assignment 1

Bhargav Rathod

2023-04-09

Echo is set to FALSE, alongside warning and message, for preventing markdown attach output errors.

Question 1

Data pre-processing.

```
set.seed(32151446) # XXXXXXXX = your student ID
cvbase = read.csv("Downloads/PsyCoronaBaselineExtract.csv")
cvbase <- cvbase[sample(nrow(cvbase), 40000), ] # 40000 rows
```

I used dim() function in order to check the dimentions of the data

```
dim(cvbase)

## [1] 40000    54
```

Based on the output from R, the study data set includes 54 columns and 40k row values (which we pre set for the study)

checking data types for each attribute using structure function and summary

```
str(cvbase)

## 'data.frame':    40000 obs. of  54 variables:
##  $ affAnx         : int  3 5 5 5 1 3 4 5 4 2 ...
##  $ affBor         : int  4 5 2 3 5 3 3 1 3 3 ...
##  $ affCalm        : int  3 1 2 1 5 2 3 1 2 2 ...
##  $ affContent     : int  4 1 2 1 4 2 3 1 3 3 ...
##  $ affDepr        : int  2 5 4 2 2 1 4 3 1 2 ...
##  $ affEnerg       : int  4 1 3 3 3 2 3 1 1 2 ...
##  $ affExc         : int  3 1 2 1 3 2 2 1 1 3 ...
##  $ affNerv        : int  3 5 5 5 2 2 4 5 3 3 ...
##  $ affExh         : int  3 2 4 2 3 2 4 5 3 2 ...
##  $ affInsp        : int  2 1 1 1 3 1 3 3 1 4 ...
##  $ affRel         : int  4 2 2 1 4 2 2 3 1 3 ...
##  $ PLRAC19        : int  4 4 5 4 3 3 6 4 3 2 ...
##  $ PLRAEco        : int  3 6 5 5 3 7 6 6 2 4 ...
##  $ disc01         : int  1 2 2 1 1 1 1 2 0 0 ...
##  $ disc02         : int  2 2 2 1 -1 1 1 2 0 0 ...
##  $ disc03         : int  0 -1 -1 -1 -1 0 0 -2 -1 2 ...
##  $ jbInsec01      : int  NA -2 -1 -1 -2 -2 0 -1 -2 -2 ...
```

```
##  $ jbInsec02      : int  NA 2 2 1 2 -2 0 0 2 2 ...
##  $ jbInsec03      : int  NA 2 -1 -1 -2 2 1 0 -1 2 ...
##  $ jbInsec04      : int  NA -2 NA -1 -2 NA 0 -1 -2 -2 ...
##  $ employstatus_1 : int  NA NA NA NA NA NA NA 1 NA NA ...
##  $ employstatus_2 : int  NA 1 NA NA NA NA NA NA NA NA ...
##  $ employstatus_3 : int  NA NA 1 1 1 NA 1 NA 1 1 ...
##  $ employstatus_4 : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ employstatus_5 : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ employstatus_6 : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ employstatus_7 : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ employstatus_8 : int  NA NA NA NA NA 1 NA NA NA NA ...
##  $ employstatus_9 : int  1 NA NA NA 1 NA NA NA NA NA ...
##  $ employstatus_10: int  NA NA NA NA NA NA NA NA NA NA ...
##  $ PFS01          : int  0 2 0 -1 -1 0 1 2 -1 -2 ...
##  $ PFS02          : int  0 2 0 -2 2 0 1 2 -1 -1 ...
##  $ PFS03          : int  0 2 -1 -2 -2 1 0 2 -1 -2 ...
##  $ fail01         : int  -1 0 1 -1 -1 1 0 2 -1 -2 ...
##  $ fail02         : int  1 0 -1 -1 -2 1 0 1 -1 -2 ...
##  $ fail03         : int  0 2 1 2 -2 0 1 0 -1 1 ...
##  $ happy          : int  7 3 7 6 8 7 6 9 7 9 ...
##  $ lifeSat        : int  3 2 5 5 4 3 3 5 5 4 ...
##  $ MLQ            : int  -1 -3 2 2 3 -1 0 1 1 1 ...
##  $ c19NormShould  : int  2 2 3 3 -1 1 1 3 2 3 ...
##  $ c19NormDo      : int  3 2 1 -2 -1 0 1 1 -2 3 ...
##  $ c19IsStrict    : int  5 3 2 3 5 6 2 4 3 6 ...
##  $ c19IsPunish    : int  3 2 1 2 4 5 2 1 2 6 ...
##  $ c19IsOrg       : int  3 3 2 4 4 4 2 3 3 6 ...
##  $ trustGovCtry   : int  NA 1 NA 3 2 4 3 NA NA 5 ...
##  $ trustGovState  : int  NA 1 NA 2 1 3 3 NA NA 5 ...
##  $ gender         : int  2 1 2 1 2 2 2 1 1 1 ...
##  $ age            : int  1 5 4 2 2 4 4 1 4 5 ...
##  $ edu            : int  2 3 7 7 6 2 6 6 7 1 ...
##  $ coded_country  : chr  "Switzerland" "Argentina" "United States of
America" "Netherlands" ...
##  $ c19ProSo01     : int  0 -2 2 1 3 1 1 3 0 3 ...
##  $ c19ProSo02     : int  -1 -3 2 2 3 0 1 3 1 3 ...
##  $ c19ProSo03     : int  0 -3 2 2 3 0 1 3 1 3 ...
##  $ c19ProSo04     : int  0 -3 2 3 3 0 1 3 1 3 ...
```

The output of str function defines data type of each column datum. Based on the output categorical variables include coded_country which character datatype. With all of the other columns being numerical int data type with employstatus columns being numeric num type.

```
summary(cvbase)

##      affAnx          affBor         affCalm        affContent
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :1.000
##  1st Qu.:2.000   1st Qu.:2.000   1st Qu.:2.000   1st Qu.:2.000
##  Median :3.000   Median :3.000   Median :3.000   Median :3.000
```

```
##  Mean   :2.723   Mean   :2.717   Mean   :2.927   Mean   :2.671
##  3rd Qu.:4.000   3rd Qu.:4.000   3rd Qu.:4.000   3rd Qu.:3.000
##  Max.   :5.000   Max.   :5.000   Max.   :5.000   Max.   :5.000
##  NA's   :512     NA's   :532     NA's   :527     NA's   :606
##     affDepr         affEnerg         affExc        affNerv          affExh
##  Min.   :1.000   Min.   :1.000   Min.   :1.00   Min.   :1.000   Min.
:1.000
##  1st Qu.:1.000   1st Qu.:2.000   1st Qu.:1.00   1st Qu.:2.000   1st
Qu.:1.000
##  Median :2.000   Median :3.000   Median :2.00   Median :2.000   Median
:2.000
##  Mean   :2.239   Mean   :2.576   Mean   :2.15   Mean   :2.587   Mean
:2.506
##  3rd Qu.:3.000   3rd Qu.:3.000   3rd Qu.:3.00   3rd Qu.:4.000   3rd
Qu.:3.000
##  Max.   :5.000   Max.   :5.000   Max.   :5.00   Max.   :5.000   Max.
:5.000
##  NA's   :603     NA's   :636     NA's   :678    NA's   :540     NA's
:632
##     affInsp          affRel         PLRAC19         PLRAEco
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :1.000
##  1st Qu.:1.000   1st Qu.:2.000   1st Qu.:3.000   1st Qu.:3.000
##  Median :2.000   Median :3.000   Median :4.000   Median :4.000
##  Mean   :2.435   Mean   :2.734   Mean   :3.556   Mean   :4.413
##  3rd Qu.:3.000   3rd Qu.:4.000   3rd Qu.:4.000   3rd Qu.:6.000
##  Max.   :5.000   Max.   :5.000   Max.   :8.000   Max.   :8.000
##  NA's   :665     NA's   :597     NA's   :152     NA's   :161
##      disc01            disc02            disc03          jbInsec01
##  Min.   :-2.0000   Min.   :-2.0000   Min.   :-2.0000   Min.   :-2.000
##  1st Qu.: 0.0000   1st Qu.: 0.0000   1st Qu.:-1.0000   1st Qu.:-2.000
##  Median : 1.0000   Median : 1.0000   Median : 0.0000   Median :-1.000
##  Mean   : 0.6375   Mean   : 0.8358   Mean   :-0.4025   Mean   :-0.594
##  3rd Qu.: 1.0000   3rd Qu.: 1.0000   3rd Qu.: 0.0000   3rd Qu.: 0.000
##  Max.   : 2.0000   Max.   : 2.0000   Max.   : 2.0000   Max.   : 2.000
##  NA's   :149       NA's   :144       NA's   :142       NA's   :10896
##    jbInsec02        jbInsec03        jbInsec04      employstatus_1
##  Min.   :-2.000   Min.   :-2.000   Min.   :-2.000   Min.   :1
##  1st Qu.: 0.000   1st Qu.:-1.000   1st Qu.:-2.000   1st Qu.:1
##  Median : 1.000   Median : 0.000   Median :-2.000   Median :1
##  Mean   : 0.555   Mean   : 0.058   Mean   :-0.985   Mean   :1
##  3rd Qu.: 1.000   3rd Qu.: 1.000   3rd Qu.: 0.000   3rd Qu.:1
##  Max.   : 2.000   Max.   : 2.000   Max.   : 2.000   Max.   :1
##  NA's   :9784     NA's   :8398     NA's   :12972    NA's   :34313
##  employstatus_2  employstatus_3  employstatus_4  employstatus_5
##  Min.   :1       Min.   :1       Min.   :1       Min.   :1
##  1st Qu.:1       1st Qu.:1       1st Qu.:1       1st Qu.:1
##  Median :1       Median :1       Median :1       Median :1
##  Mean   :1       Mean   :1       Mean   :1       Mean   :1
##  3rd Qu.:1       3rd Qu.:1       3rd Qu.:1       3rd Qu.:1
##  Max.   :1       Max.   :1       Max.   :1       Max.   :1
```

```
##  NA's   :33265   NA's   :29025   NA's   :36558   NA's   :37981
##  employstatus_6  employstatus_7  employstatus_8  employstatus_9
##  Min.   :1       Min.   :1       Min.   :1       Min.   :1
##  1st Qu.:1       1st Qu.:1       1st Qu.:1       1st Qu.:1
##  Median :1       Median :1       Median :1       Median :1
##  Mean   :1       Mean   :1       Mean   :1       Mean   :1
##  3rd Qu.:1       3rd Qu.:1       3rd Qu.:1       3rd Qu.:1
##  Max.   :1       Max.   :1       Max.   :1       Max.   :1
##  NA's   :36972   NA's   :36410   NA's   :39264   NA's   :31816
##  employstatus_10     PFS01              PFS02             PFS03
##  Min.   :1       Min.   :-2.00000   Min.   :-2.0000   Min.   :-2.0000
##  1st Qu.:1       1st Qu.:-1.00000   1st Qu.: 0.0000   1st Qu.:-1.0000
##  Median :1       Median : 0.00000   Median : 1.0000   Median : 0.0000
##  Mean   :1       Mean   :-0.02958   Mean   : 0.5724   Mean   :-0.2504
##  3rd Qu.:1       3rd Qu.: 1.00000   3rd Qu.: 1.0000   3rd Qu.: 1.0000
##  Max.   :1       Max.   : 2.00000   Max.   : 2.0000   Max.   : 2.0000
##  NA's   :39091   NA's   :175        NA's   :157       NA's   :153
##      fail01            fail02            fail03            happy
##  Min.   :-2.0000   Min.   :-2.0000   Min.   :-2.0000   Min.   : 1.00
##  1st Qu.:-1.0000   1st Qu.:-1.0000   1st Qu.: 0.0000   1st Qu.: 5.00
##  Median : 0.0000   Median :-1.0000   Median : 1.0000   Median : 7.00
##  Mean   :-0.0643   Mean   :-0.4134   Mean   : 0.3562   Mean   : 6.34
##  3rd Qu.: 1.0000   3rd Qu.: 0.0000   3rd Qu.: 1.0000   3rd Qu.: 8.00
##  Max.   : 2.0000   Max.   : 2.0000   Max.   : 2.0000   Max.   :10.00
##  NA's   :158       NA's   :161       NA's   :144       NA's   :544
##     lifeSat           MLQ          c19NormShould      c19NormDo
##  Min.   :1.000   Min.   :-3.0000   Min.   :-3.000   Min.   :-3.000
##  1st Qu.:3.000   1st Qu.: 0.0000   1st Qu.: 2.000   1st Qu.: 1.000
##  Median :4.000   Median : 1.0000   Median : 2.000   Median : 2.000
##  Mean   :4.145   Mean   : 0.8464   Mean   : 2.005   Mean   : 1.302
##  3rd Qu.:5.000   3rd Qu.: 2.0000   3rd Qu.: 3.000   3rd Qu.: 2.000
##  Max.   :6.000   Max.   : 3.0000   Max.   : 3.000   Max.   : 3.000
##  NA's   :138     NA's   :136       NA's   :153      NA's   :149
##   c19IsStrict     c19IsPunish       c19IsOrg      trustGovCtry
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :1.000
##  1st Qu.:3.000   1st Qu.:2.000   1st Qu.:3.000   1st Qu.:2.000
##  Median :4.000   Median :4.000   Median :4.000   Median :3.000
##  Mean   :4.122   Mean   :3.501   Mean   :3.897   Mean   :3.023
##  3rd Qu.:5.000   3rd Qu.:5.000   3rd Qu.:5.000   3rd Qu.:4.000
##  Max.   :6.000   Max.   :6.000   Max.   :6.000   Max.   :5.000
##  NA's   :181     NA's   :184     NA's   :172     NA's   :9359
##  trustGovState       gender           age             edu
##  Min.   :1.000   Min.   :1.000   Min.   :1.000   Min.   :1.000
##  1st Qu.:2.000   1st Qu.:1.000   1st Qu.:2.000   1st Qu.:4.000
##  Median :3.000   Median :1.000   Median :3.000   Median :5.000
##  Mean   :3.088   Mean   :1.388   Mean   :2.894   Mean   :4.407
##  3rd Qu.:4.000   3rd Qu.:2.000   3rd Qu.:4.000   3rd Qu.:5.000
##  Max.   :5.000   Max.   :3.000   Max.   :8.000   Max.   :7.000
##  NA's   :9454    NA's   :225     NA's   :253     NA's   :295
##  coded_country        c19ProSo01        c19ProSo02        c19ProSo03
```

```
##  Length:40000       Min.   :-3.0000   Min.   :-3.0000   Min.   :-3.0000
##  Class :character   1st Qu.: 0.0000   1st Qu.: 0.0000   1st Qu.: 0.0000
##  Mode  :character   Median : 1.0000   Median : 1.0000   Median : 1.0000
##                     Mean   : 0.9627   Mean   : 0.6715   Mean   : 0.5444
##                     3rd Qu.: 2.0000   3rd Qu.: 2.0000   3rd Qu.: 2.0000
##                     Max.   : 3.0000   Max.   : 3.0000   Max.   : 3.0000
##                     NA's   :141       NA's   :156       NA's   :154
##    c19ProSo04
##  Min.   :-3.00
##  1st Qu.: 0.00
##  Median : 2.00
##  Mean   : 1.29
##  3rd Qu.: 2.00
##  Max.   : 3.00
##  NA's   :161
```

