RadiusIntelligence Code Challenge

Dr. H. Pezeshki

February 3, 2018

Contents

1 Prompt

This note has been prepared on February 3, 2018 by Dr. H. Pezeshki in response to the Radius Intelligence coding challenge.

The `R` statistical programming environment [2] has been chosen, and the `knitr` package [3] has been used for document preparation.

2 Input data

The (expanded) input data is in `JSON` format. This was converted into a standard R `data.frame` using the R `jsonlite` package [1].

```
> require (jsonlite)
> theInput = fromJSON (txt = '../provided/datascience-cc-1-master/data_analysis.json')
> str (theInput)


'data.frame': 1000000 obs. of  10 variables:
 $ city            : chr  "SAN DIEGO" "ATLANTA" "NEOSHO" "CINCINNATI" ...
 $ name            : chr  "AMD CUSTOM" "Real Hope Real Estate Inc" "Jimmy Sexton Photography" "YOU'RE ART" ..
 $ zip             : chr  "92131" "30345" "64850" "45243" ...
 $ revenue         : chr  "$20 to 50 Million" "Less Than $500,000" "Less Than $500,000" "Less Than $500,000"
 $ time_in_business: chr  "10+ years" "10+ years" "10+ years" "10+ years" ...
 $ phone           : chr  "3123628000" NA "4046331779" "4174513798" ...
 $ state           : chr  "CA" "GA" "MO" "OH" ...
 $ address         : chr  "10085 SCRIPPS RANCH CT STE A" "2566 SHALLOWFORD RD NE STE 104 # 302" "212 E MAIN S
 $ headcount       : chr  "50 to 99" "1 to 4" "1 to 4" "1 to 4" ...
 $ category_code   : chr  "44420000" "31490000" "53120000" "54000000" ...


> saveRDS (file='theInput.rds', theInput)
```

3 Fill rates

The following code fragment generates the (raw) fill rates in a `data.frame` called `M`. The results are shown in Table 3.

```
> thefills = function (x) {
+   sum (is.na (x))
+ }
> tmp = apply (X=theInput, MARGIN = 2, FUN= thefills)
>
> Nobs = dim (theInput)[1]
> vecfillrates = round (100 * (Nobs - tmp)/Nobs, digits = 3)
>
> tmp = as.numeric (vecfillrates)
> M = as.data.frame (x=matrix (data=tmp, nrow = 1
+                       , byrow = TRUE
+                       , dimnames = list ('Raw fill', names (vecfillrates)))
+             , stringsAsFactors = FALSE)
```

	city	name	zip	revenue	time_in_business	phone	state	address	headcount	category_code
Raw fill	99.999	99.999	99.999	94.309	91.612	59.089	99.999	99.999	96.235	99.999

Table 3.1: Raw fill rate

4 True-valued fill rates

The following code fragment was used to calculate the true fill-rates. The results are shown in Table 4

```
> thoseMissing = function (x) {
+   ndx = grep (pattern = '(^\\s+$|^$|null|none|^0$)', x=x, ignore.case = TRUE)
+   x[ndx] = NA_character_
+   x
+ }
>
> hf = theInput
> for (j in 1:dim(hf)[2]) {
+   hf[,j] = thoseMissing (hf[,j])
+ }
>
>
> tmp = apply (X=hf, MARGIN = 2, FUN= thefills)
>
> Nobs = dim (hf)[1]
> vecTrueFillRates = round (100 * (Nobs - tmp)/Nobs, digits = 3)
> tmp = as.numeric (vecTrueFillRates)
> tmp = as.data.frame (x=matrix (data=tmp, nrow = 1
+                       , byrow = TRUE
```

```
+                       , dimnames = list ('True fill', names (vecTrueFillRates)))
+             , stringsAsFactors = FALSE)
>
> M = rbind (M, tmp)
> # print (M)
```

	city	name	zip	revenue	time_in_business	phone	state	address	headcount	category_code
Raw fill	99.999	99.999	99.999	94.309	91.612	59.089	99.999	99.999	96.235	99.999
True fill	99.990	99.984	99.989	94.300	91.605	59.080	99.990	99.989	96.227	99.991

Table 4.1: Raw and true fill rates

5 Cardinalities of individual columns

The following code fragment was used to calculate the cardinalities displayed in Table 5

```
> card = function (x) {
+   length (unique (na.omit (x)))
+ }
>
> cardinalities = apply (X=hf, MARGIN = 2, FUN = card)
> tmp = as.numeric (cardinalities)
> tmp = as.data.frame (x=matrix (data=tmp, nrow = 1
+                       , byrow = TRUE
+                       , dimnames = list ('Cardinalities', names (vecTrueFillRates)))
+             , stringsAsFactors = FALSE)
```

	city	name	zip	revenue	time_in_business	phone	state	address	headcount	category_code
Cardinalities	13714	890653	26391	11	5	575148	53	892103	9	1178

