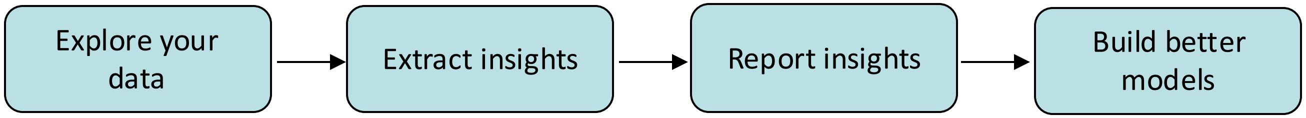
Creating plots with base R

Data visualization in R

- Data visualization is an essential part in the data analysis process:



Explore your data → Extract insights → Report insights → Build better models

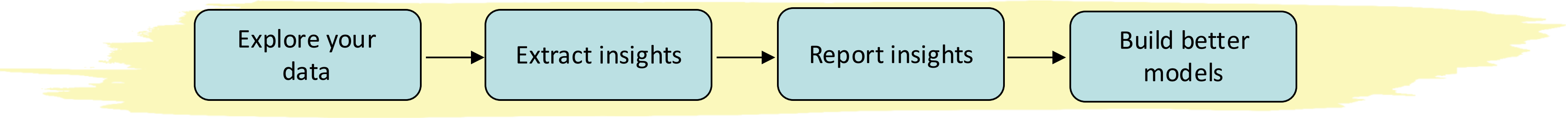
- R's plotting packages enable customized graphs and more:

Packages	Description
Base R Graphics/grDevices	Built-in plotting functionalities in R base
Ggplot2	"Grammar of Graphics": build your plot from various layers
Lattice	Provides functionalities for producing Trellis graphics
Plotly	Create Interactive Web Graphics via "plotly.js"

https://wiki.python.org/moin/NumericAndScientific/Plotting

Data visualization in R

- Data visualization is an essential part in the data analysis process:



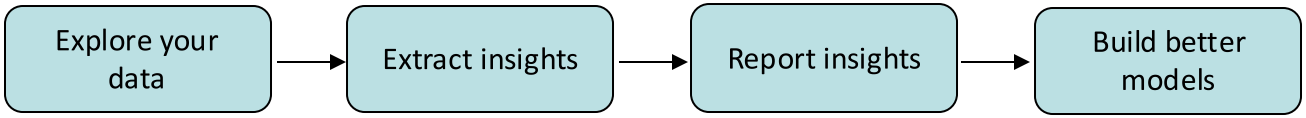
- R's plotting packages enable customized graphs and more:

Packages	Description
Base R Graphics/grDevices	Built-in plotting functionalities in R base
Ggplot2	"Grammar of Graphics": build your plot from various layers
Lattice	Provides functionalities for producing Trellis graphics
Plotly	Create Interactive Web Graphics via "plotly.js"

https://wiki.python.org/moin/NumericAndScientific/Plotting

Data visualization in R

- Data visualization is an essential part in the data analysis process:



Explore your data → Extract insights → Report insights → Build better models

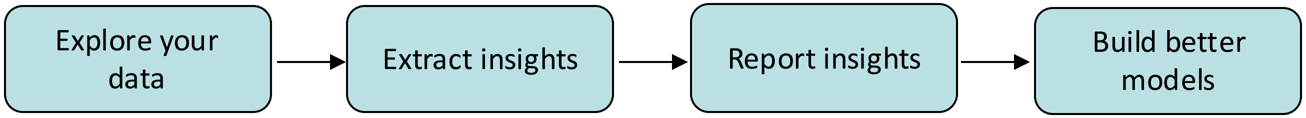
- R's plotting packages enable customized graphs and more:

Packages	Description
Base R Graphics/grDevices	Built-in plotting functionalities in R base
Ggplot2	"Grammar of Graphics": build your plot from various layers
Lattice	Provides functionalities for producing Trellis graphics
Plotly	Create Interactive Web Graphics via "plotly.js"

https://wiki.python.org/moin/NumericAndScientific/Plotting

Data visualization in R

- Data visualization is an essential part in the data analysis process:



