Table::Hack 表形式データの前分析ソフトウェア

my recommendations on research & development tools.

下野寿之†

キーワード：ビッグデータ、データ分析、前分析、表形式データ、TSV、タブ文字

1. まえがき

近年多くの企業が取引や在庫管理などで使ったデータを蓄積するようになりました。また、蓄積されたデータに対して簡単な分析のできるソフトウェアは多数世の中に存在します。ところが、蓄積された大きなデータをうまく分析するソフトウェアが存在しないように思います。Microsoft Excel (以下、エクセル) は約100万行を超えると動きませんし、R言語もPythonのPandasもメモリーサイズを超えるような大きなデータはロード不能になってしまいます。データベースにSQL言語のクエリを書いて問合せをしたり、大きなテキストデータに対してUnixまたはLinuxのコマンドを駆使したりして分析をするにしても、意外にそれらの機能は限られています。例えば、2個の変数の値の組合せが何個出現したのか数えてクロス集計表を作成することは、エクセルだとピボットテーブルの機能ですが、SQLでもUnix/Linuxコマンドでも現状困難です。

分析を期待されたデータが数多く世の中に埋まったまま活かすことができないという困った状況を打開したいと思い、新しいソフトウェア (以下ソフトと短縮します) を私は作ることにしました。どのようなソフトがあれば再利用性が高いかを考え、表形式のテキストファイルでよく使われるファイル形式である、CSV形式またはTSV形式のデータを処理対象としました。エクセルの表や多くのソフトのSQLの出力は、クリップボードにコピーすればタブ文字で区切られたTSV形式となりますので、コピペ (コピーアンドペースト) だけで新作ソフトと両方向にデータの受け渡しもできます。

その新しいソフトは企業の基幹システム等でありがちな古いシステムでも動くようにPerl 5.14 (2011年公開) もしくはそれ以前と互換性を保つように開発を行い、CPANにアップロードしました。インターネットが使える環境であれば、コマンドライン端末でcpan Table::Hackを実行すれば、すぐにインストールして使うことができます。

2. 新しいソフトウェアTable::Hackの設計

大きなデータを分析するための既存のソフトに対して、思いつき得る分析目的が達成できるように、データを整形してインポートしたりアップロードしたりするのは、現状実に大変です。分析プロジェクトに使う期間に対して前処理が80%から95%を占めるという話があります。しかし、さらにその前に与えられたデータをよく理解するための作業が必要です。このような蓄積されたデータをまず理解するための作業を「前分析」と名付けたいと思います。前分析をしないと、分析をする計算機環境を作ることすらままなりません。私のソフトは前分析に力を入れています。

テキスト形式のファイルをそのまま扱うことについては、既存のUnixのコマンドはなかなか便利です。しかし、ファイルの先頭行が列名の並びで2行目からデータが始まるテキストファイルの場合、例えば、ソート (整列) の処理はちょっと不便です。テキスト処理で古く (1977年) から使われているAWK言語を使って、コマンドライン上で `awk 'NR==1 {print} NR>1 {print | "sort"}'` なんて器用なことをすれば良いのかもしれませんが、そんなプログラミングをデータ分析の最中に始めたら、デバッグに多くの時間が取られます。エクセルで関数を何個も組み合わせ、何のテストもなくすぐ使える人は、きっとどこにもいないでしょう。結局デバッグの作業のために、本質的な作業の20倍くらい時間がかかるのではないかと思います。

そういう訳で、もっと根本的に簡単な仕組みが必要だと思います。私のソフトはUnixコマンドのように、コマンドとオプションと必要に応じてパラメータの組合せで、素早く済ませたい処理を素早く書けるようにしました (コマンドラインで、コマンド名の後ろに必要な数だけパラメータを持ったオプションを必要な数だけつなげて入力すると、所望の動作をします)。必要な場面で迅速に使えることを重視し、思い出しやすい名前 (基本的に英単語2個の組合せかつ10文字以内) で、一度でも使うと動作が理解しやすく、既存のコマンドと遜色のない計算速度を持つようにしています。

また、今までデータ分析で感じてきたような不便さを避けるため、いくつかの親切設計を盛り込みました。例えば、

† 株式会社デジタルガレージ (DG Lab.)