finally summary function stating mean, median, and first and third quartile ranges of the numerical data.

Further, in order to clean the data, first step taken is to check upon the null values in the data set.

In the code below I used is.na function to check if a particular data oint being observed in the process contains null values and showing it's count using sum function to add up output values from is.na function.

Additionally, in order to check upon the number of null values in each column, I used apply function instead of loops for its better efficiency and wrote function to divide the sum of null values in a column by the length of it, in order to obtain percent values of null datapoints in a column

```
sum(is.na(cvbase))

## [1] 426824

#percent missing values per variable
apply(cvbase, 2, function(col)sum(is.na(col))/length(col))

##          affAnx          affBor         affCalm      affContent
affDepr
##        0.012800        0.013300        0.013175        0.015150
0.015075
##        affEnerg          affExc         affNerv          affExh
affInsp
##        0.015900        0.016950        0.013500        0.015800
0.016625
##          affRel         PLRAC19         PLRAEco          disc01
disc02
##        0.014925        0.003800        0.004025        0.003725
0.003600
##          disc03       jbInsec01       jbInsec02       jbInsec03
```

```
jbInsec04
##        0.003550        0.272400        0.244600        0.209950
0.324300
##  employstatus_1  employstatus_2  employstatus_3  employstatus_4
employstatus_5
##        0.857825        0.831625        0.725625        0.913950
0.949525
##  employstatus_6  employstatus_7  employstatus_8  employstatus_9
employstatus_10
##        0.924300        0.910250        0.981600        0.795400
0.977275
##           PFS01           PFS02           PFS03          fail01
fail02
##        0.004375        0.003925        0.003825        0.003950
0.004025
##          fail03           happy         lifeSat             MLQ
c19NormShould
##        0.003600        0.013600        0.003450        0.003400
0.003825
##       c19NormDo     c19IsStrict     c19IsPunish        c19IsOrg
trustGovCtry
##        0.003725        0.004525        0.004600        0.004300
0.233975
##   trustGovState          gender             age             edu
coded_country
##        0.236350        0.005625        0.006325        0.007375
0.000000
##      c19ProSo01      c19ProSo02      c19ProSo03      c19ProSo04
##        0.003525        0.003900        0.003850        0.004025
```

Based on the result employstatus 1 to 10 columns seems of contain too many NaN values. Apart from that, the rest of the columns contains fewer than 30% null values.

Studying employstatus data column it is defined by a label and no response possibilities, thus resulting in single output being 1 for applicable cases and the rest of the data points gets null values, the solution to which can be addition of a binary response being either 1 for acceptance or 0 for rejection. Thereby, replacing null values in the employ status 1 to 10 columns by 0 would eliminate all the null values without adding cost

converting the NA in employer status to 0, using is.na function in order to select the specific values in the column being null and replacing it with 0

```
cvbase$employstatus_1[is.na(cvbase$employstatus_1)] = 0
cvbase$employstatus_2[is.na(cvbase$employstatus_2)] = 0
cvbase$employstatus_3[is.na(cvbase$employstatus_3)] = 0
cvbase$employstatus_4[is.na(cvbase$employstatus_4)] = 0
cvbase$employstatus_5[is.na(cvbase$employstatus_5)] = 0
cvbase$employstatus_6[is.na(cvbase$employstatus_6)] = 0
cvbase$employstatus_7[is.na(cvbase$employstatus_7)] = 0
cvbase$employstatus_8[is.na(cvbase$employstatus_8)] = 0
```

```r
cvbase$employstatus_9[is.na(cvbase$employstatus_9)] = 0
cvbase$employstatus_10[is.na(cvbase$employstatus_10)] = 0
```

Further checking if the changes have been correctly made using previously stated approach to count the percent of null values in each column.

```r
sum(is.na(cvbase))

## [1] 72129

#percent missing values per variable
apply(cvbase, 2, function(col)sum(is.na(col))/length(col))

##          affAnx          affBor         affCalm      affContent
affDepr
##        0.012800        0.013300        0.013175        0.015150
0.015075
##        affEnerg          affExc         affNerv          affExh
affInsp
##        0.015900        0.016950        0.013500        0.015800
0.016625
##          affRel         PLRAC19         PLRAEco          disc01
disc02
##        0.014925        0.003800        0.004025        0.003725
0.003600
##          disc03       jbInsec01       jbInsec02       jbInsec03
jbInsec04
##        0.003550        0.272400        0.244600        0.209950
0.324300
##  employstatus_1  employstatus_2  employstatus_3  employstatus_4
employstatus_5
##        0.000000        0.000000        0.000000        0.000000
0.000000
##  employstatus_6  employstatus_7  employstatus_8  employstatus_9
employstatus_10
##        0.000000        0.000000        0.000000        0.000000
0.000000
##           PFS01           PFS02           PFS03          fail01
fail02
##        0.004375        0.003925        0.003825        0.003950
0.004025
##          fail03           happy         lifeSat             MLQ
c19NormShould
##        0.003600        0.013600        0.003450        0.003400
0.003825
##       c19NormDo     c19IsStrict     c19IsPunish        c19IsOrg
trustGovCtry
##        0.003725        0.004525        0.004600        0.004300
0.233975
##   trustGovState          gender             age             edu
coded_country
```

```
##        0.236350        0.005625        0.006325        0.007375
0.000000
##      c19ProSo01      c19ProSo02      c19ProSo03      c19ProSo04
##        0.003525        0.003900        0.003850        0.004025
```

Based on the result above it states complete removal of null values from employstatus 1 to 10 columns.