Table 5.1: cardinalities

6 Exploratory analysis

One clearly observes that the columns `revenue`, `time_in_business` and `headcount` are *ordered factors*, and a measure of quantitative analysis is possible if one uses *ranks* of the entries rather than factor level numbers. Thus, we first convert these fields from `character` to `ordered factor` as follows:

```
> hc_levels = c(
+   "1 to 4", "5 to 9", "10 to 19", "20 to 49", "50 to 99"
+   , "100 to 249", "250 to 499", "500 to 999", "Over 1,000"
+ )
> hf$headcount = factor (x=hf$headcount, levels = hc_levels, ordered = TRUE)
>
> revenue_levels = c (
+   "Less Than $500,000", "$500,000 to $1 Million", "$1 to 2.5 Million", "$2.5 to 5 Million"
+   , "$5 to 10 Million", "$10 to 20 Million", "$20 to 50 Million"
+   , "$50 to 100 Million", "$100 to 500 Million", "Over $500 Million", "Over $1 Billion"
+ )
> hf$revenue = factor (x = hf$revenue, levels = revenue_levels, ordered = TRUE)
>
> time_in_business_levels = c(
+   "Less than a year", "1-2 years", "3-5 years", "6-10 years", "10+ years"
+ )
> hf$time_in_business = factor (x = hf$time_in_business, levels = time_in_business_levels, ordered = TRUE)
> str (hf)


'data.frame': 1000000 obs. of  10 variables:
 $ city            : chr  "SAN DIEGO" "ATLANTA" "NEOSHO" "CINCINNATI" ...
 $ name            : chr  "AMD CUSTOM" "Real Hope Real Estate Inc" "Jimmy Sexton Photography" "YOU'RE ART" ..
 $ zip             : chr  "92131" "30345" "64850" "45243" ...
 $ revenue         : Ord.factor w/ 11 levels "Less Than $500,000"<..: 7 1 1 1 2 4 3 1 1 1 ...
 $ time_in_business: Ord.factor w/ 5 levels "Less than a year"<..: 5 5 5 5 5 4 5 5 5 NA ...
 $ phone           : chr  "3123628000" NA "4046331779" "4174513798" ...
 $ state           : chr  "CA" "GA" "MO" "OH" ...
 $ address         : chr  "10085 SCRIPPS RANCH CT STE A" "2566 SHALLOWFORD RD NE STE 104 # 302" "212 E MAIN S
 $ headcount       : Ord.factor w/ 9 levels "1 to 4"<"5 to 9"<..: 5 1 1 1 1 2 1 1 1 3 ...
 $ category_code   : chr  "44420000" "31490000" "53120000" "54000000" ...
```

Let us inquire if time-in-business, revenue and headcount are predictive of one another. We can quantify this using the *rank correlations* of the three columns. The following code fragment was used to generate the result in Figure 6.1. One observes that there is no observable correlation amongst these three covariates and any predictive analysis of a new feature of the business (besides the ones in the input data) should use all three of headcount, revenue and time-in-business.

```
> tmp = hf[, c('time_in_business', 'revenue', 'headcount')]
> tmp$time_in_business = as.numeric (tmp$time_in_business)
> tmp$revenue = as.numeric (tmp$revenue)
> tmp$headcount = as.numeric (tmp$headcount)
> theCorrelations = cor (tmp, use="pairwise.complete.obs", method = "spearman")
```

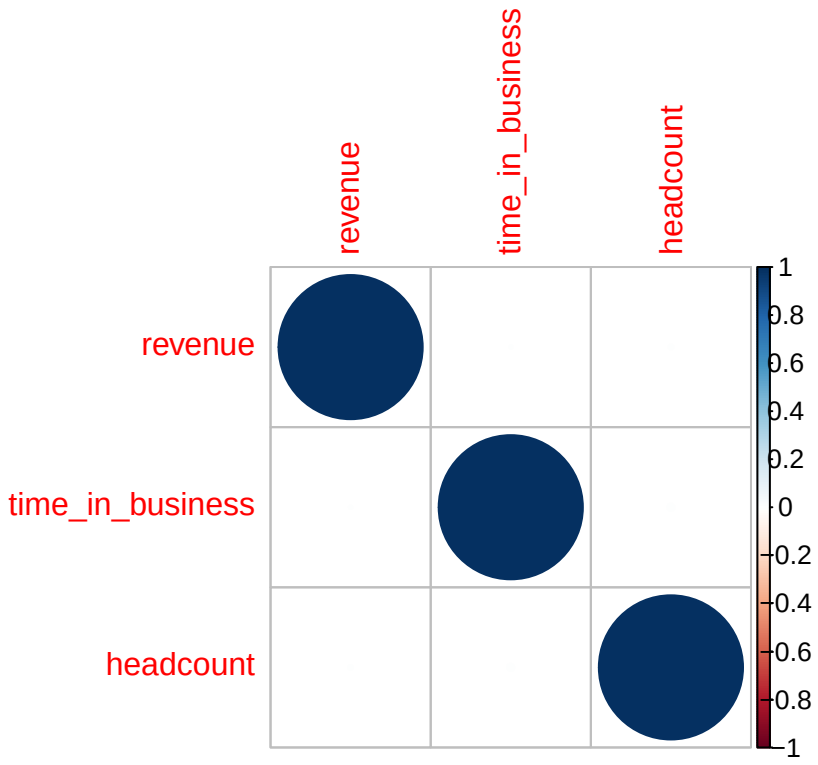

Figure 6.1: Rank correlations

References

[1] Jeroen Ooms. The jsonlite package: A practical and consistent mapping between json data and r objects. *arXiv:1403.2805 [stat.CO]*, 2014.

[2] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2016. `https://www.R-project.org`.

[3] Yihui Xie. *knitr: A General-Purpose Package for Dynamic Report Generation in R*, 2018. R package version 1.19.