"A Pre-analysis Toolset for Tabular Files" by Toshiyuki Shimono (Digital Garage, Inc. (DG Lab.), Tokyo)

表1 大きなデータの分析をする観点からのソフトウェアの比較（特に最終行は主観的なものです）

ソフト	Excel	R	Pandas	SQL	Hadoop	Unix系命令	Table::Hack
主要な用途	表計算	統計解析	数値計算	DB操作	分散処理	file操作	前分析
着手容易さ	◎	○	○	△	×	○	Unix同様
透明性	×	○	◎	ペンダ依存	成熟途上か	○	○
経年互換性	△	△	先行き不明	○	先行き不明	◎	△?
巨大データ	×	メモリーサイズの制約		要設定	◎	○	○
列の指定法	相対/絶対etc	列名も列番号も可能		列名	列名	列番号が主	柔軟
悪く言うと	泥沼化容易	freeze多め	古い環境×	DB設計面倒	要専門知識	諸刃の剣	開発力不足

計算がなかなか終わらない時に「どこまで処理が進んだのかかわからないけれど、時間が掛かりすぎているから処理を強制終了しようかどうか」などと迷った経験を持つ人はたくさんいることでしょう。そのための対策として、いくつかのコマンドは、計算中断によく使われるキー操作であるCtrl + Cが押下されたら、まずは「何件のデータをすでに読み込んだ」などの進行状況を表示して、本当に計算停止したい場合はCtrl + ¥を押すことの指示を表示します。マニュアルの表示内容が多いと、ヘルプが必要な切迫した場面でこそ文章の量に圧倒されますので、--help optionというオプションとパラメータの組合せにより、そのコマンドのオプションの説明だけを簡潔に表示するようにしました。また、多くの端末のコマンドラインは、ANSIエスケープコードで文字やその背景に着色ができるので、わかり易いように色を多用しています（すぐ色を消すこともできるようにしています）。作成した多数のコマンドは共通したオプションを持ちます。通常のアルファベットを用いたオプションだけだと、提供するコマンド間で共通するオプションはわかりにくいと思われるので、==でデータの1行目の特別扱いの指定、-/で区切り文字の変更などの記号を用いたオプションも多用しています。

ところで、Unix哲学というものを覚えてしょうか。いろいろな派生形がありますが、Peter H. Salus 著のUNIXの1/4世紀によりますと、それは以下の3点です。

- (1) 一つの機能を確実に実行するプログラムを書く
- (2) 他のプログラムと一緒に動作するプログラムを書く
- (3) テキストストリームを処理するプログラムを書く

これは議論があるかもしれませんが、2018年にいたってなおUnix哲学に基づかないソフトウェアを使うとデジタルデータに対して「何か不具合の心配があっても迅速な検証手段がない、間違いがあっても気づきにくい、気付いたとしても訂正手段がない」という不具合がすぐに起きると思います。新作ソフトTable::Hackはそのような不具合を世の中からなくすことを目指しています。

3. Table::Hackの各コマンドの機能

Unix系のコマンド(プログラム)は便利で利用性の高い汎用的な機能をすでに多数提供しています。less, wc,

gawk, cut, head, tail, grep, sed, od, iconv, nkf, file, diff, sdiff, sort, uniq, paste, column, foldをパイプやプロセス置換と言われる構文とともに使いこなすと何かと役に立ちます(プロセス置換は、“<”と”)の間にコマンド文を挟んで使います。そのコマンド文の出力結果を持つファイルのように振る舞います。中間ファイルを残さなくて済む便利な機能です。それでもなお、先に述べた不具合を解決するには機能が足りないの、汎用性と再利用性の高いひとまとまりのソフトウェアが必要であると考え、「この機能がないとデータ分析はやってられない」の繰り返しで、数十個のコマンドを作りました。本稿ではその一部のコマンドを紹介します。

【expskip】

中身が何かわからない大きなテキストデータに対して、1, 10, 100, 1000, .. 行目だけをすぐ抽出できたら何と便利なことでしょう。用意されたさまざまなオプションを組み合わせると、抽出する行を1, 2, 4, 8, 16, 32, .. 行目とすることも容易です。何も指定しない場合は1, 2, 5, 10, 20, 50, 100, .. 行目となります。特に指定のない限り、最初と最後のそれぞれ3行は出力します。

【colorplus】

データの入ったテキストファイルを、例えば、colorplus -t5によって列ごとに色をつけることができます(図1)。データを確認する際によく使われるless等で閲覧する時に、例えば、左から11列目や23列目のデータが気になったら即座に目で追うことができます。

図1 colorplus -t5の実行結果の例

```
$ wc -lwc [af-hj]*[0-9]* | colors -3
4625970 71557089 739668527 all0923.tsv
60 301 2398 f4
1219 9110 294593 ff分析0709.xlsx
108187 858334 30738967 findings-140309055018-p.pptx
13178 102580 3329990 findings0626.pptx
6 42 3361 gimp-2.8.12.tar.bz2.torrent
196 457 11244 hansoku4.html
663 3996 131756 homework02a1.pdf
247 1147 35000 homework03P1.pdf
222 963 48128 houseFormJapanese360201.doc
4749948 72534019 774263964 total
```