Dropping na values for the remaining data using na.omit function over cvbase, alongside dim for cheking the dimentions of the dataset after removal of na values.

```
cvbase = na.omit(cvbase)

dim(cvbase)

## [1] 18903    54
```

The data is reduced to 18,903 row values, which could potentially be a result of rows being eliminated for having fewer or even single null values in them.

```
colnames(cvbase)

##  [1] "affAnx"          "affBor"          "affCalm"         "affContent"
##  [5] "affDepr"         "affEnerg"        "affExc"          "affNerv"
##  [9] "affExh"          "affInsp"         "affRel"          "PLRAC19"
## [13] "PLRAEco"         "disc01"          "disc02"          "disc03"
## [17] "jbInsec01"       "jbInsec02"       "jbInsec03"       "jbInsec04"
## [21] "employstatus_1"  "employstatus_2"  "employstatus_3"
"employstatus_4"
## [25] "employstatus_5"  "employstatus_6"  "employstatus_7"
"employstatus_8"
## [29] "employstatus_9"  "employstatus_10" "PFS01"           "PFS02"
## [33] "PFS03"           "fail01"          "fail02"          "fail03"
## [37] "happy"           "lifeSat"         "MLQ"             "c19NormShould"
## [41] "c19NormDo"       "c19IsStrict"     "c19IsPunish"     "c19IsOrg"
## [45] "trustGovCtry"    "trustGovState"   "gender"          "age"
## [49] "edu"             "coded_country"   "c19ProSo01"      "c19ProSo02"
## [53] "c19ProSo03"      "c19ProSo04"
```

data visualization

The code blow below is designed to show the top 10 country with pro social covid behavior (c19ProSo01) stating the number of People willing to help others who suffer from coronavirus through se of pie chart.

Pie chart was selected as other than bar chart and box plot it is the only plot suitable for depicting categorical variable split comparing over a numerical variable. Box plot was avoided for larger categorization value being 10, which can cause clustering and compression of box plots.

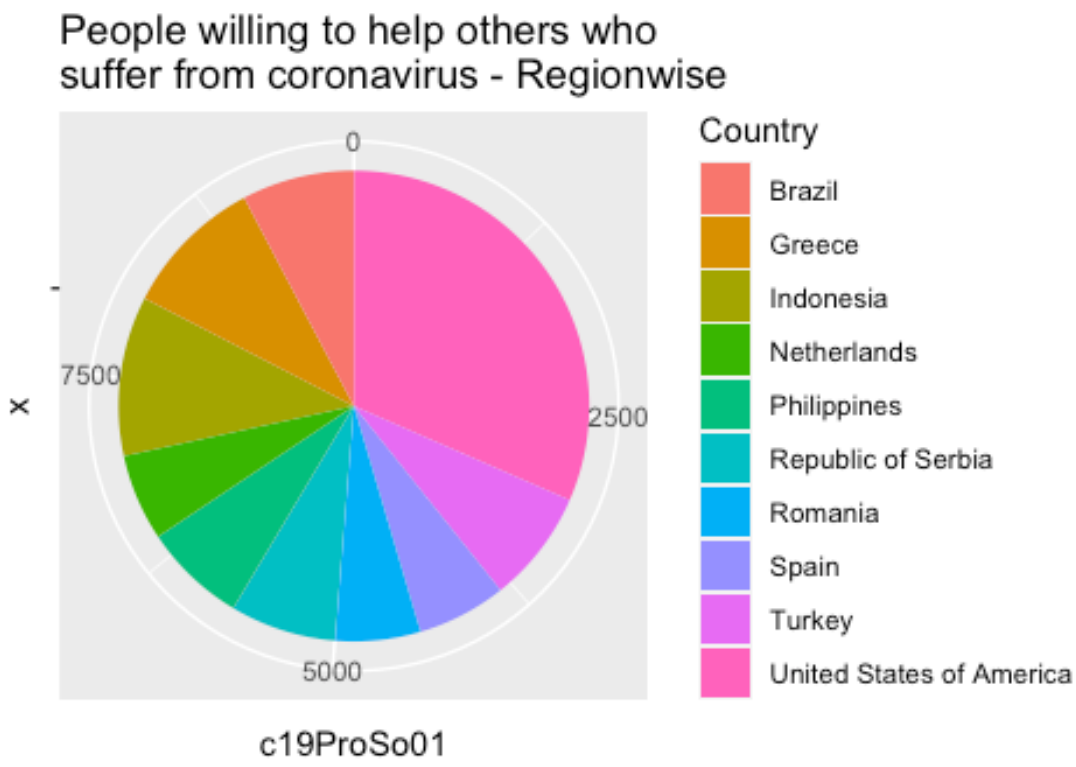
first step is to prepare dataset for the plot containing the entires grouped by country name and the rest of the column values being summed up. Sum operation was used as the median and mean won't be sortable.

```
# selecting coded_country, and all of the pro social covid attribute based
columns
temp_data_ = cvbase[, c(50:54)]
# aggregrate performed to group the data based on country name, and summing
up the values of c19ProSo01, 02, 03 and 04
reg_15 <- aggregate(temp_data_[, c(-1)], list(temp_data_$coded_country), sum)
# renaming the index column to country to avoid changing names
names(reg_15)[1] <- "Country"
# finally head in order to check the output briefly
head(reg_15)
```

```
##      Country c19ProSo01 c19ProSo02 c19ProSo03 c19ProSo04
## 1    Albania         -1          3          1          1
## 2    Algeria         47         48         33         46
## 3  Argentina        361        238        232        577
## 4  Australia        281        117        181        556
## 5    Austria         28         10         27         37
## 6 Azerbaijan          1          2          0          0
```

Furhter using attach function to attach the dataset to the code environment, alongside use of ggplot funcition to plot the pie chart

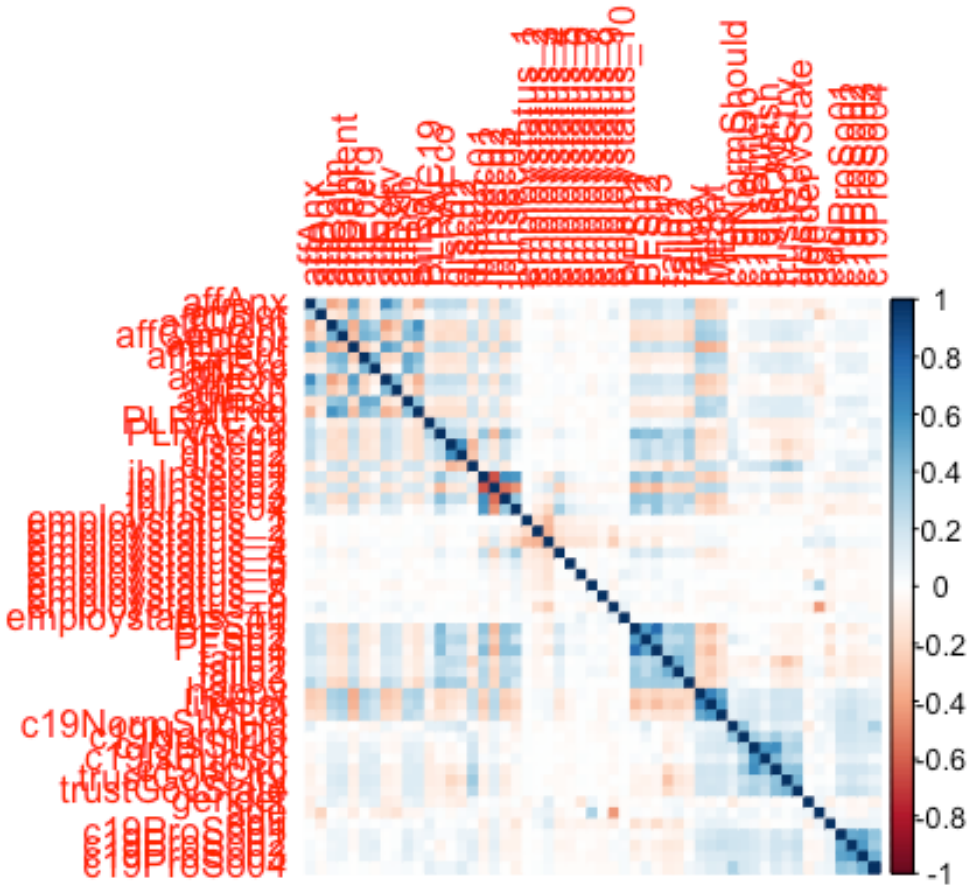
```
attach(reg_15)
library(ggplot2)
# arranging it descending order in order to get the highest values
reg_15 = reg_15[order(-reg_15$c19ProSo01),]
# filtering data to just 10 values for better visualization
reg_15 <- reg_15[1:10,]
# suing geom_bar function to plot pie chart
bp<- ggplot(reg_15, aes(x="", y=c19ProSo01, fill=Country))+
    geom_bar(width = 1, stat = "identity")
# adding title to the plot
bp + coord_polar("y")+labs(title="People willing to help others who
suffer from coronavirus - Regionwise")
```



Based on the results above it can be seen that USA lead the rest of the countries in the dataset with quite significant majority over pro socila covid behaviour.

Further corelation between numerical values is observed through heat map. As clustered facets can't provide understandable output for it's complexity involved in representation of single correlation. Thus, heat map was used in order to achieve basic understanding of the correlation using corrplot package.

```
library(corrplot)

## corrplot 0.92 loaded

# to select numeric columns
Num.cols <- sapply(cvbase, is.numeric)
# calculating correlation
Cor.data <- cor(cvbase[, Num.cols])
# for plotting the graph
corrplot(Cor.data, method = 'color')
```



Based on the graph above it can be seen that the correlation between column variables has been mostly less significant for having too fewer and less darker red values in the heat map indicating lower corrlation between most variables.

question 2a

Instead of making a new data frame I added a new column to the cvbase dataset in order to divide the countries among 2 clusters being either my focus country (Pakistan) or the rest of the world naming the new column as grouped_countries.

```
# introducing new column grouped_countries pre set to chr "other countries"
as it contributes majority of the dataset
cvbase$grouped_countries = "other countries"
# modifying the column. value in grouped_countries for appearance of Pakistan
in the coded_country column.
cvbase$grouped_countries[cvbase$coded_country == "Pakistan"] = "Pakistan"
# unique is used to test the final data obtained
unique(cvbase$grouped_countries)

## [1] "other countries" "Pakistan"
```

The output shows 2 different unique values in grouped_counries, which indicates it's correctness.