Explore your data → Extract insights → Report insights → Build better models

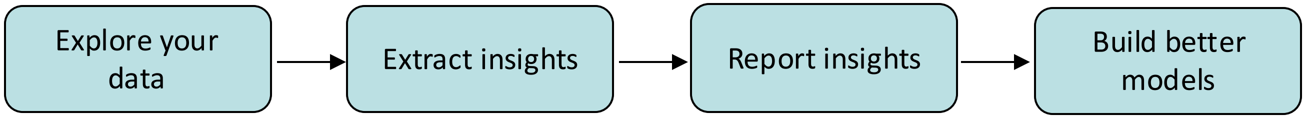
- R's plotting packages enable customized graphs and more:

Packages	Description
Base R Graphics/grDevices	Built-in plotting functionalities in R base
Ggplot2	"Grammar of Graphics": build your plot from various layers
Lattice	Provides functionalities for producing Trellis graphics
Plotly	Create Interactive Web Graphics via "plotly.js"

https://wiki.python.org/moin/NumericAndScientific/Plotting

Data visualization in R

- Data visualization is an essential part in the data analysis process:



Explore your data → Extract insights → Report insights → Build better models

- R's plotting packages enable customized graphs and more:

Packages	Description
Base R Graphics/grDevices	Built-in plotting functionalities in R base
Ggplot2	"Grammar of Graphics": build your plot from various layers
Lattice	Provides functionalities for producing Trellis graphics
Plotly	Create Interactive Web Graphics via "plotly.js"

https://wiki.python.org/moin/NumericAndScientific/Plotting

Data visualization in R
Why use R base for plotting?

- Base graphics are usually constructed in a piecemeal manner, with each aspect of the plot handled separately through a series of function calls.
- The R base functionalities for plotting have several advantages:
  - No additional package installation necessary
  - Easy to learn and thus, quick for standard plots
- However, there are also disadvantages:
  - As it provides less flexibility than alternative plotting packages, it is difficult for advanced visualization.

Data visualization in R
Why use R base for plotting?

- Base graphics are usually constructed piecemeal, with each aspect of the plot handled separately through a series of function calls.
- The R base functionalities for plotting have several advantages:
  - No additional package installation necessary
  - Easy to learn and thus, quick for standard plots
- However, there are also disadvantages:
  - As it provides less flexibility than alternative plotting packages, it is difficult to use for advanced visualization.

Data visualization in R
Why use R base for plotting?

- Base graphics are usually constructed piecemeal, with each aspect of the plot handled separately through a series of function calls.
- The R base functionalities for plotting have several advantages:
  - No additional package installation necessary
  - Easy to learn and thus, quick for standard plots
- However, there are also disadvantages:
  - As it provides less flexibility than alternative plotting packages, it is difficult to use for advanced visualization.

Good plots have 3 characteristics

Plots should be:

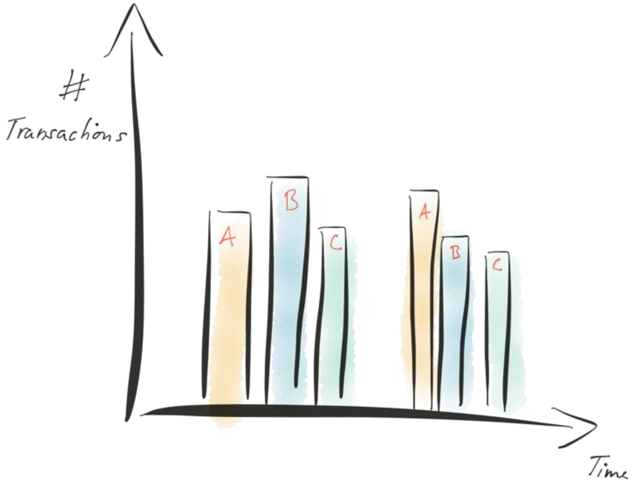
- Informative
- Easy to understand
- Visually appealing

How to plot:
Steps

1. Choose the plot type
2. Find the appropriate R function
3. Transform data
4. Create the plot
5. Improve aesthetic features of the plot
6. Save plot