図2 colorplus -3でレイアウトを崩さずに数のみ着色

```
~ % cat olympic.tsv | expskip -g | column -t
1 西暦 大会名 金 銀 銅 計 順位
2 1912 ストックホルム 0 0 0 0
3 1920 アントワープ 0 2 0 2 17
5 1928 アムステルダム 2 2 1 5 15
10 1960 ローマ 4 7 7 18 8
20 2004 アテネ 16 9 12 37 5
21 2008 北京 9 6 10 25 8
22 2012 ロンドン 7 14 17 38 11
23 2016 リオデジャネイロ 12 8 21 41 6
~ % colsummary --n2 olympic.tsv | sed 1d
1 22 1968.182 西暦 1912..2016 1996, 1932 , 1(x22)
2 21 0.000 大会名 アテネ..東京 ロサンゼルス, シドニー , 2, 1(x20)
3 14 6.455 金 0..16 4, 0 3(x2), 2(x4), 1(x8)
4 11 6.091 銀 0..14 8, 6 5, 4..2(x2), 1(x6)
5 13 7.409 銅 0..21 7, 8 3(x3), 2(x3), 1(x7)
6 15 19.955 計 0..41 18, 25 4, 3, 2(x2), 1(x11)
7 12 10.227 順位 , 3..23 5, 8 4, 3(x2), 2(x3), 1(x6)
```

図3 expskipとcolsummaryの使用例

colorplus -3で数値のみ下位から3桁ごとに着色可能です(図2)。

colorplus -0でテキストからすべてのANSIエスケープコードの色情報のみを除去することもできます。指定した正規表現のみを指定した色で着色することもできます。

【colsummary】

一つ一つのテーブルを手早く理解するための、最も実用性が高いコマンドの一つです。テーブルの各列について、①異なる値が何通り出現するか、②出現値の最小値と最大値、③最頻値を上位いくつか、④それらの出現回数、などを算出します。これらの統計値は、一つ一つの列に対して既存のコマンドで出すこともできますが、何列もあるテーブル全体に対するコマンド一発の実行で整理された形で出力するのに比べると、非常に大きな手間が掛かることでしょう(図3)。

「コマンド名は英語2単語の組合せとする」という原則に沿っていませんが、こういうコマンドも用意しています。

【freq】

どんな文字列の行が何個あったかを数えます。同じ機能はUnixコマンドを組み合わせて、sort | uniq -cでも可能ですが、この方法はソートの計算を伴うのでスピードがかなり落ちてしまいます。他、出力結果の順序の指定の仕

```
~/Box Sync % headmit **/*.g19 | cols -p 13 | freq -f -r -s -%
CUMRA ACCUM RATIO FREQ LINE_STR
99.97% 661916 99.972% 661916 200 OK
99.98% 661981 0.010% 65 404 Not Found
99.99% 662822 0.006% 41 500 read timeout
99.99% 662862 0.006% 40 503 Service Temporarily Unavailable
100.00% 662883 0.003% 21 500 Can't connect to twitter.com:443
100.00% 662894 0.002% 11 500 Can't connect to twitter.com:443 (timeout)
100.00% 662898 0.001% 4 500 Internal Server Error
100.00% 662899 0.000% 1 500 Server closed connection without sending any data back
662899 lines processed, 662899 lines are counted, 8 different lines appears. (freq)
```

図4 freqの使用例

方や累積和とその割合の表示などさまざまなオプションを備えたことで、単に既存のUnix/Linuxのコマンドを工夫して組みあわせること以上の機能を多数こなします。例えば、freq -y 2.. filenameのような形式のオプションをつけたコマンドライン入力をすれば、2回以上出現した文字列のみを対象に集計をします。クライアントとの打ち合わせ中でも、このコマンドを使って気軽にデータの重複の様子を見せることもできます(図4)。

【sampler】

各行に対して、入力した順に、指定された確率で無作為に抽出して出力します。オプションで、統計学で言う所の復元抽出または非復元抽出の方式選択も可能です。乱数シードも指定できるので、再現性も確保できます。1行ごとに異なる抽出確率を指定することもできます。

【csel】

特定の列を取り出すために、cutコマンドやAWK言語はよく利用されますが、それでもなおもっと簡単な仕組みが必要と考えて作ったコマンドです。付加するオプションとして、-pで取り出したい列の番号を指定し、-dで取り出さない列番号を指定できます。-hまたは-tでそれぞれ最も左側または右側に出力させたい列番号を指定します。列は複数でも範囲でも、さらに負の数で右からの位置も指定可能です。-=のオプションを使えば、入力データの1行目を列名の並びと解釈して、番号の代わりに列名で列指定が可能です。いくつものテーブルで同じような列名の値を参照するときに便利です。