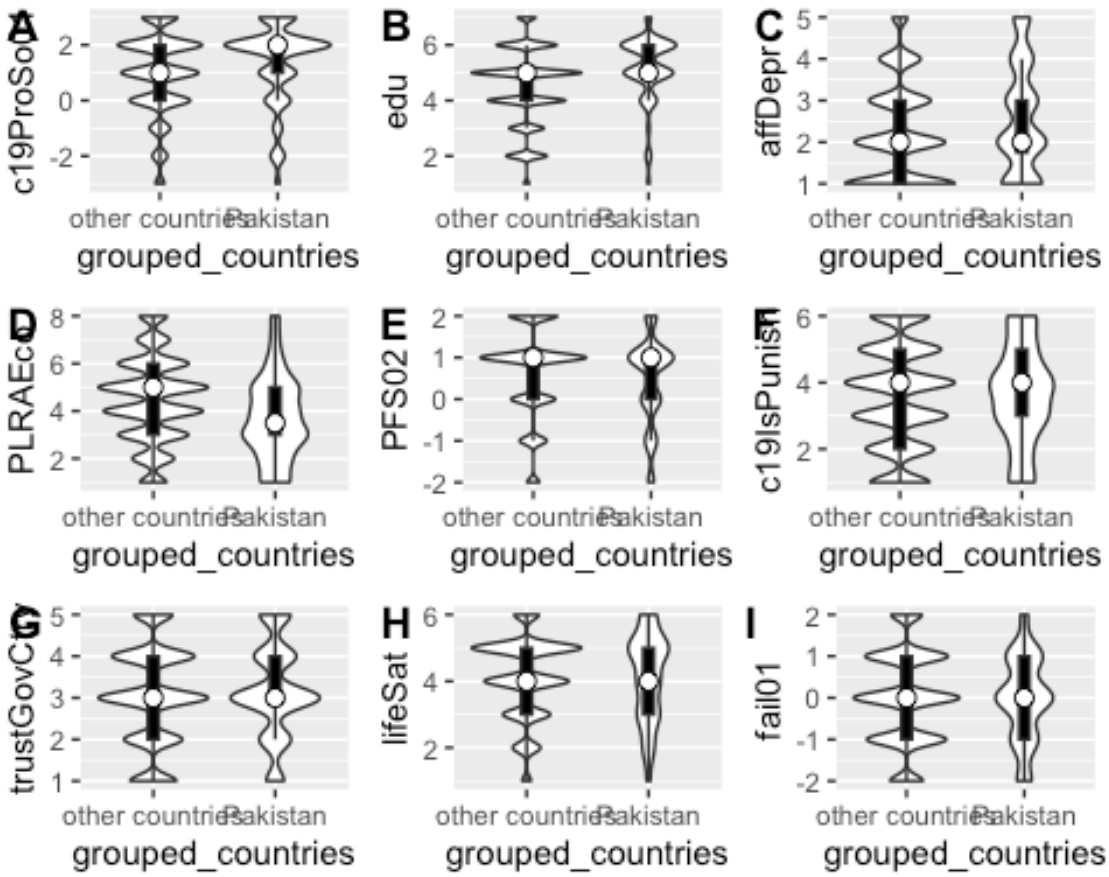
Further for q2 for comparing the characteristics of 2 categorical variables over rest of numerical variables I plotted a violin plot for it representing density plots which describes data better compared to histogram, which can't be performed on the dataset as the number of datapoints for Pakistan are way less compared to the rest of the world. Thus it won't properly visualize the data. Furhte rthis method reduces 2 plots which histogram might require to single plot.

plotting violin plot for the cvbase dataset, with use of ggplot to produce violin plot adn box plot in it, alongside cowplot for combining ggplot outputs into single chart. For testing participant responses, I compared the 2 groups performance to one column from each other category described in the dataset using cowplot to combine graphs and geomviolin alongside geom_boxplot

violin graph

```
library(ggplot2)
library(cowplot)

aa <- ggplot(cvbase, aes(x = grouped_countries, y = c19ProSo01)) +
geom_violin() +
    geom_boxplot(width = .09, fill = "black", outlier.colour = NA) +
  stat_summary(fun.y = median, geom = "point", fill = "white", shape = 21,
size = 2.5)

## Warning: The `fun.y` argument of `stat_summary()` is deprecated as of
ggplot2 3.3.0.
## ℹ Please use the `fun` argument instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

bb <- ggplot(cvbase, aes(x = grouped_countries, y = edu)) + geom_violin() +
    geom_boxplot(width = .09, fill = "black", outlier.colour = NA) +
  stat_summary(fun.y = median, geom = "point", fill = "white", shape = 21,
size = 2.5)

cc <- ggplot(cvbase, aes(x = grouped_countries, y = affDepr)) + geom_violin()
+
    geom_boxplot(width = .09, fill = "black", outlier.colour = NA) +
  stat_summary(fun.y = median, geom = "point", fill = "white", shape = 21,
size = 2.5)

dd <- ggplot(cvbase, aes(x = grouped_countries, y = PLRAEco)) + geom_violin()
+
    geom_boxplot(width = .09, fill = "black", outlier.colour = NA) +
  stat_summary(fun.y = median, geom = "point", fill = "white", shape = 21,
```

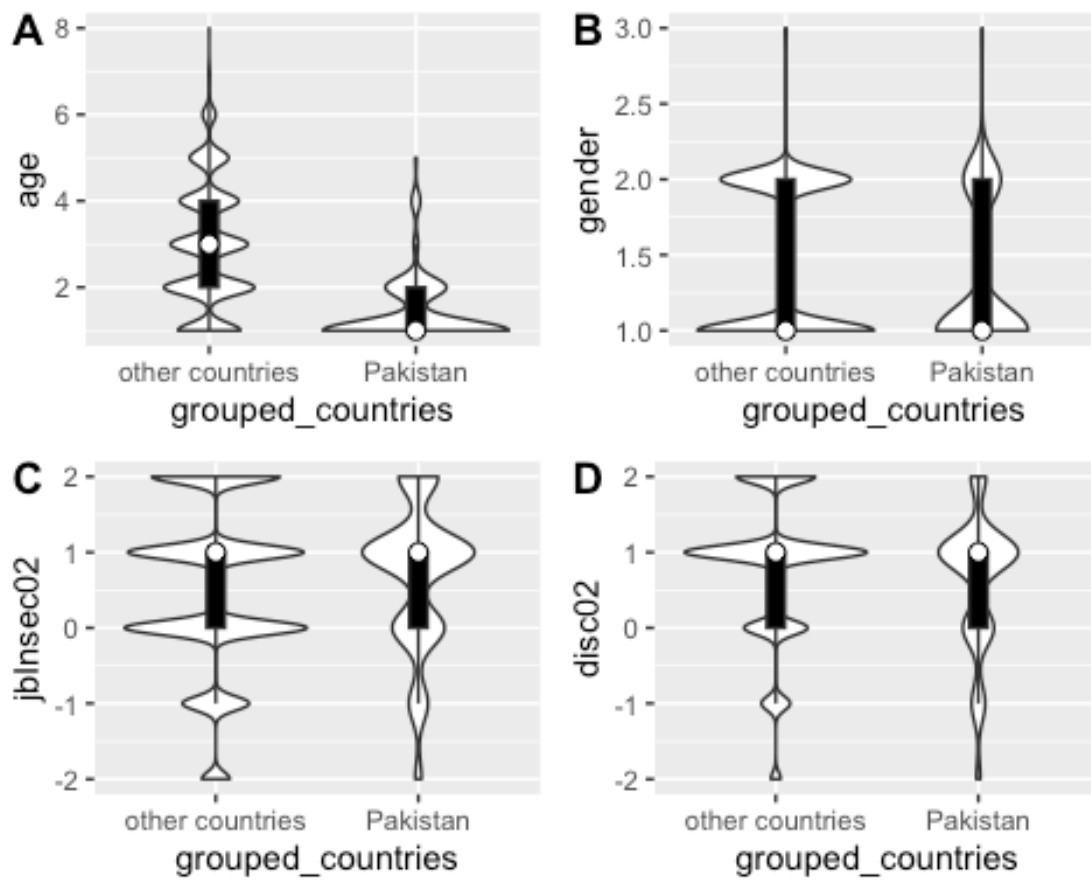
```
size = 2.5)


ee <- ggplot(cvbase, aes(x = grouped_countries, y = PFS02)) + geom_violin() +
    geom_boxplot(width = .09, fill = "black", outlier.colour = NA) +
  stat_summary(fun.y = median, geom = "point", fill = "white", shape = 21,
size = 2.5)


ff <- ggplot(cvbase, aes(x = grouped_countries, y = c19IsPunish)) +
geom_violin() +
    geom_boxplot(width = .09, fill = "black", outlier.colour = NA) +
  stat_summary(fun.y = median, geom = "point", fill = "white", shape = 21,
size = 2.5)


gg <- ggplot(cvbase, aes(x = grouped_countries, y = trustGovCtry)) +
geom_violin() +
    geom_boxplot(width = .09, fill = "black", outlier.colour = NA) +
  stat_summary(fun.y = median, geom = "point", fill = "white", shape = 21,
size = 2.5)

hh <- ggplot(cvbase, aes(x = grouped_countries, y = lifeSat)) + geom_violin()
+
    geom_boxplot(width = .09, fill = "black", outlier.colour = NA) +
  stat_summary(fun.y = median, geom = "point", fill = "white", shape = 21,
size = 2.5)

ii <- ggplot(cvbase, aes(x = grouped_countries, y = fail01)) + geom_violin()
+
    geom_boxplot(width = .09, fill = "black", outlier.colour = NA) +
  stat_summary(fun.y = median, geom = "point", fill = "white", shape = 21,
size = 2.5)


plot_grid(aa, bb, cc, dd, ee, ff, gg, hh, ii, labels = "AUTO")
```



The above graph could not cluster more than 9 graphs clearly, thus of combination graphs each representing 9 clusters is done. Based on the graph above it can be observed that participants for pakistan had better score over c19ProSo01 test indicating more willingness to help the ones suffering in covid. Furhter, educational background among the rest of the world seems evenly divided however Pakistan data shows it being having more participants from people having Masters and PhDs. The median for affDepr suggested nearly same results for both the groups. Further PLRAEco depicted the respondents from Pakistan responsed to lower liklihood of covid happening to them compared to rest of the world. Further having nearly similar trends in responses to trust in goverment alongside lifestats and disempowerment.

```
library(ggplot2)
library(cowplot)

jj <- ggplot(cvbase, aes(x = grouped_countries, y = age)) + geom_violin() +
    geom_boxplot(width = .09, fill = "black", outlier.colour = NA) +
  stat_summary(fun.y = median, geom = "point", fill = "white", shape = 21,
size = 2.5)

kk <- ggplot(cvbase, aes(x = grouped_countries, y = gender)) + geom_violin()
+
    geom_boxplot(width = .09, fill = "black", outlier.colour = NA) +
  stat_summary(fun.y = median, geom = "point", fill = "white", shape = 21,
```

```
size = 2.5)

ll <- ggplot(cvbase, aes(x = grouped_countries, y = jbInsec02)) +
geom_violin() +
    geom_boxplot(width = .09, fill = "black", outlier.colour = NA) +
  stat_summary(fun.y = median, geom = "point", fill = "white", shape = 21,
size = 2.5)

mm <- ggplot(cvbase, aes(x = grouped_countries, y = disc02)) + geom_violin()
+
    geom_boxplot(width = .09, fill = "black", outlier.colour = NA) +
  stat_summary(fun.y = median, geom = "point", fill = "white", shape = 21,
size = 2.5)

plot_grid(jj, kk, ll, mm , labels = "AUTO")
```