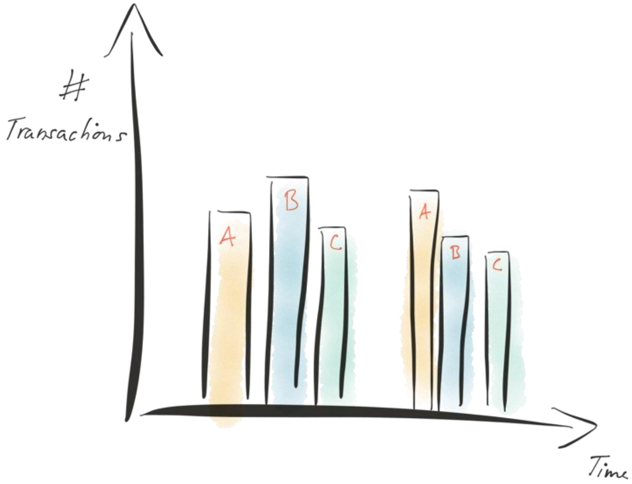
Step 1: Choose the plot type
Decide the best way to convey the information

- What do you want to show?
  - A single variable?
  - The relationship between multiple variables?
- Is your data continuous or discrete?

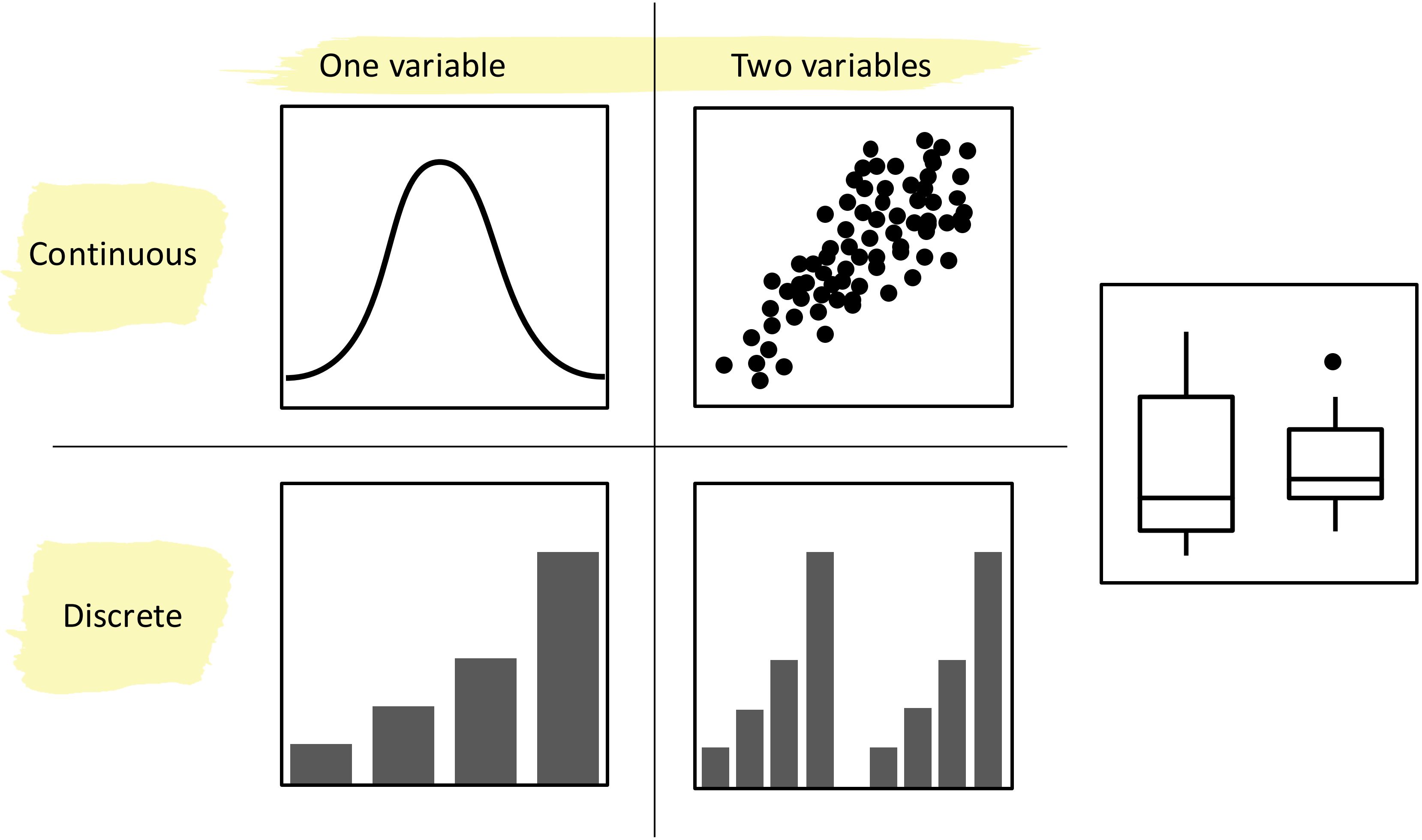


Step 1: Choose the plot type
Decide the best way to convey the information

- What do you want to show?
    - A single variable?
    - The relationship between multiple variables?
- Is your data continuous or discrete?



