Smarter, shorter, AI Software

Tim Menzies
mailto:youremail@example.comtimm@ieee.org
http://yourhomepage.comhttp://timm.fyi

January 5, 2024

Contents

Chapter 1

Do We Understand AI Software?

For AI and SE, do we know what we are doing? Can we build AI software, succinctly, from just a handful of parts? And can other people understand, critique and improve that system?

Let's check. Given a few hundred lines of cade, can we build, say, an explanation system for a multi-objective semi-supervised optimizer. To put that another way, we want to learn human-readable rules for multi-goal problems, after labeling just a handful of examples. These rules have to be *effective*; i.e. they have to recognize parts of the problem space where (say) we see the fastest *and* cheapest car designs. And we're going to assume that this learner can only ask humans about 20 examples, or less (since if we demand more, humans get tired and make mistakes (or they do not have time to answer all our nagging questions).

1.1 Some Definitions

In this work, *data* are tables with *rows* and *columns*. Columns are also known as *features*, *aributes*, or *variables*.

Rows contain multiple X, Y features where X are the *independent variables* (that can be observed, and sometimes controlled) while Y are the *dependent* variables (e.g. number of defects). When Y is absent, then *unsupervised* learners seek mappings between the X values. For example, *clustering* algorithms find groupings of similar rows (i.e. rows with similar X values).

Usually most rows have values for most X values. But with *text mining*, the opposite is true. In principle, text miners have one column for each work in text's language. Since not all documents use all words, these means that the rows of a text mining data set are often "sparse"; i.e. has mostly missing values.

When Y is present and there is only one of them (i.e. $|Y| = 1$) then *supervised* learners seek mappings from the X features to the Y values. For example, *logistic regression* tries to fit the X, Y mapping to a particular equation.

When there are many Y values (i.e. $|Y| > 1$), then some weight $W_i$reports the *heaven* of that value. For this we want to minimimze/maximze then heaven is 0,1 respictively.

- *Clustering* algorithms find groups of rows;
- *Classifiers* find how those groups relate to the target a symbolic Y variables;
- *Classifiers* find how those groups relate to the target a numierc Y variables;
- *Optimizers* are tools that suggest "better" settings for the X values (and, here, "better" means settings that improve the expected value of the Y values).

Apart from W, X, Y, we add Z, the *hyperparameter* settings that control how learners performs regression or clustering. For example, a KNeighbors algorithm needs to know how many nearby rows to use for its classification (in which case, that $k \in Z$). Usually the Z values are shared across all rows (exception: some optimizers first cluster the data and use different Z settings for different clusters).

N things can be Too many choices, not enough time to look at them all.

- e.g. Hundreds of cars in a car yard, you try three, then buy one;
- e.g. You can't test everything – so you just test a few;
- e.g. Software has 10^9 of options – but you have time to try a few.

So lets apply *sequential model optimization*:

- **?**, **?**, **?**, **?**, **?**, **?**
- e.g. Hundreds of cars in a car yard, you try three, then buy one; Some terminology

$$\underbrace{y_1, y_2, ...}_{\textit{dependent variables, goals}} = f(\underbrace{x_1, x_2, x_3, x_4, x_5, x_6, ...}_{\textit{indendepentvariables}})$$

-
- But depndent variables are more expesnive to collect
- e.g. A supermarket has 100 apples. Which ones are tasty? So lets walk data incrementally:

1.2 Problem

Too many choices, not enough time to look at them all. line 1.

- adas

```
  function NUM.new(i,at,txt) -- --> NUM;  constructor;
  i.at, i.txt = at or 0, txt or "" -- column position
  i.n, i.mu, i.m2 = 0, 0, 0
  i.lo, i.hi = math.huge, -math.huge
  i.w = i.txt:find"-$" and -1 or 1 end

function NUM.new(i,at,txt) --> NUM;  constructor;
  i.at, i.txt = at or 0, txt or "" -- column position
  i.n, i.mu, i.m2 = 0, 0, 0
  i.lo, i.hi = math.huge, -math.huge
  i.w = i.txt:find"-$" and -1 or 1 end



function NUM.new(i,at,txt) --> NUM;  constructor;
  i.at, i.txt = at or 0, txt or "" -- column position
  i.n, i.mu, i.m2 = 0, 0, 0
  i.lo, i.hi = math.huge, -math.huge
  i.w = i.txt:find"-$" and -1 or 1 end


function NUM.new(i,at,txt) --> NUM;  constructor;
  i.at, i.txt = at or 0, txt or "" -- column position
  i.n, i.mu, i.m2 = 0, 0, 0
  i.lo, i.hi = math.huge, -math.huge
  i.w = i.txt:find"-$" and -1 or 1 end


function NUM.new(i,at,txt) --> NUM;  constructor;
  i.at, i.txt = at or 0, txt or "" -- column position
  i.n, i.mu, i.m2 = 0, 0, 0
  i.lo, i.hi = math.huge, -math.huge
  i.w = i.txt:find"-$" and -1 or 1 end


function NUM.new(i,at,txt) --> NUM;  constructor;
  i.at, i.txt = at or 0, txt or "" -- column position
  i.n, i.mu, i.m2 = 0, 0, 0
  i.lo, i.hi = math.huge, -math.huge
  i.w = i.txt:find"-$" and -1 or 1 end

function NUM.new(i,at,txt) --> NUM;  constructor;
  i.at, i.txt = at or 0, txt or "" -- column position
  i.n, i.mu, i.m2 = 0, 0, 0
  i.lo, i.hi = math.huge, -math.huge
  i.w = i.txt:find"-$" and -1 or 1 end

function NUM.new(i,at,txt) --> NUM;  constructor;
  i.at, i.txt = at or 0, txt or "" -- column position
  i.n, i.mu, i.m2 = 0, 0, 0
  i.lo, i.hi = math.huge, -math.huge
  i.w = i.txt:find"-$" and -1 or 1 end

function NUM.new(i,at,txt) --> NUM;  constructor;
  i.at, i.txt = at or 0, txt or "" -- column position
  i.n, i.mu, i.m2 = 0, 0, 0
  i.lo, i.hi = math.huge, -math.huge
  i.w = i.txt:find"-$" and -1 or 1 end
```

1.3 Test section one

1.4 What is Prolog?

- A programming language associated with artificial intelligence and computational linguistics.
- Based on formal logic.
- Declarative: Describe the problem, not how to solve it.
- Known for its ability to handle symbolic reasoning and knowledge representation.

some text here some text here some text here some text here some text here

asdsa

1.5 References

Bibliography