The data graph above furhter indicated more male articipants in the dataset compared to females for both the groups, with Pakistan having younger participats compared to relatively higher age groups for the rest of the world. Somehow both the groups showed higher disontent about the future of the socety. Lastly, it can also be observed that both the groups depicted lower job insecurity, for having higher agreement over security of the job through jbInsecurity02.

question 2b

To predict contribution of other factors towards pro-social attitudes (c19ProSo01,2,3 and 4) for the focus country I created a separate sub dataset called filter, containing entries having Pakistan in their coded_country.

```
# to filter entries/rows with Pakistan as it's country name
filter = cvbase[cvbase$coded_country == "Pakistan",]
# checking the dimentions of the data set in order to ensure correctness
dim(filter)

## [1] 180  55

colnames(filter)

##  [1] "affAnx"            "affBor"            "affCalm"
##  [4] "affContent"        "affDepr"           "affEnerg"
##  [7] "affExc"            "affNerv"           "affExh"
## [10] "affInsp"           "affRel"            "PLRAC19"
## [13] "PLRAEco"           "disc01"            "disc02"
## [16] "disc03"            "jbInsec01"         "jbInsec02"
## [19] "jbInsec03"         "jbInsec04"         "employstatus_1"
## [22] "employstatus_2"    "employstatus_3"    "employstatus_4"
## [25] "employstatus_5"    "employstatus_6"    "employstatus_7"
## [28] "employstatus_8"    "employstatus_9"    "employstatus_10"
## [31] "PFS01"             "PFS02"             "PFS03"
## [34] "fail01"            "fail02"            "fail03"
## [37] "happy"             "lifeSat"           "MLQ"
## [40] "c19NormShould"     "c19NormDo"         "c19IsStrict"
## [43] "c19IsPunish"       "c19IsOrg"          "trustGovCtry"
## [46] "trustGovState"     "gender"            "age"
## [49] "edu"               "coded_country"     "c19ProSo01"
## [52] "c19ProSo02"        "c19ProSo03"        "c19ProSo04"
## [55] "grouped_countries"
```

Further, as the values in the dataset are categorical, I cannot use linear regression over it. Thus for comparing relationship between categorical variables I used chi square error methodology through chisq.test function. For the same I needed to remove the entries with 0 as value in it as chi square methodology produces error for 0 as values of datum. for it dividing the difference of predicted and original by original and 0/0 produces error

further I used lapply in order to apply the chisq.test function to every column of dataset, lastly using rbind to only display the relevant columns.

Lastly I analysed the data just for single Pro Social covid attitude column group, as all the coulmns are representative of the same ideology and tone with same categorical response values alongside stronger correlation between each of them.

```
# replacing 0 with 10
filter[filter== 0] <- 10
```

```
#applying chi square test algorithm to every column of dataset
hey = lapply(filter[,-c(55, 50)], function(x) chisq.test(filter[,51], x));
```

```
## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
```

```
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
```

```
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
```

```
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect

## Warning in chisq.test(filter[, 51], x): Chi-squared approximation may be
## incorrect
```

```
do.call(rbind, hey)[,c(1,3)]
```

```
##                 statistic p.value
## affAnx          13.50743  0.9570049
## affBor          22.17854  0.5686113
## affCalm         17.12439  0.8433083
## affContent      25.75022  0.3659593
## affDepr         21.31251  0.6202276
## affEnerg        34.98005  0.06869713
## affExc          26.65122  0.3209713
## affNerv         25.15897  0.3971634
## affExh          29.14916  0.2145677
## affInsp         17.42317  0.8300615
## affRel          32.36113  0.1182872
## PLRAC19         39.88108  0.5643747
## PLRAEco         46.47     0.2933535
## disc01          35.66585  0.05911961
## disc02          39.8076   0.02242645
## disc03          26.82627  0.312616
## jbInsec01       34.08384  0.08319006
## jbInsec02       45.93883  0.004502537
## jbInsec03       25.59785  0.3738802
## jbInsec04       34.19543  0.08125523
## employstatus_1  4.954954  0.5496035
## employstatus_2  16.74854  0.01025322
## employstatus_3  8.629433  0.1955159
## employstatus_4  8.410967  0.2095138
## employstatus_5  8.572523  0.1990844
## employstatus_6  11.21337  0.08200161
## employstatus_7  10.55578  0.1031148
## employstatus_8  7.093016  0.3123308
## employstatus_9  15.58015  0.01619395
## employstatus_10 13.82987  0.03159541
```

```
## PFS01           33.28847  0.09813155
## PFS02           33.67717  0.09057211
## PFS03           55.35432  0.0002803914
## fail01          17.30148  0.8355222
## fail02          18.42589  0.7818737
## fail03          28.94824  0.2220848
## happy           65.88742  0.1287328
## lifeSat         34.39851  0.2651822
## MLQ             57.17197  0.01384999
## c19NormShould   65.13559  0.002082977
## c19NormDo       61.6148   0.004959841
## c19IsStrict     50.1176   0.01205758
## c19IsPunish     30.17754  0.4565878
## c19IsOrg        40.68397  0.09228215
## trustGovCtry    43.20371  0.009427663
## trustGovState   35.89785  0.05615233
## gender          15.22813  0.2291971
## age             37.67418  0.03742939
## edu             73.7079   0.0002120947
## c19ProSo01      1080      2.480107e-203
## c19ProSo02      180.2022  2.331936e-23
## c19ProSo03      202.2224  4.982199e-25
## c19ProSo04      147.6154  1.836059e-15
```

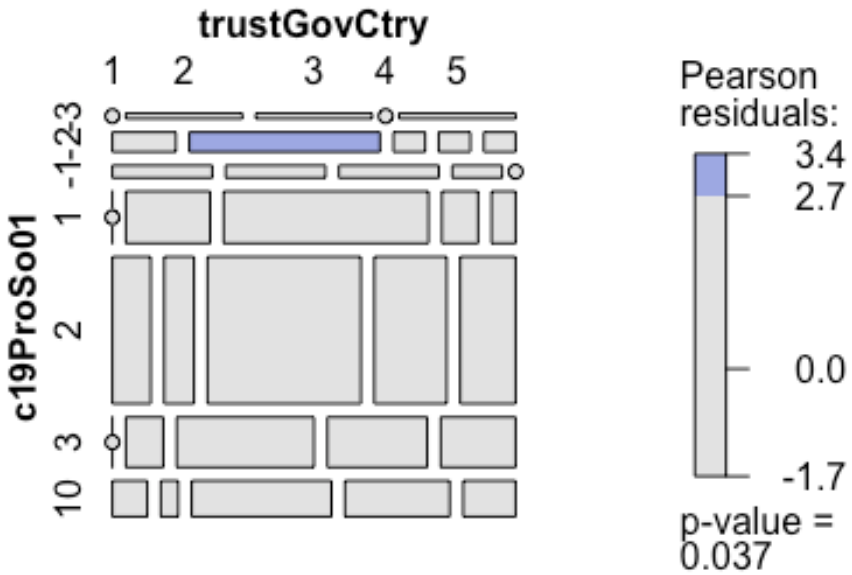
Based on the output it seems like the might be some internal working error in the chisq.test function which might be producing error lines, as 0 values are removed from the dataset. Based on the output the predictors with lower p values signifies the factors of most significance proving lower probability (p value) against null hypothesis.

Thus based on the the following columns seems related to pro social covid attitude in dataset for participants from Pakistan:

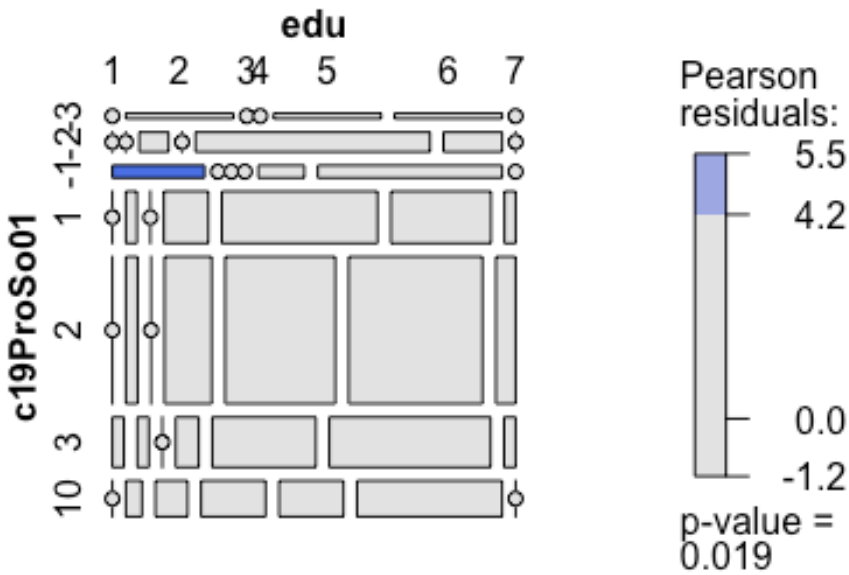
trustGovCtry, edu, c19NormShould, jbInsec02 and PFS03

Further I tried visualizing the correlation between the categorical variables plotting the mosaic graph. Which accurately depicts the relation between the countries. Using the mosaic function of the vcd library

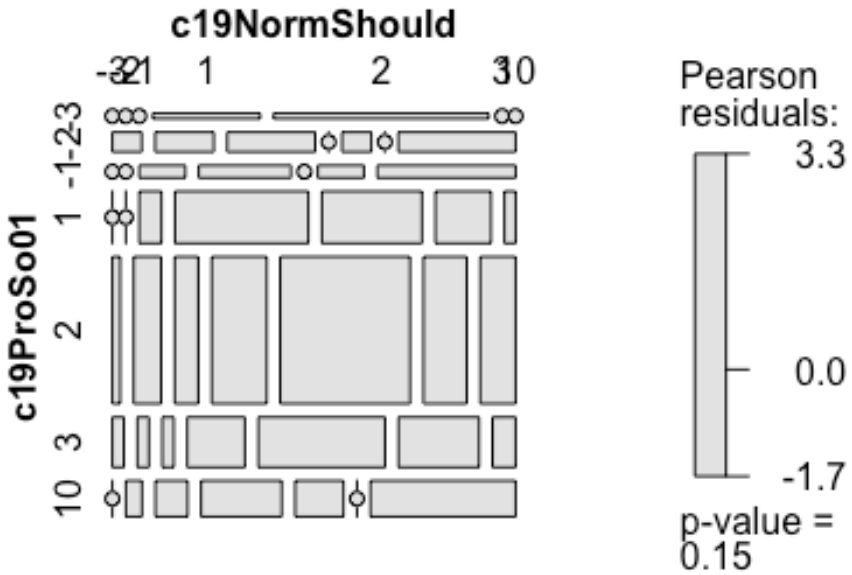
```
library(vcd)

## Loading required package: grid

mosaic( ~ c19ProSo01 + trustGovCtry, data = filter, gp=shading_max)
```