Different combinations of variables can be portrayed with different plot types



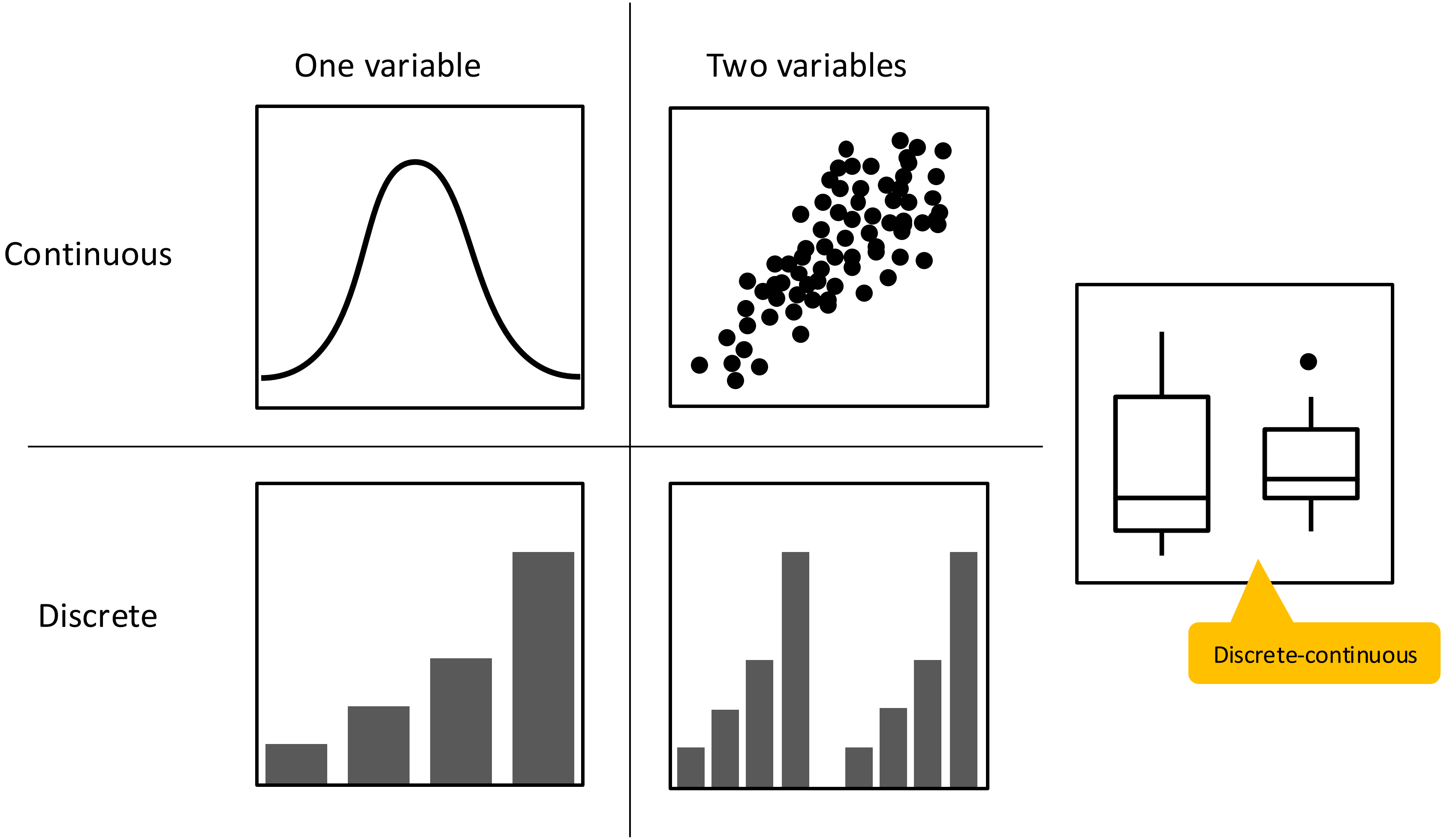
One variable

Two variables

Continuous

Discrete

Different combinations of variables can be portrayed with different plot types



