geometric objects** (points, lines, bars)
- with data being statistically transformed (summarised, log-transformed)
- and mapped onto a specific facet and coordinate system

From Mine Cetinkaya-Rundel
(http://rpubs.com/minebocek/117428)

The basic anatomy of a ggplot...plot

```
library(ggplot2)
```

Remember to have the ggplot package loaded

Start by specifying the **data** with variables

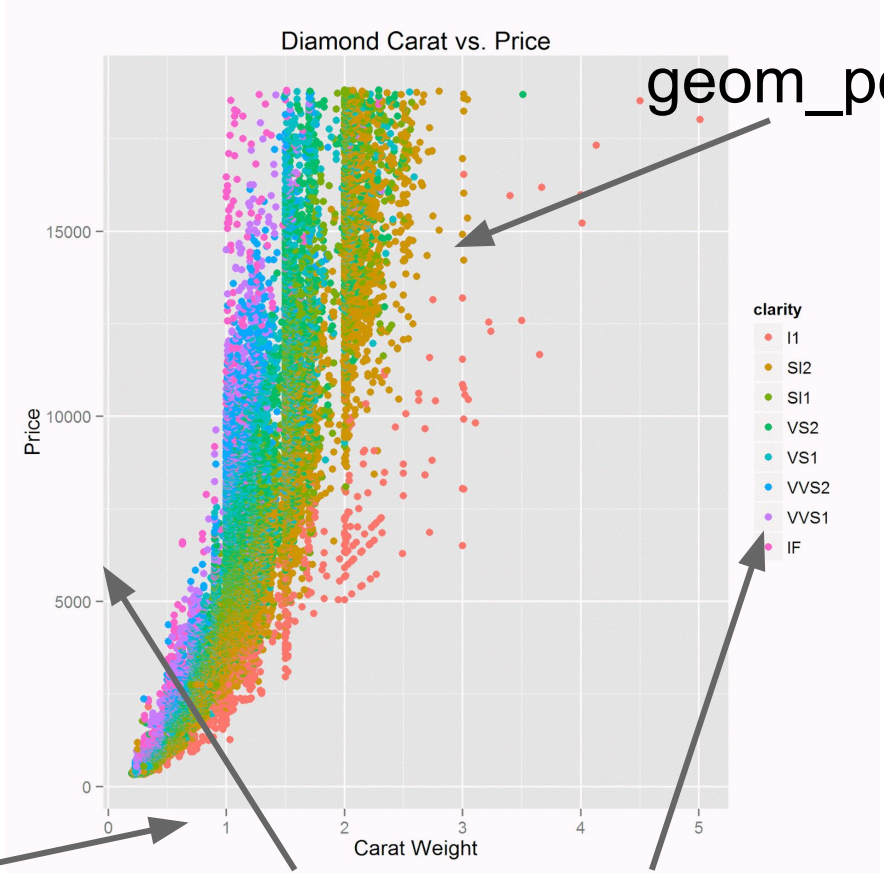
```
dat%>%
  ggplot(aes(x = ,y = ,...))+
```

Then specify what is to be plotted on
x and y | aes = aesthetics
(position of x and y)

```
geom_boxplot()+
geom_point()+
geom_bar()+
```

Then, choose at least one **GEOM** - aka geometric object
They specify how your data will be plotted, e.g.
geom_point() -> scatterplots
geom_line() -> line plots, for trend lines

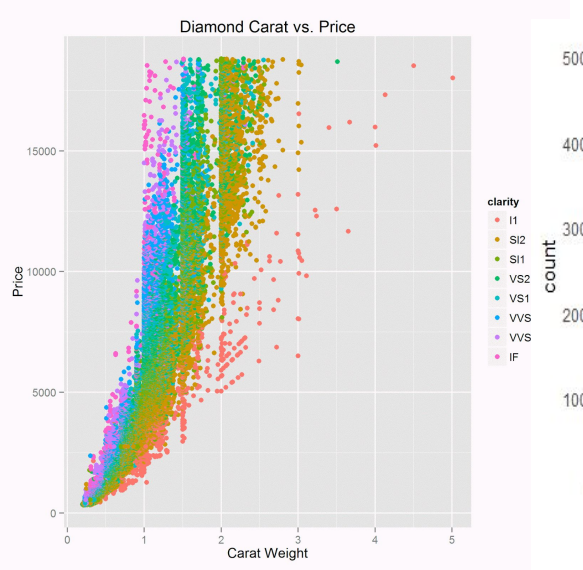
geom_point()



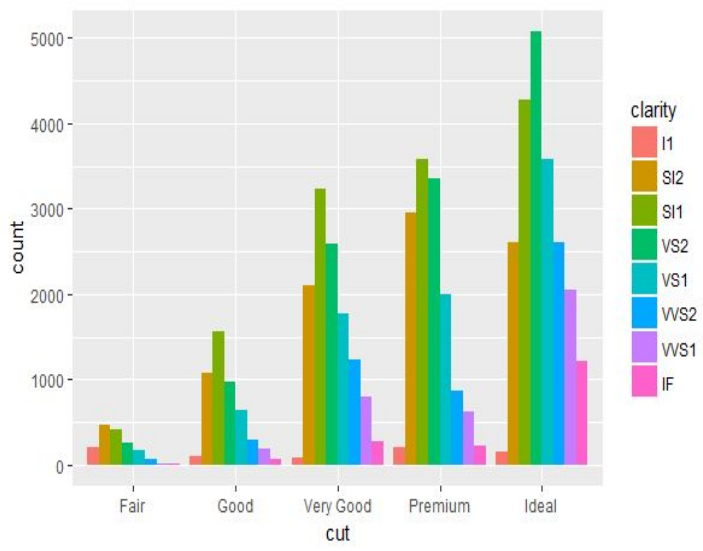
ggplot(aes(x = Carat Weight, y = Price, color = clarity))

Geom examples

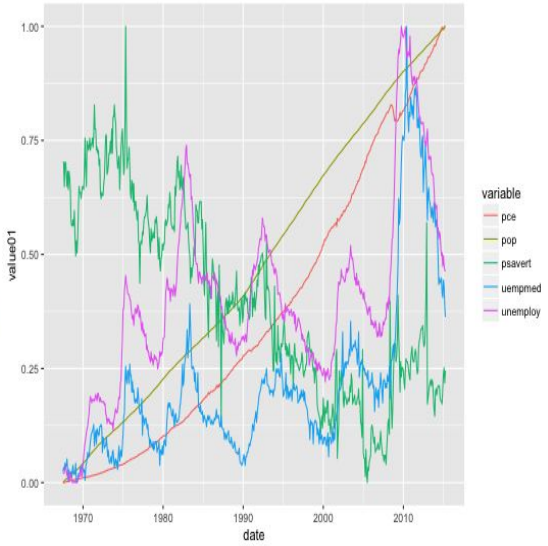
geom_point()



geom_bar()



geom_line()



Customizing

```
library(ggplot2)

dat%>%
  ggplot(aes(x = ,y = ,...))+

  geom_boxplot()+

  labs(title="my plot title")+
  xlab("my x label title")+
  ylab("my y label title")

  theme_bw()+
  theme_classic()+
  theme_dark()+
  theme()
```

By default, x and y labels will be the variable names in your “dat” [i.e. the dataframe with x and y], if you want to change you can with these command

You can also choose THE LOOK of your graph if you want. Here are some examples. It is also possible to create your own custom theme

Challenge !

Create your own graph!

Ideas:

1) Scatterplot of age (x) versus fare (y) with survived as colors
2) Passenger class (x) versus fare (y) with survived as colors

#rcatladies