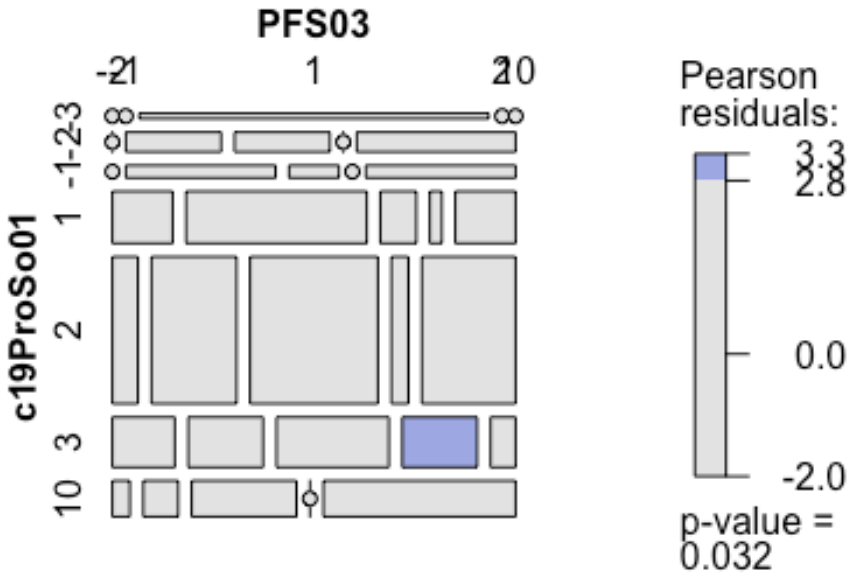

```
library(vcd)

mosaic( ~ c19ProSo01 + edu, data = filter, gp=shading_max)
```



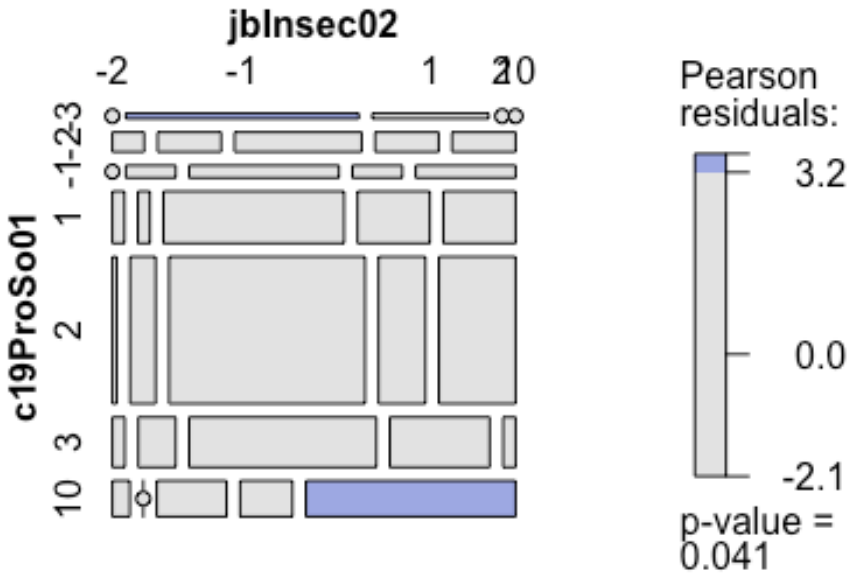

```
library(vcd)

mosaic( ~ c19ProSo01 + c19NormShould, data = filter, gp=shading_max)
```




```
library(vcd)

mosaic( ~ c19ProSo01 + PFS03, data = filter, gp=shading_max)
```




```
library(vcd)

mosaic( ~ c19ProSo01 + jbInsec02, data = filter, gp=shading_max)
```



#question 2c

Firslty extracting entries not having Pakistan as their country in the coded_country and using head to analyse output

```
other_country_data <- cvbase[!(cvbase$grouped_countries=="Pakistan"),]
head(other_country_data)

##       affAnx affBor affCalm affContent affDepr affEnerg affExc affNerv
affExh
## 35185      5      5       1          1       5        1      1       5
2
## 42017      5      3       1          1       2        3      1       5
2
## 55666      1      5       5          4       2        3      3       2
3
## 43319      4      3       3          3       4        3      2       4
4
## 3229       2      3       2          3       2        2      3       3
2
## 58373      3      1       3          3       2        3      2       3
3
##       affInsp affRel PLRAC19 PLRAEco disc01 disc02 disc03 jbInsec01
jbInsec02
```

```
## 35185       1      2       4       6      2      2     -1        -2
2
## 42017       1      1       4       5      1      1     -1        -1
1
## 55666       3      4       3       3      1     -1     -1        -2
2
## 43319       3      2       6       6      1      1      0         0
0
## 3229        4      3       2       4      0      0      2        -2
2
## 58373       3      3       3       3      0      1      0         0
0
##       jbInsec03 jbInsec04 employstatus_1 employstatus_2 employstatus_3
## 35185         2        -2              0              1              0
## 42017        -1        -1              0              0              1
## 55666        -2        -2              0              0              1
## 43319         1         0              0              0              1
## 3229          2        -2              0              0              1
## 58373         1        -1              0              0              1
##       employstatus_4 employstatus_5 employstatus_6 employstatus_7
## 35185              0              0              0              0
## 42017              0              0              0              0
## 55666              0              0              0              0
## 43319              0              0              0              0
## 3229               0              0              0              0
## 58373              0              0              0              0
##       employstatus_8 employstatus_9 employstatus_10 PFS01 PFS02 PFS03
fail01
## 35185              0              0               0     2     2     2
0
## 42017              0              0               0    -1    -2    -2
-1
## 55666              0              1               0    -1     2    -2
-1
## 43319              0              0               0     1     1     0
0
## 3229               0              0               0    -2    -1    -2
-2
## 58373              0              0               0     1     1     1
0
##       fail02 fail03 happy lifeSat MLQ c19NormShould c19NormDo c19IsStrict
## 35185      0      2     3       2  -3             2         2           3
## 42017     -1      2     6       5   2             3        -2           3
## 55666     -2     -2     8       4   3            -1        -1           5
## 43319      0      1     6       3   0             1         1           2
## 3229      -2      1     9       4   1             3         3           6
## 58373     -1      1     9       5   2             2         2           4
##       c19IsPunish c19IsOrg trustGovCtry trustGovState gender age edu
## 35185           2        3            1             1      1   5   3
## 42017           2        4            3             2      1   2   7
```

```
## 55666           4        4            2             1      2   2   6
## 43319           2        2            3             3      2   4   6
## 3229            6        6            5             5      1   5   1
## 58373           4        4            5             4      2   4   4
##            coded_country c19ProSo01 c19ProSo02 c19ProSo03 c19ProSo04
## 35185          Argentina         -2         -3         -3         -3
## 42017        Netherlands          1          2          2          3
## 55666         Kazakhstan          3          3          3          3
## 43319              Japan          1          1          1          1
## 3229             Germany          3          3          3          3
## 58373 Republic of Serbia          2          1          1          1
##       grouped_countries
## 35185   other countries
## 42017   other countries
## 55666   other countries
## 43319   other countries
## 3229    other countries
## 58373   other countries
```

Further, just like the analysis done for 2b, I used chi square methodology to predict p values between categorical variables, on c19ProSo01, lapply for execution of chisq.test function over entire dataset and rbind to combine the results to just showcase relevant values

```
other_country_data[other_country_data== 0] <- 10
hey = lapply(other_country_data[,-c(55, 50)], function(x)
chisq.test(other_country_data[,51], x));
```

```
## Warning in chisq.test(other_country_data[, 51], x): Chi-squared
approximation
## may be incorrect

## Warning in chisq.test(other_country_data[, 51], x): Chi-squared
approximation
## may be incorrect

## Warning in chisq.test(other_country_data[, 51], x): Chi-squared
approximation
## may be incorrect
```

```
do.call(rbind, hey)[,c(1,3)]
```

```
##                 statistic p.value
## affAnx          264.8736  1.826835e-42
## affBor          233.7546  2.67436e-36
## affCalm         385.9885  5.608452e-67
## affContent      391.5818  4.005179e-68
## affDepr         234.2361  2.149799e-36
## affEnerg        540.6779  5.768987e-99
## affExc          383.477   1.833076e-66
```

```
## affNerv         221.2631  7.5831e-34
## affExh          245.1339  1.517954e-38
## affInsp         583.3278  7.263887e-108
## affRel          332.024   5.647461e-56
## PLRAC19         458.6311  1.868579e-71
## PLRAEco         412.1122  2.80962e-62
## disc01          714.7448  1.95821e-135
## disc02          882.7594  6.51559e-171
## disc03          944.2369  6.097863e-184
## jbInsec01       540.5038  6.271511e-99
## jbInsec02       657.0922  2.572631e-123
## jbInsec03       600.1004  2.259392e-111
## jbInsec04       457.5667  1.03386e-81
## employstatus_1  6.633248  0.3560988
## employstatus_2  12.54202  0.05091352
## employstatus_3  47.32309  1.61304e-08
## employstatus_4  15.53769  0.01646265
## employstatus_5  40.82323  3.13735e-07
## employstatus_6  18.11546  0.005950033
## employstatus_7  27.09551  0.0001389615
## employstatus_8  16.7189   0.01037379
## employstatus_9  20.48362  0.002270407
## employstatus_10 73.5221   7.73111e-14
## PFS01           539.4135  1.058076e-98
## PFS02           631.3118  6.575448e-118
## PFS03           588.346   6.490301e-109
## fail01          839.8936  7.671846e-162
## fail02          832.9702  2.232819e-160
## fail03          849.2483  8.059961e-164
## happy           996.775   1.280571e-173
## lifeSat         1025.116  2.551445e-196
## MLQ             1854.813  0
## c19NormShould   2092.407  0
## c19NormDo       1918.318  0
## c19IsStrict     1256.711  2.253346e-245
## c19IsPunish     1065.648  6.930091e-205
## c19IsOrg        1490.032  5.268958e-295
## trustGovCtry    946.338   2.185355e-184
## trustGovState   1253.578  9.200169e-250
## gender          34.20552  0.0006259892
## age             154.6695  8.370554e-15
## edu             275.3009  1.213603e-38
## c19ProSo01      112338    0
## c19ProSo02      13406.7   0
## c19ProSo03      12035.64  0
## c19ProSo04      7598.68   0
```

