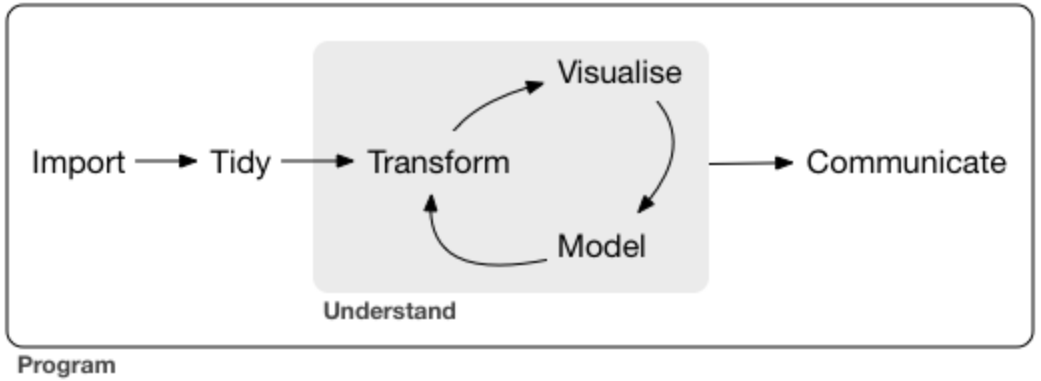
Data Exploration

(A lot of this material is from R for Data Science, a freely available book)



One of the first tasks is to better understand the data. We looked at that in the Bayesian context from the *epistemological* and *ontological* perspectives: how we think the data were created, and what properties exist in our data.

Often that means background reading, e.g., a literature review or domain modeling. What are your intuitions about how the data would be generated, e.g., for job application information?

Next, take the data and get it loaded into R or Jupyter. My favourite toolset is the `Tidyverse`, a set of libraries and philosophies for manipulating R data frames.

Loading data is initially easy but quickly becomes a major challenge as data gets larger and more complex. A non-trivial amount of effort can be spent wrangling data into the correct format. A big part of TidyR is to get the data into a "shape" that is easy to work with.

Let us consider the following datasets: http://promise.site.uottawa.ca/SERepository/datasets-page.html. They are mostly in a common data exchange format called ARFF.

Loading Data - Tibbles

We can begin by (after reading the dataset description) looking at common **descriptive** properties of the data.

Use R's `$` column notation to explore each column.

Tidy Data:

1. Each variable must have its own column.
2. Each observation must have its own row.
3. Each value must have its own cell.

Factors: factors are categorical data, possibly ordered, like "letter grade" (ordered) or country name.

Let's say we are interested in how well Halstead complexity `v(g)` predicts defects.

Filtering and aggregating

In Tidyverse language, use the TidyVerse chaining operator %>% (pipe) to group operations on dataframes/tibbles.

Useful operations include `mutate`, `filter`, `summarize/group_by`. I find these operations extremely powerful for manipulating the data. You can think of them as data frame analogues for SQL query statements.

I suggest you load your data into a data frame and then chain filters into it, rather than "fixing" the dataset in a separate step. That way you get to see all the data wrangling in one place. It can be *really* easy to miss a step and end up with outliers that are included, aggregations that don't make sense, etc.

Probability distributions and fit

the `ggplot()` library in R (and MatPlotLib in Python) are best in class visualization approaches with a vast amount of options and when combined with the filtering/aggregating, can do almost anything.

Use ggplot to explore your data and dimensionality as we showed in the R walkthru. Don't feel bad about using SO or references; I did preparing this tutorial!

Visualization can help understand what is happening. Descriptive stats such as mean, median, variance can give some numeric information conditional on the distribution involved. These are typically generalized as **location** and **dispersion**, e.g., where the big clump of data lives in the normal, and how spread out it is.

Correlation

Thinking back to our causal models, some of the variables are correlated, possibly causally. We should look for correlation because it is a great source of confusion; either implying some effect that is really due to a hidden mediator, or making us run extra analyses that are all based on the same underlying mechanism.

Checking Distributions with QQ Plots

Regression

(Find things)

OLS in Python - Geron p 106 Grus p 185

SGD alg Grus p 187

Regularization (Grus p 200)

Clustering

(Group things)

K-means in Python (Grus P 170)

Optimization

The simplest approach: GATE

G enerate from data, priors, fuzzing

A ssess how well that approach works, quickly

T ry (**e**valuate, maybe slow e.g. ask a human)

https://github.com/txt/ase19/blob/master/docs/optimize.md#top

Genetic Algorithms

NSGAII

Diff Evolution

JMetal