TEST-DRIVEN DATA ANALYSIS

Overview of Test-Driven Data Analysis

Test-driven data analysis (TDDA) is an approach to improving the *quality, correctness* and *robustness* of analytical processes by transferring the ideas of test-driven development from the domain of software development to that of data analysis, extending and adjusting them where appropriate.

A Methodology and a Toolset

TDDA is primarily a methodology that can be implemented in many different ways. Stochastic Solutions also develops an open-source (MIT-licensed) Python module, tdda, to providing tooling support for test-driven data analysis.

Motivation

High-quality data analysis right is hard. In addition to the problems common to all software development, with data analysis we typically face other challenges, including

- poorly specified analytical goals
- problematical input data—poorly specified, missing values, incorrect linkage, outliers, data corruption
- possibility of misapplying methods
- problems with interpreting input data and results
- changes in distributions of inputs, invalidating previous analytical choices
- reconciliation across multiple or changing data sources.

Key Ideas

Reference Tests. Reproducible research emphasizes the need to capture executable analytical processes and inputs to allow other people to reproduce and verify them. *Reference tests* build on these ideas by also capturing expected outputs and a verification procedure (a "diff" tool) for validating that the outputs are as expected. The Python tdda module supports verification of complex objects (e.g. graphs, dataframes) with exclusions and regeneration of verified reference outputs.

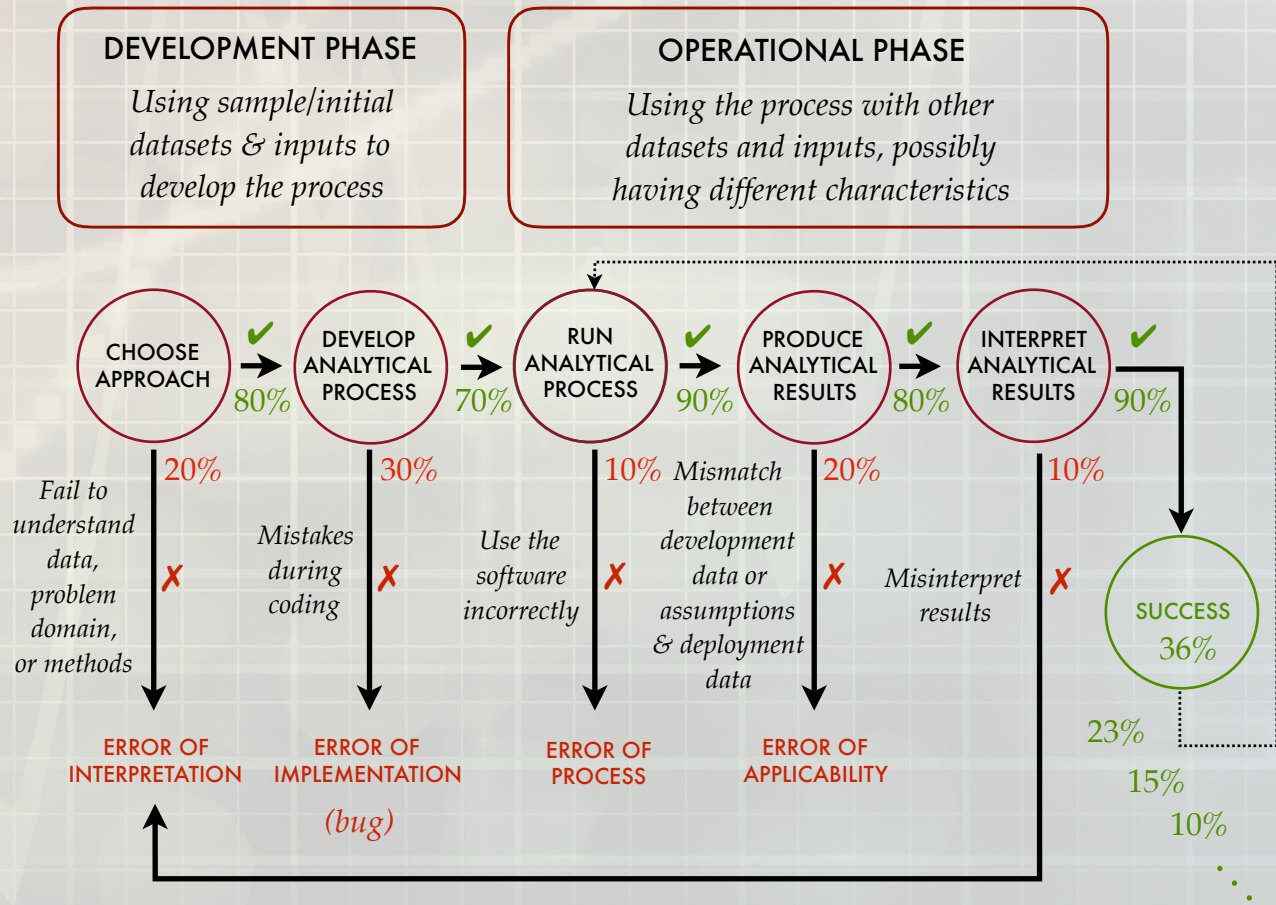
Constraint Discovery & Verification. There are often things we know should be true of input, output and intermediate datasets, that can be expressed as constraints—allowed ranges of values, uniqueness and existence constraints, allowability of nulls etc. The Python tdda module not only *verifies* constraints, but can also *generate* them from example datasets. This significantly reduces the effort needed to capture and maintain constraints as processes are developed and used. Think of constraints as *unit tests for data*.

Resources

Python TDDA library (tdda):

```
pip install tdda  or
git clone https://github.com/tdda/tdda.git
```

Blog: http://tdda.info
Twitter: @tdda0
Video: https://www.youtube.com/watch?v=FIw_7aUuY50



A typical analytical process with the kinds of errors that can occur at each stage. The percentages are (optimistic) possible success rates for each stage, leading to a 36% success rate first time, falling each time the process is re-used.