Statistical Computing Portfolio 1

Rachel

2022-09-27

For this portfolio I will be using data obtained from the World Bank's DataBank on Gender Statistics [1]. Since there are approximately 1000 possible metrics and 265 countries to consider, I will only be focusing on the employment data

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

## corrplot 0.92 loaded

##
## Attaching package: 'sjmisc'

## The following object is masked from 'package:tidyr':
##
##     replace_na
```

#Reproducibility and Literate Programming

One of the most methodologically challenging aspect of data analysis is the difficulty in developing code that can easily be recreated by other researchers, or even the original researcher. This has typically been driven by two main factors:

- The code produced may not run correctly on computer systems other than the one it was developed on
- The research often includes many steps (finding data, reformatting, performing computiations and producing results and figures), where the approach of one step depends on the results of the previous one. The details of these components are also not recorded well enough to reproduce the data analysis.

[1] https://databank.worldbank.org/reports.aspx?source=gender-statistics#

While these were valid reasons historically, they no longer apply to current computer systems. Another potential issue is the data might not be public, but it would still be beneficial to share the program applied to synthetic data.

One way to remedy this is by using literate programming.

For the dataset I will be using, the data needs to be tidied before being analysed. The information will be read from the raw data file and saved to a new file after some processing. I will use R markdown to demonstrate the steps I take. First the data needs to be loaded as follows:

```
data <- read_csv("Raw_Gender_Data.csv", show_col_types = FALSE)
data
```

```
## # A tibble: 598 x 220
##    Time  Time ~1 Serie~2 Serie~3 Unite~4 Afgha~5 Alban~6 Alger~7 Ameri~8 Andor~9
##    <chr> <chr>   <chr>   <chr>   <chr>   <chr>   <chr>   <chr>   <chr>   <chr>
##  1 2020  YR2020  A woma~ SG.GET~ 1       1       1       1       ..      ..
##  2 2020  YR2020  A woma~ SG.NGT~ 1       0       1       0       ..      ..
##  3 2020  YR2020  A woma~ SG.DNG~ 1       0       1       0       ..      ..
##  4 2020  YR2020  A woma~ SG.IND~ 1       0       1       1       ..      ..
##  5 2020  YR2020  Access~ SH.HIV~ ..      9       53      87      ..      ..
##  6 2020  YR2020  Access~ SH.HIV~ ..      9       45      80      ..      ..
##  7 2020  YR2020  Adjust~ SE.PRM~ ..      ..      ..      ..      ..      ..
##  8 2020  YR2020  Adjust~ SE.PRM~ ..      ..      ..      ..      ..      ..
##  9 2020  YR2020  Adjust~ SE.PRM~ ..      ..      ..      ..      ..      ..
## 10 2020  YR2020  Adoles~ SP.ADO~ 11.183  57.509  19.4332 9.3594  ..      ..
## # ... with 588 more rows, 210 more variables: `Angola [AGO]` <chr>,
## #   `Antigua and Barbuda [ATG]` <chr>, `Argentina [ARG]` <chr>,
## #   `Armenia [ARM]` <chr>, `Aruba [ABW]` <chr>, `Australia [AUS]` <chr>,
## #   `Austria [AUT]` <chr>, `Azerbaijan [AZE]` <chr>,
## #   `Bahamas, The [BHS]` <chr>, `Bahrain [BHR]` <chr>,
## #   `Bangladesh [BGD]` <chr>, `Barbados [BRB]` <chr>, `Belarus [BLR]` <chr>,
## #   `Belgium [BEL]` <chr>, `Belize [BLZ]` <chr>, `Benin [BEN]` <chr>, ...
```

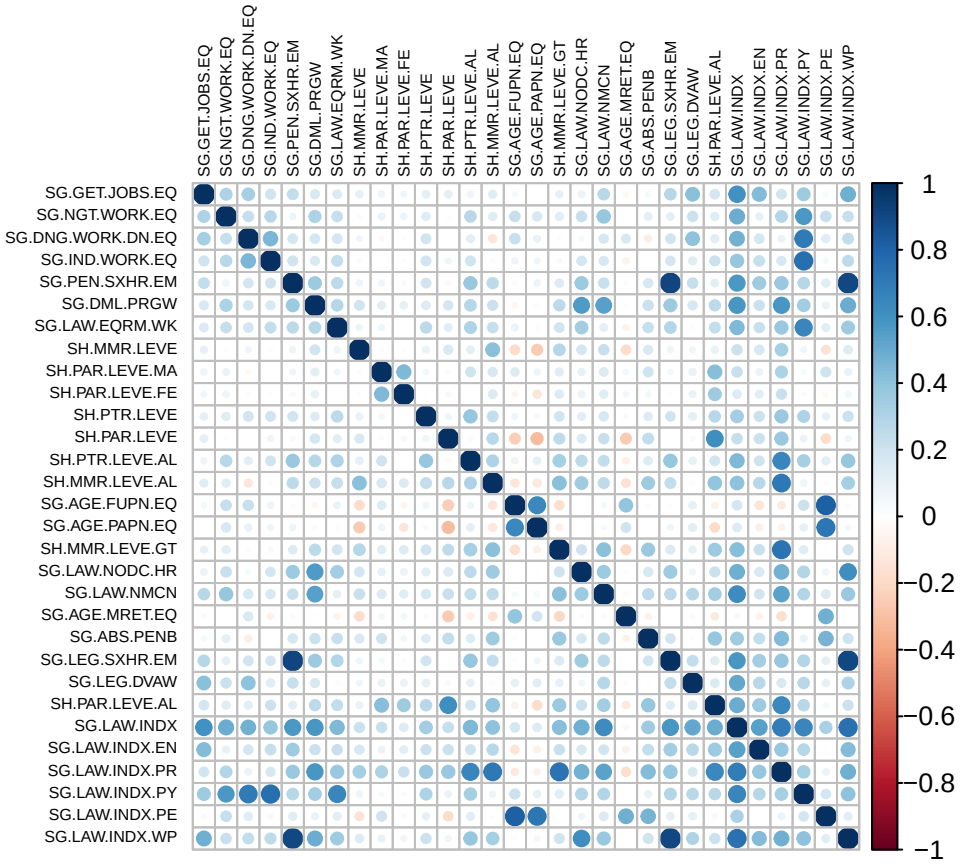
We can see on first glance that there is a lot of missing values from our data, so we want to tidy it so we are left with data containing no missing values. Before deciding which rows and colu

