Notes for ‘Using Perl with Semi-Structured Data’ Talk

Samantha E Montag, MS
Tuesday, November 14, 2017

Introduction

• First of all, thank you for coming
• I have wanted to talk about this unique bit of code I wrote early on to clean semi-structured data in SAS for a while now
• Feel free to interrupt me if you have questions

Outline

• What is semi-structured data?
• What is Perl?
• How to use Perl Expressions
• An example with renal biopsy reports
• An example with human leukocyte antigen (HLA) lab reports

Types of Data

Common Data Structure Model

1. Data is often broken down into two groups: Quantitative and Qualitative
2. However, I would suggest an alternate contrast is more appropriate in this example: Structured and Unstructured
   a. We typically require data to be in a structured form to analyze:
      • Continuous
      • Ordinal
      • Categorical
      • Binary

Data Structure as a Spectrum

1. Not all data collected is in this format though
   a. Studies often capture information with unstructured ‘free text’ fields
   b. For analysis this data must then be transformed it into structured shapes
      • for example medications are often captured with full names that must be categorized
   c. But what about when the useful information is buried inside of a paragraph of text?
      • notes in charts are a good example
   d. We often don’t use that information because there is a burden to extracting that information into discrete fields
      • barring something like a chart review to manually do it
2. Sometimes these paragraphs of unstructured text actually have underlying patterns that we can use to extract that information
   a. Humans are creatures of habit and often do things in patterns without necessarily intending to
Semi-Structured Data Example

1. Example:
   a. Alternately, the infrastructure in data entry might incorporate forms or templates to use to ensure consistent reporting
      • This information isn’t collected in a vacuum, someone needs to use it and be able to get information out of it

Perl

What is Perl?

• Perl is a scripting language developed in 1987 by Larry Wall
• It gained popularity for its ‘regular expressions’ and string parsing abilities
• The ‘regular expressions’ are “a set of functions that implement pattern matching using the same syntax and semantics as Perl 5”
• Perl does not explicitly stand for anything however some use the backronym: “Practical Extraction and Reporting Language”

Character Syntax

<table>
<thead>
<tr>
<th>Character</th>
<th>Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>/.../</td>
<td>Starting and ending regex delimiters</td>
</tr>
<tr>
<td></td>
<td>OR</td>
</tr>
<tr>
<td>()</td>
<td>Grouping</td>
</tr>
<tr>
<td>.</td>
<td>Match any character</td>
</tr>
<tr>
<td>\w</td>
<td>Match a word character (alphanumeric plus “_“)</td>
</tr>
<tr>
<td>\W</td>
<td>Match a non-word character</td>
</tr>
<tr>
<td>\s</td>
<td>Match a whitespace character</td>
</tr>
<tr>
<td>\S</td>
<td>Match a non-whitespace character</td>
</tr>
<tr>
<td>\d</td>
<td>Match a digit character</td>
</tr>
<tr>
<td>\D</td>
<td>Match a non-digit character</td>
</tr>
<tr>
<td>[...]</td>
<td>Match a character in the brackets</td>
</tr>
<tr>
<td>[^...]</td>
<td>Match a character not in the brackets</td>
</tr>
<tr>
<td>[a-z]</td>
<td>Match a character in the range a to z</td>
</tr>
</tbody>
</table>

• all regular expressions start and end with forward slashes
• the vertical bar stands for OR
• the others allow for character selection

Example Character Syntax

• /mom/ would identify **mom** only, not mama or mother
• /mom|mama|mother/ would identify any of the three exactly
• /m[oa]m/ would identify **mom** and **mama**, but not mother
• /m[oa][nt]/ would identify **mom**, **mama**, and **mother**

Repetition Syntax
<table>
<thead>
<tr>
<th>Character</th>
<th>Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>Match 0 or more times</td>
</tr>
<tr>
<td>+</td>
<td>Match 1 or more times</td>
</tr>
<tr>
<td>?</td>
<td>Match 1 or 0 times</td>
</tr>
<tr>
<td>{n}</td>
<td>Match exactly n times</td>
</tr>
<tr>
<td>{n,}</td>
<td>Match at least n times</td>
</tr>
<tr>
<td>{n,m}</td>
<td>Match at least n but not more than m times</td>
</tr>
</tbody>
</table>

- sometimes you want to identify if something appears once or more
- provides additional flexibility to regular expressions

**Example Repetition Syntax**

1. perhaps you are looking to pull out zip codes from an address:
   - `/\d{5}/` would get any 5 digit code
   - but maybe you want to be more specific
   - maybe you want to identify addresses in Illinois using the zip code `/6[02]\d{3}/`

2. maybe instead you are looking for phone numbers:
   - `/\d{3}-\d{3}-\d{4}/` would pick up any combination following this standard entry 312-503-3676
   - but `/([2-9]\d{2})-[2-9]\d{2}-\d{4}/` takes into account that the first digit of the 3 digit groups can’t be a 1
     - further other delimiters could be used

**SAS Functions**

<table>
<thead>
<tr>
<th>Function</th>
<th>Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>regex-id=prxparse(perl-regex)</td>
<td>set regex-id to use as reference in other prx functions</td>
</tr>
<tr>
<td>position=prxmatch(perl-regex</td>
<td>regex-id,source)</td>
</tr>
<tr>
<td>new-string=prxchange(perl-regex</td>
<td>regex-id,n,source)</td>
</tr>
</tbody>
</table>

- there are other sas functions, but these are the main three in SAS
- prxparse is used to establish the regexID to reference (if you want to reference the same thing over and over)
- prxmatch is the perl equivalent of an index function; both return the starting position of the string you are looking for
- prxchange alters the old string to a new string with changes you specify
  - the regexid in this case is a bit different, it has 3 parts:
    1. starts with s/
    2. then the regex to search for followed by /
    3. then the regex of how to replace it followed by /

**Contrast with Common SAS Expressions**

- for example, you are cleaning medication data and anyone who took any of the following medications is considered to have been on a particular course of treatment
• or maybe, you are looking for causes of death that include particular phrases
• what about when you are given a dataset generated in excel and there are some typos in the diagnosis (because it is long and hard to spell and once the typo was in, they kept referencing it)
• essentially Perl provides...
  – a more concise way to search text
  – greater flexibility than typical SAS functions like index

Real World Examples

Kidney Biopsies

The Question?