【venn】

複数の表があれば、同じ意味と同じ符号体系を持つ列が異なる表に(外部キーとして)またがって存在していることは当然のように起こります。したがって、似たような値を持ついくつかの列の、値の包含関係を調べたいと思うことはよくあることです。あるはずのER図がないことも多いものです。このコマンドvennは集合の数が6個でも7個でも場合によってはそれ以上でも、集合間の要素の包含関係がわかるような表を出力します。一つ前に説明したcselを使い、venn <(csel -p1 tbl1)<(csel -p2 tbl2)<(csel -p3 tbl3)のようなプロセス置換を多用した使い方が多くなることでしょう。

再び2単語の組合せに戻ります。


```
~ % cols -p3,4 olympic.tsv | crosstable -m q | colorplus -s "0" -b blue
```

金*銀	0	2	1	3	4	5	6	7	8	9	10	14
0	0	2	1	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	1	0	1	0	0	0
4	0	0	0	1	0	0	0	1	0	0	1	0
5	0	0	0	0	0	0	0	0	1	0	0	0
6	0	0	0	0	1	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	1	0	0	0	1
9	0	0	0	0	0	0	2	0	0	0	0	0
10	0	0	0	0	0	0	0	0	1	0	0	0
11	0	0	0	0	0	0	0	1	0	0	0	0
12	0	0	0	0	0	0	0	0	1	0	0	0
13	0	0	0	0	0	0	0	0	1	0	0	0
16	0	0	0	0	0	1	0	0	0	1	0	0

```
~ %
```

図5 crosstableを使用した例（図中のcolsは以前のcsl）

【crosstable】

2列のデータに対し、1列目の値と2列目の値の組合せが何個出現したかをクロス集計表（分割表）として出力します。エクセルのピボットテーブルと似たような機能を持ちます（図5）。

直接テーブルデータを分析するものではないものの、上記のようなコマンドを使っている状況下で欲しくなるような機能として、次のようなコマンドも提供しています。

【latexable】

企業に蓄積されたデータの分析をすると多数の表形式のデータに接します。エクセルの仕様書、SQLの出力結果、そして、このTable::Hackの出力結果です。多数の表をばらばらにファイルなどで管理して、必要に応じて一つ一つ参照するよりは、一つの文書にまとめて、それを眺めて次の分析を考えた方が仕事は捗るというものです。どうしてなかなか、図表や画像を多数載せた文書は、今でもOffice系のソフトだと動作が不安定です。20個以上のテーブルを相手にしたいなら、結局、分析結果をLaTeX形式の表としてまとめていくのが良いのではないのでしょうか。そのためのLaTeXのコマンドをTSV形式のデータから生成するのが、このlatexableです。

【madeafter】

「先週の木曜日の会議の後の分析でいろいろファイルを作ったがどこに行ったっけ?」、「マネジメント上ファイル編成を考えなおしたい。同じチームのあの人の作ったいろいろなファイルの作成と利用状況を大雑把にうまく把握したい」などと思うこともあることでしょう。そういう場面

では、lsコマンドやfindコマンドでも、まだ足りないかもしれません。人間の目はばらばらの情報を扱うのも、多数の日時のような桁数の多いものの比較もあまり得意ではありません。このmadeafterのコマンドは、ファイル時刻情報の主要な3項目であるアクセス/更新/作成時刻からそれぞれ現在までの秒数を一度に表示します。分、時間、日に単位を変更することもできます。

4. むすび

データ分析と言いますと、ディープラーニングによる画像や音声認識やランダムフォレストによる分類などの高度な数理的な手法を連想する人も多いと思いますが、クロス集計の表をいくつか作れば用事は済むこともよくあります。ところが、その前の段階の、どういふ変数で集計をすれば良いか、入手したデータの意味の理解は現状難しく、どのような手順を経ればデータの理解が迅速にできてデータ分析環境をうまく構築できるのか、それ自体が未解明です。それが解決されると、社会に埋もれた商業的ないし科学的な情報の活用が非常に効率的になることでしょう。なお、今回紹介したソフトウェアを使っている人の話を伺いますと、「普段のデータ分析において、他のプログラミングなしに、いかにしてこのコマンドだけで用事を済ますかを考えるようになった」と聞いています。私としては、エクセルがない時代のエクセルを作ったような気でこのソフトを作ったので、嬉しい気持ちは半面あり、データを他の高度なソフトで使い易いようにするための、データの事前分析のソフトを作ったつもりなので、困るような気持ちも半面あります。人類のデータに対する理解はまだ始まったばかりだと思います。

（2018年3月28日受付）

【文 献】

- 1) P.H. Salus: "UNIXの1/4世紀", ASCII (2000)
- 2) T. Shimono: "A hacking toolset for big tabular files", in Proceedings of 2016 IEEE International Conference on Big Data, pp.2902-2910

しもの としゆき 下野 寿之 東京大学大学院情報理工学系研究科後期博士課程修了。チームラボ(株)、(株)ゴーガ、電話会社の研究所、(独)統計センターなどを経て、現在、(株)デジタルガレージ(DG Lab)に所属。