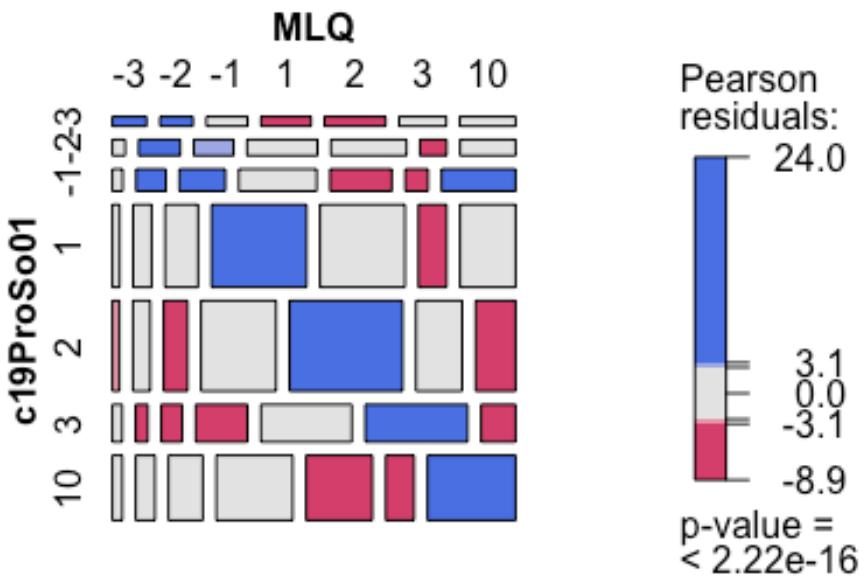
Based on the output from the function we have achieved exceptional 0 p value absolutely rejecting the null hypothesis and correlation etween so many variables being quite stronger

Based on the output the following columns seems to be relevant to predicting pro social behaviours:

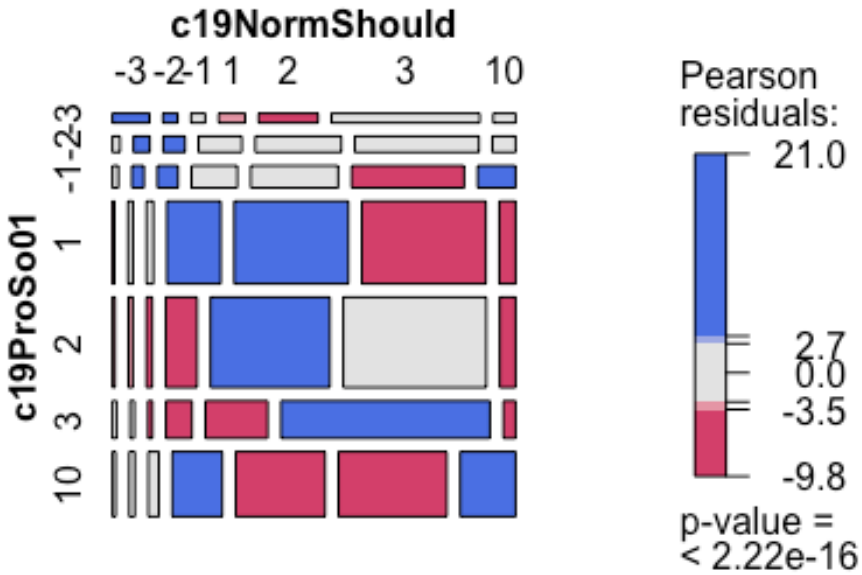
MLQ, c19NormShould, c19NormDo, c19IsOrg, c19IsStrict, c19IsPunish, trustGovCtry, trustGovState, gender, all of PFS columns, lifesat, happy, all, fo, job, insecurity, columns, affEnerg

Further plotting soem of the important variables with lowest p value through mosaic graph

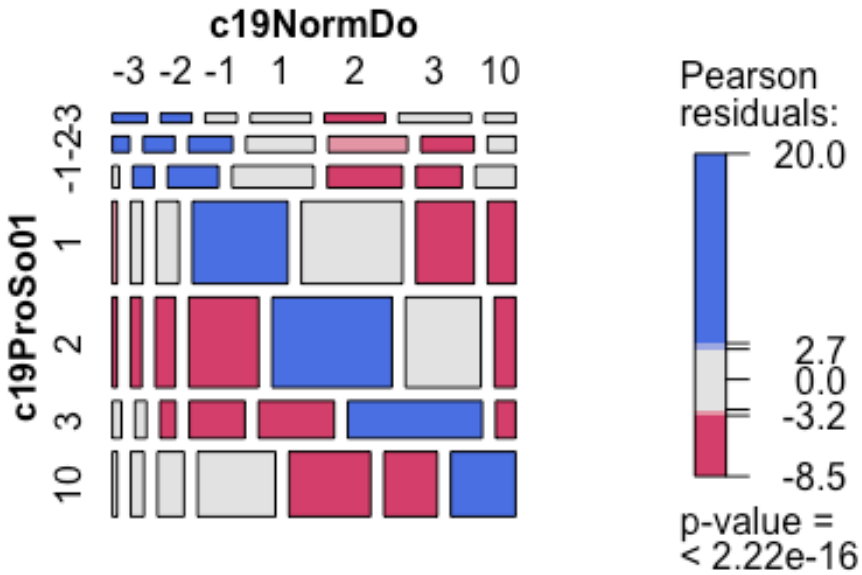
```
library(vcd)

mosaic( ~ c19ProSo01 + MLQ, data = other_country_data, gp=shading_max)
```



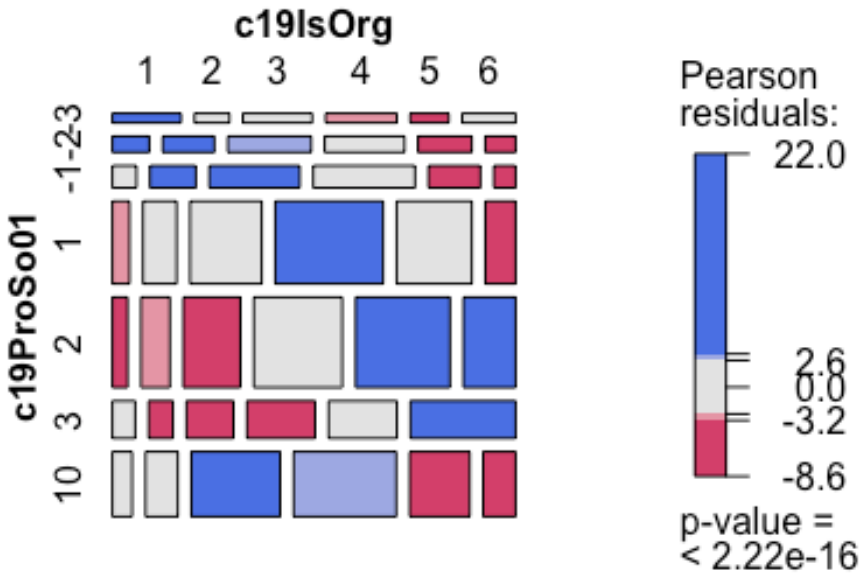

```
library(vcd)

mosaic( ~ c19ProSo01 + c19NormShould, data = other_country_data,
gp=shading_max)
```



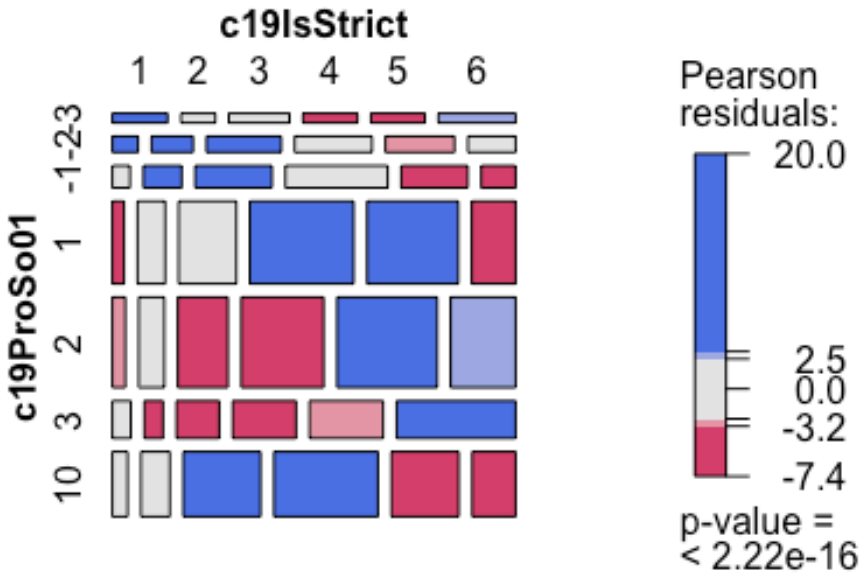

```
library(vcd)

mosaic( ~ c19ProSo01 + c19NormDo, data = other_country_data, gp=shading_max)
```



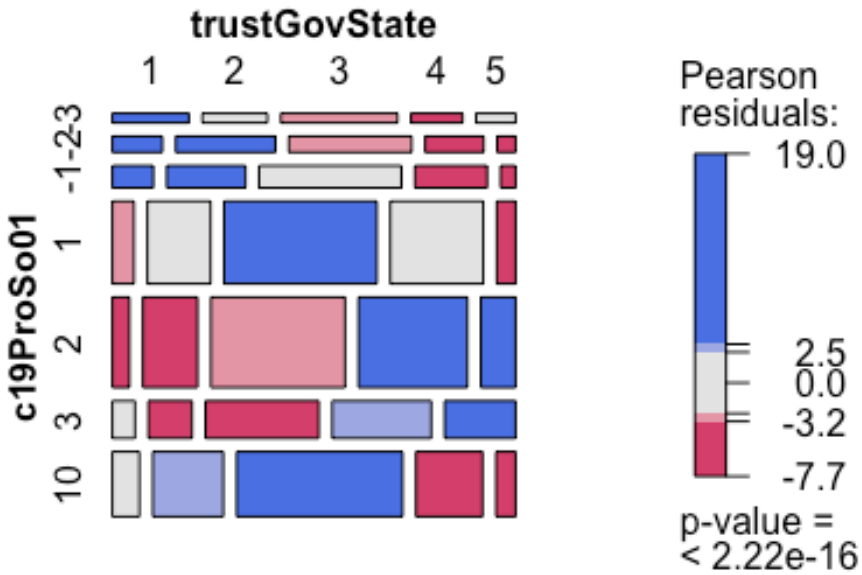

```
library(vcd)

mosaic( ~ c19ProSo01 + c19IsOrg, data = other_country_data, gp=shading_max)
```




```
library(vcd)

mosaic( ~ c19ProSo01 + c19IsStrict, data = other_country_data,
gp=shading_max)
```




```
library(vcd)

mosaic( ~ c19ProSo01 + trustGovState, data = other_country_data,
gp=shading_max)
```



question 3

For question 3 I took data from 2019 world happiness report collect by UN every year. The report outlined the state of world happiness, causes of happiness and misery, and policy implications highlighted by case studies, including several social, economic, health, political or other indicators. source (https://www.kaggle.com/datasets/unsdsn/world-happiness). The following data used is the data in the "2019.csv" file of the dataset.

```
happy <- read.csv("Downloads/2019.csv")
head(happy)

##   Overall.rank Country.or.region Score GDP.per.capita Social.support
## 1            1           Finland 7.769          1.340          1.587
## 2            2           Denmark 7.600          1.383          1.573
## 3            3            Norway 7.554          1.488          1.582
## 4            4           Iceland 7.494          1.380          1.624
## 5            5       Netherlands 7.488          1.396          1.522
## 6            6       Switzerland 7.480          1.452          1.526
##   Healthy.life.expectancy Freedom.to.make.life.choices Generosity
## 1                   0.986                        0.596      0.153
## 2                   0.996                        0.592      0.252
## 3                   1.028                        0.603      0.271
```

```
## 4                    1.026                          0.591      0.354
## 5                    0.999                          0.557      0.322
## 6                    1.052                          0.572      0.263
##   Perceptions.of.corruption
## 1                     0.393
## 2                     0.410
## 3                     0.341
## 4                     0.118
## 5                     0.298
## 6                     0.343
```