One variable

Two variables

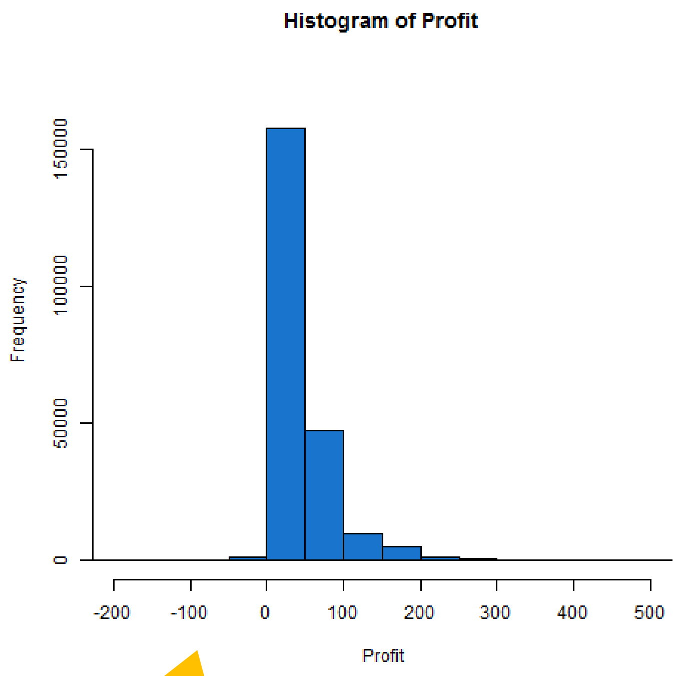
Continuous

Discrete

Discrete-continuous

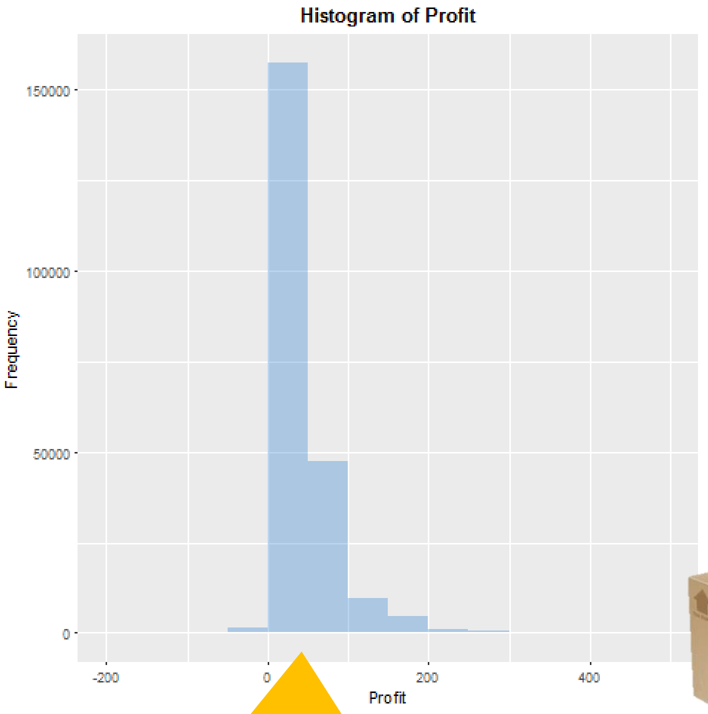
Step 2: Find the function – Base graphics and ggplot2 are the most used plotting tools