• Clinicians wanted to know if the criteria captured in the kidney biopsies were associated with graft or patient survival
• I’m thinking alright, they have to write up the information somehow for the clinician to make a decision

The Raw Data

• They gave me this data and at first I said no way, how was I supposed to get anything out of this

The Template

• But one of the clinicians codes in R and said, hold on, it isn’t completely free text, most of the doctors who write these reports follow a template

Within the Raw Data

• And upon looking further I saw that many of these reports did contain a section that followed this template

Extracting the Information

1. trim the preamble
2. find the start and stop points for each piece
   • then calculate the distance between them
3. pull the useful information out from these pieces

This sounds straightforward but the template wasn’t exactly the same in every case. Additionally, I did this in individual data steps and created many derived variables to check that the code I was creating worked. This was important because it wasn’t always working for everyone.

I started using an index function and quickly realized that additional options were needed.

The prxmatch function became more practical and allowed flexibility in the search.

I then used substr and scan to split this single variable of text into its individual pieces and pull the useful information from after the colon.
Applications

Using the fields from this template, we can determine if a patient has sub-acute or acute rejection of the transplanted kidney.

This code has been used in a number of projects: - Late Outcome Surrogate Score® - Mini Kidney - Astellas Astagraf - projects using EDW data

Thus far some of the components of this template have shown associations with the outcomes, but further research is needed. Transplant also has designed several upcoming grants taking into consideration the new ease of availability of this data (previously done with labor intensive chart review).

HLA Reports

The Question

- Clinicians wanted to know if donor-specific human leukocyte antigen (HLA) test results were associated with graft and patient survival
- Specifically, if the presence of donor-specific HLA are causing rejection
- One of my majors in undergrad was Biology with a concentration in Physiology so I took immunobiology
  - this helped provide a lot of context on this project and helped me communicate effectively with the clinicians about issues that arose
  - simply put: your body has a signature encoded in your DNA to help determine self vs non-self
    - this signature is determined by approximately 6 loci on chromosome 6, of which you have 2 copies, therefore it is a 6-12 locus passcode since the two chromosomes could have the same gene in a particular locus
    - when something is not recognized your immune system attacks it and remembers it for the future
    - this triggers an inflammatory response like when you get a scratch
    - when you get a transplant, immunosuppressants suppress this immune response
      - having an immune response to an organ can cause it to fail which is the opposite of what we want
    - however the immune response is strong and immunosuppressants can only do so much
    - oftentimes organs are selected to have some compatibility between the signatures for the donor and recipient but there are rarely perfect matches
    - the clinicians wanted to know if having one donor-specific HLA vs more (considering the strength of the response) had different effects on the outcomes
- But again, I’m thinking what does the data look like, how can we capture this in an analyzable way?
  - there are two of each of 6 options for the donor and recipient, how do we know if there is agreement

The Raw Data

- They gave me this data and there are some clear patterns but I wasn’t sure what I was looking at
- After talking with the team at the lab that generates these reports, I learn that they use drop-down menus to add sentences with blanks to the notes field
- Additionally we learn that Class I (A,B,C) and Class II (DR,DP,DQ) results are analyzed separately

Extracting the Information

1. Split results into Class I and Class II results
2. Break the block of text into sentences
   - determine if donor specific
   - determine results
   - determine titer
• determine counts of results
• determine counts of titers
• determine which locus are discussed; this step is what required Perl

Again, I did this in individual data steps and created many derived variables to check that the code I was creating worked. I initially used an index function but there were so many options that I was making mistakes and something simpler was needed. The prxmatch function helped abridge code that initially took 45 lines of code down to 15 lines, from 5200 characters to 1250.

Applications

This data really hasn’t been published in this quantity by other institutions. Limitations include how to handle the data and how to establish clear questions using this data.

We have presented the epidemiology of this data as well as some preliminary findings. - There were demographic differences between those who had DSA at transplant, those who developed DSA during follow-up, and those who didn’t develop it * Patients with DSA at transplantation, or who developed dnDSA, were numerically more likely to be younger, female, black, and have a higher PRA score, and were less likely to be privately insured, compared with patients who never had DSA. * Those who had DSA at transplantation were numerically more likely to have had a prior transplant than those who developed dnDSA during follow-up, or never had DSA. - We found that while overall patients with new DSA had similar outcomes to those without DSA, contrasted with those who had DSA at transplant, this pattern did not persist at the locus level. * Developing DSA appeared to be more strongly associated with graft survival than patient survival * This makes sense because you can survive without the graft under medical supervision (dialysis)

However this data is still controversial and further research is needed.

Future Directions

• Both of these codes were validated by hand using a sample with over 95% agreement; further half of the disagreement was human error that was corrected by the code - there is always room for improvement, particularly the Kanwar biopsies
• Both codes allow for use of data that had previously not been widely used/published on in such large samples
• We would like to continue the research to answer questions that have not been answerable thus far
• Finally, I want to continue to improve my skills using Perl in SAS and potentially learn how to use it in R - especially since EDW now can use R code

Conclusions

• Perl functions in SAS can provide a useful way to parse text when cleaning data for analysis

Thank You For Listening