```
data <- data[,-c(1,2,3)]
data <- data %>% na_if("..")
rowSums(is.na(data))
```

```
##   [1]  28  28  28  28  89  89 216 216 216  23 127 215 216 216 216 216 216 216
##  [19] 216 216 216 216 216 216 216 216 216 216 216 216 216 216 216 216 216 216
##  [37] 216 216 216 216 216 216 216 216 216 216 216 216 216 216 216 216 216 216
##  [55] 216 216 216 216 216 216 155 156 175 175 216 216 215 216 216 216 209 210
##  [73]  28 216 216  28 179 179 179 179 179 179 186 186 186 184 184 184 185 185
##  [91] 185 179 179 179 181 181 181 184 184 184 216 216 216 216 216 216 216 216
## [109] 216 216 216 216  30 121  30 121  30 121  30 123  30 123  30 123  43  49
## [127]  49 183 183 183 212 213 213 213 213 213 216 216 216 216 216 216 216 216
## [145] 216 216 216 216 216 216 216 213 213 213 117 117 117  23  21  22  23 197
## [163]  33  30  33 181 216 216 216 216 216 188 188 187 216  43  58  58 212 212
## [181] 212  43  63  63  63  43  63  63  63  43  38  38  90  90  90  89  90  90
## [199]  57 216 216 216 216 216 214  30 123  30 123  30 123  30 119 216  30 119
## [217] 216  30 120 216 135 136 136 136 137 137 135 136 136  30  30  30  30  28
```

```
## [235] 216 216 216  43  63  63  28  28  28  28  28 216 216 201 201 201 201 201
## [253] 201 201 213 213 213 213 155 156 155 214 212 214 212 212 212 212 216 216
## [271] 216 216 216 216 215 215 213 213 216 216 216 216 216 216 216 216 216  50
## [289]  50  24  24  24  24  24  24 216 216 216  24  24  24 216 216  24  24  28
## [307]  28 147 145 145  32  61  47 216 216 216 214 214 197 191 213 213 216 216
## [325] 216  54  54  54  84  84 216 216 216 216 216 212 212 212 212 212 212 212
## [343] 212 212 216 212 212 212 212 212 212 153 153 153 216 142  43  43  43 216
## [361] 216 216 216 216 216 215 215 215 216 216 216  30 119  30 120 216 216 216
## [379] 216 142 145 145 143 216 143 160 143 216 143 216 159 216 159 159 216 159
## [397] 216 163 165 163 163 216 150 216 216 216 216 216 215 214 214 216 216 216
## [415] 216 216 216 216 138 138 138 216 216 216 216  43  43  43 216 172  28  28
## [433]  28  28  28  28  28  28  28  28 143 142 216 216 216 132 133 134 136 137
## [451] 137 132 133 133  30 118  30 118  30 118  30 120  30 120  30 120 211 216
## [469] 161 216 216 216 216 216 216 216 214 215 215 215 215 215 215 216 216  28
## [487] 214 214 214 214 214 214 214 214 214 214 214 214 214 214 214 214 214 214
## [505] 214 214 214 214 214 214 214 214 214 214 214 214 214 214 214 214 214 214
## [523] 216 216 216 216 216 216 216 216 216 216 216 216 216 216 216 216 216 216
## [541] 216 216 216 216 216 216 214 214 214 214 214 214 216 216 216 216 216 216
## [559] 215 214 213 213 213 213 213 213 213 213 213 213 213 213 213 213 209 208
## [577] 213 213 213 213 213 213 213  28  28  28  28  28  84 216 216 216 216 217
## [595] 217 217 217 217
```

```
data <- data[rowSums(is.na(data)) == 28, ]
data <- data[,colSums(is.na(data)) < nrow(data)]
data <- rotate_df(data, cn = TRUE)
data <- sapply(data, as.numeric)
cortable <- cor(data)
corrplot(cortable,tl.cex = 0.5, tl.col = "black")
```


SG.GET.JOBS.EQ
SG.NGT.WORK.EQ
SG.DNG.WORK.DN.EQ
SG.IND.WORK.EQ
SG.PEN.SXHR.EM
SG.DML.PRGW
SG.LAW.EQRM.WK
SH.MMR.LEVE
SH.PAR.LEVE.MA
SH.PAR.LEVE.FE
SH.PTR.LEVE
SH.PAR.LEVE
SH.PTR.LEVE.AL
SH.MMR.LEVE.AL
SG.AGE.FUPN.EQ
SG.AGE.PAPN.EQ
SH.MMR.LEVE.GT
SG.LAW.NODC.HR
SG.LAW.NMCN
SG.AGE.MRET.EQ
SG.ABS.PENB
SG.LEG.SXHR.EM
SG.LEG.DVAW
SH.PAR.LEVE.AL
SG.LAW.INDX
SG.LAW.INDX.EN
SG.LAW.INDX.PR
SG.LAW.INDX.PY
SG.LAW.INDX.PE
SG.LAW.INDX.WP
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1