Base graphics



1 Base graphics is the built-in graphics toolbox

ggplot2 package

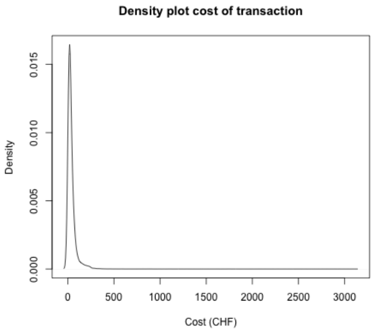


2 `ggplot2` is a very popular package that facilitates making plots by deconstructing them into layers

Step 2: Find the function – Base plots available in R

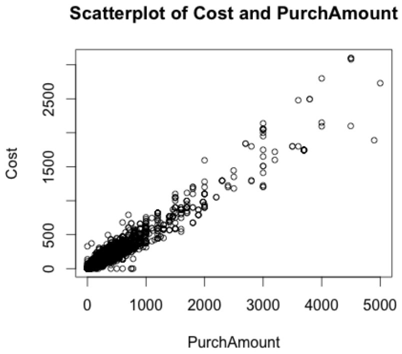
Regular plots (lines, density)

```
plot(density(x),
          ...)
```



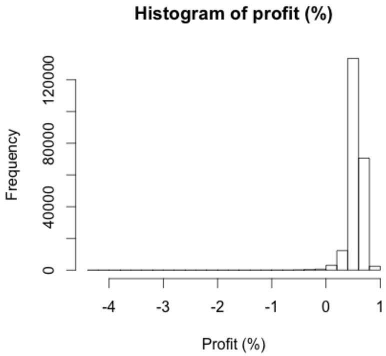
Scatterplot

```
plot(x,y,...)
```



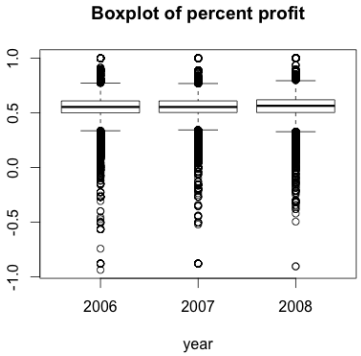
Histogram

```
hist(x,...)
```



Boxplot

```
boxplot(x,y,...)
```



… there's more!

Step 3: Transform data
Some graphs might require transformed data input

- It is quite rare that you can plot your data right away, i.e. certain **plots have requirements** on how the data should look like.
- In most cases it is **necessary to transform** your data before plotting it.
- Examples:

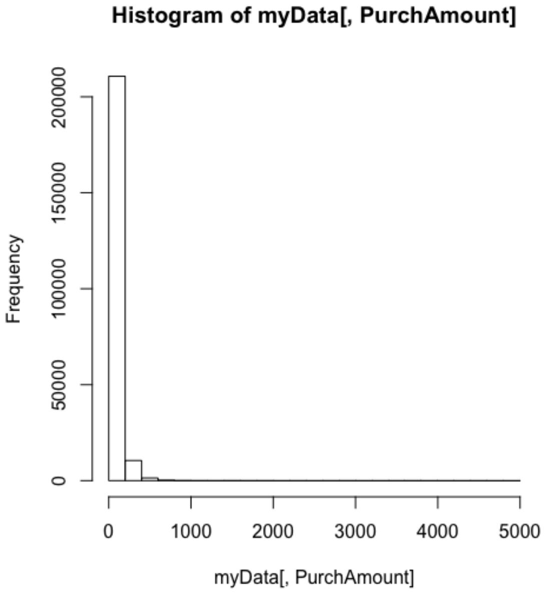
  Lecture: Basic techniques for investigating data objects

  Lecture: Advanced techniques for aggregating observations

  - Transform times and dates for aggregation of month or years
  - Group data for better overview
  - Logarithmic transformations for nicer distributions

Step 4: Create the plot (1/2)

Customer	TransDate	Quantity	PurchAmount	Cost
149332	15.11.2005	1	199.95	107.00
172951	29.08.2008	1	199.95	108.00
120621	19.10.2007	1	99.95	49.00
149236	14.11.2005	1	39.95	18.95
149236	12.06.2007	1	79.95	35.00
...	...	...	...	...



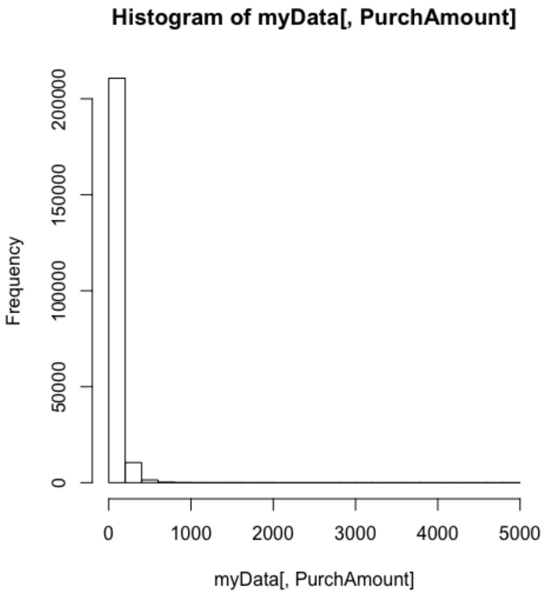
Multilayer principle:

Object to plot

```
hist(myData[, PurchAmount])
```

Step 4: Create the plot (1/2)

Customer	TransDate	Quantity	PurchAmount	Cost
149332	15.11.2005	1	199.95	107.00
172951	29.08.2008	1	199.95	108.00
120621	19.10.2007	1	99.95	49.00
149236	14.11.2005	1	39.95	18.95
149236	12.06.2007	1	79.95	35.00
...	...	...	...	...



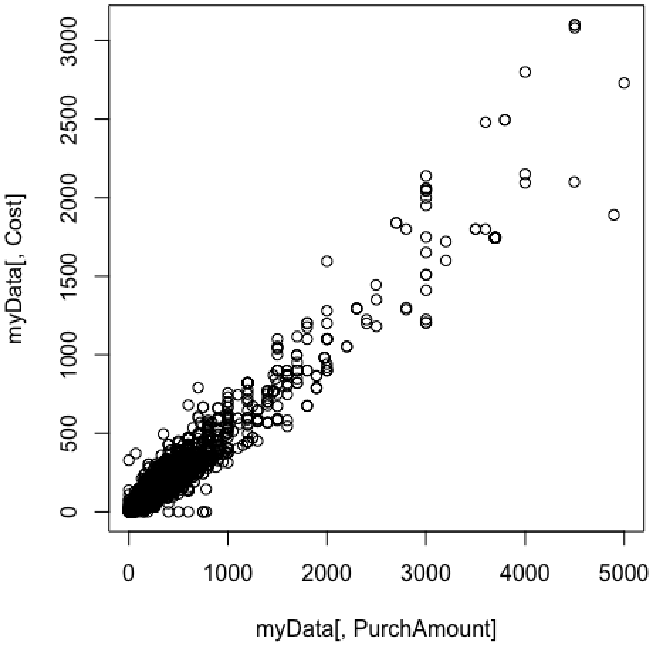
Multilayer principle:

Object to plot

```
hist(myData[, PurchAmount])
```

Step 4: Create the plot (2/2)

Customer	TransDate	Quantity	PurchAmount	Cost
149332	15.11.2005	1	199.95	107.00
172951	29.08.2008	1	199.95	108.00
120621	19.10.2007	1	99.95	49.00
149236	14.11.2005	1	39.95	18.95
149236	12.06.2007	1	79.95	35.00
...	...	...	...	...



x variable 1

y variable 2

```
plot(x=myData[, PurchAmount], y=myData[, Cost])
```

Creating plots with base R