Furhter, checking summary of the data for checking data types of the column values.

```
summary(happy)

##   Overall.rank    Country.or.region      Score       GDP.per.capita
##  Min.   :  1.00   Length:156         Min.   :2.853   Min.   :0.0000
##  1st Qu.: 39.75   Class :character   1st Qu.:4.545   1st Qu.:0.6028
##  Median : 78.50   Mode  :character   Median :5.380   Median :0.9600
##  Mean   : 78.50                      Mean   :5.407   Mean   :0.9051
##  3rd Qu.:117.25                      3rd Qu.:6.184   3rd Qu.:1.2325
##  Max.   :156.00                      Max.   :7.769   Max.   :1.6840
##  Social.support  Healthy.life.expectancy Freedom.to.make.life.choices
##  Min.   :0.000   Min.   :0.0000          Min.   :0.0000
##  1st Qu.:1.056   1st Qu.:0.5477          1st Qu.:0.3080
##  Median :1.272   Median :0.7890          Median :0.4170
##  Mean   :1.209   Mean   :0.7252          Mean   :0.3926
##  3rd Qu.:1.452   3rd Qu.:0.8818          3rd Qu.:0.5072
##  Max.   :1.624   Max.   :1.1410          Max.   :0.6310
##    Generosity     Perceptions.of.corruption
##  Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.1087   1st Qu.:0.0470
##  Median :0.1775   Median :0.0855
##  Mean   :0.1848   Mean   :0.1106
##  3rd Qu.:0.2482   3rd Qu.:0.1412
##  Max.   :0.5660   Max.   :0.4530

length(unique(cvbase$coded_country))

## [1] 92
```

Summary output shows average, median, and quantile values of the dataset. Further as there are 150 countries in happy dataset while 92 countries in the cvbase dataset, I eliminated the rows from happy dataset which didn't match the country name in rows from the cvbase dataset

```
# filtering out countries which are common between both the dataset

happy <- happy[happy$Country.or.region %in% cvbase$coded_country,]
```

```
#checking for null values in the filtered column
sum(is.na(happy))

## [1] 0

# checking dimentions of the filtered happy data
dim(happy)

## [1] 87  9

#checking sum of unique values in country in happy data
length(unique(happy$Country.or.region))

## [1] 87
```

Based on the results above, the number of common countries between oth the datasets are 87 countries, thus clustering would be done only on those.

Further performing data scaling on happy dataset before clustering as fluctuations in range of responses between columns can lead to misleading weightage given to attributes. Thus scaling is done to reduce the inputs with range of -4 to 4.

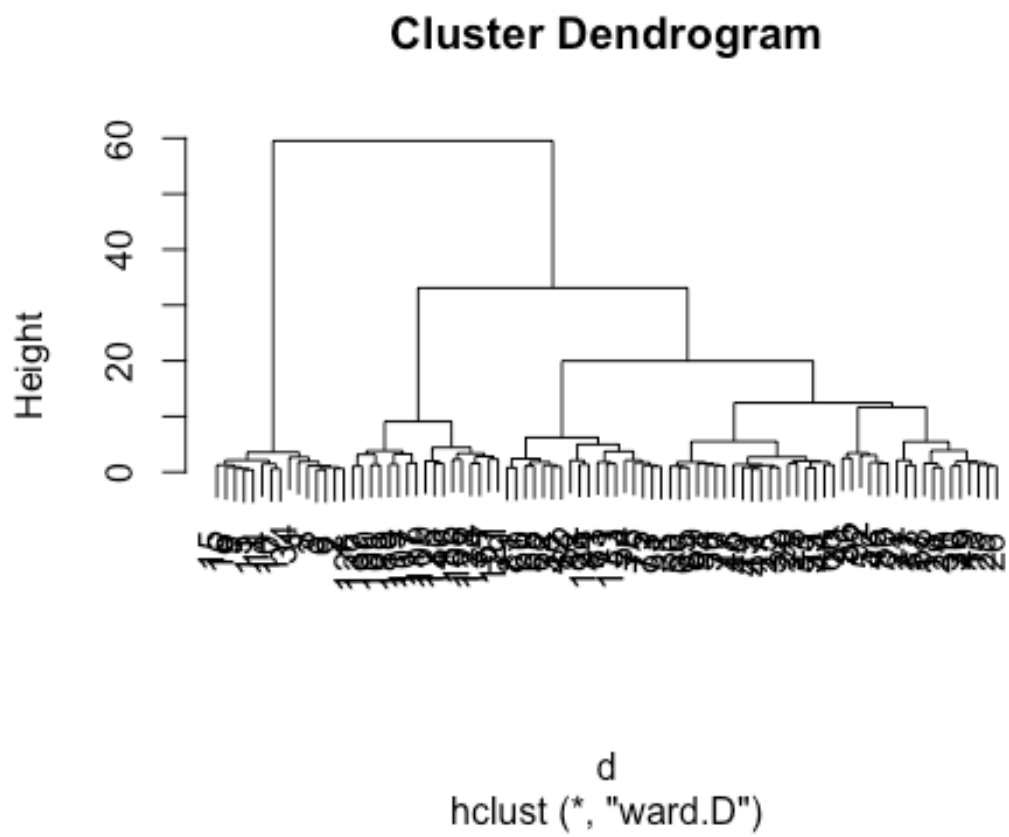
```
# needed to unselect the char columns as scaling doesnt work with char
columns
scaled_data = happy[c(-1,-2)]
# scale function to scale the datapoints
scaled_data = scale(scaled_data)
# finally cheking the dimentions of the output
dim(scaled_data)

## [1] 87  7
```

Further I performed Hierarchical clustering over dataset as it does not require us to pre-specify the number of clusters to be generated as is required by the k-means approach. Furhter, it is better for visualization purposes for it's output results in an attractive tree-based representation of the observations, called a dendrogram.

Initially setting the number of clusters to 7 for having 87 different datasets and requiremnt of formation of clusters among max 8 countries so based on equavent distribution, I used 7 cluster for the model.

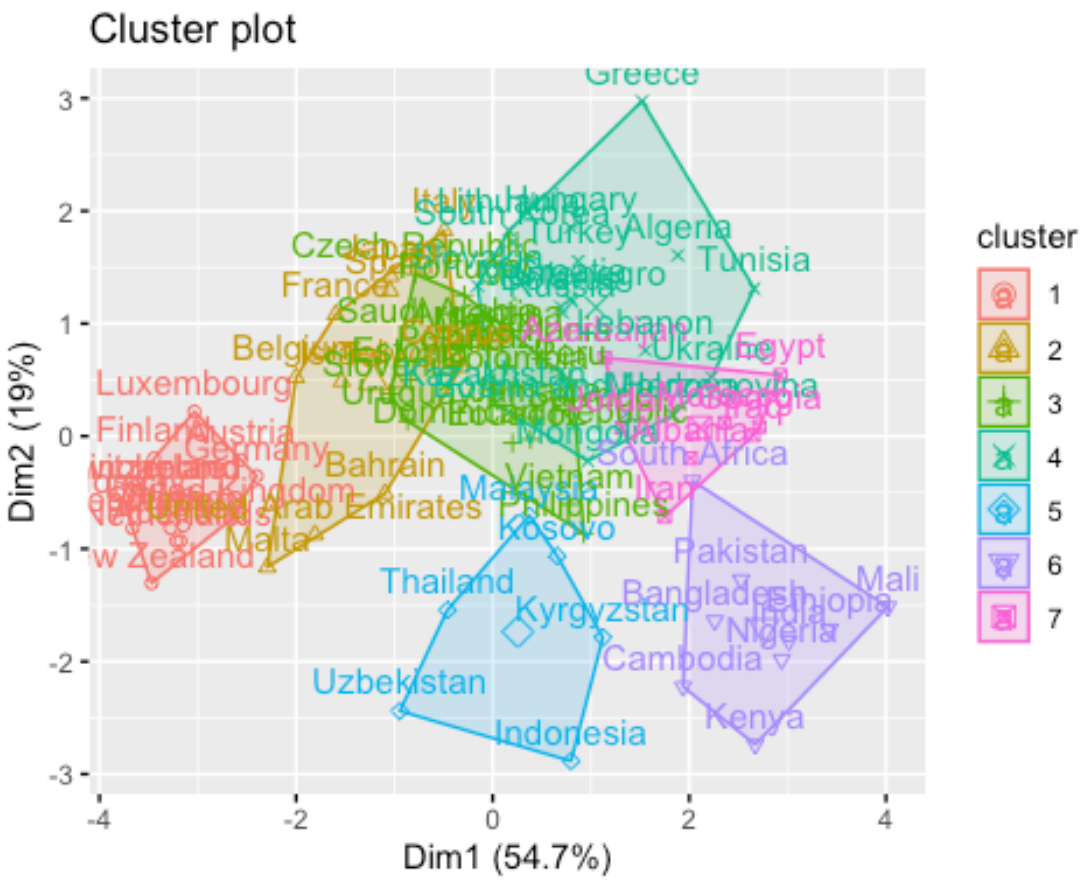
```
# Create distance matrix
d <- dist(scaled_data, method = 'euclidean')

# Ward Hierarchical Clustering
fit <- hclust(d, method = 'ward.D')
fit2 <- hclust(d, method = "complete")

# Create 7 clusters again
groups <- cutree(fit, k = 7)
```

```
# Attach labels
happy$Cluster <- groups

#plotting fit
plot(fit)
```



Further using factoextra library for it's fviz_cluster function providing a grouped cluster plot for the data generated

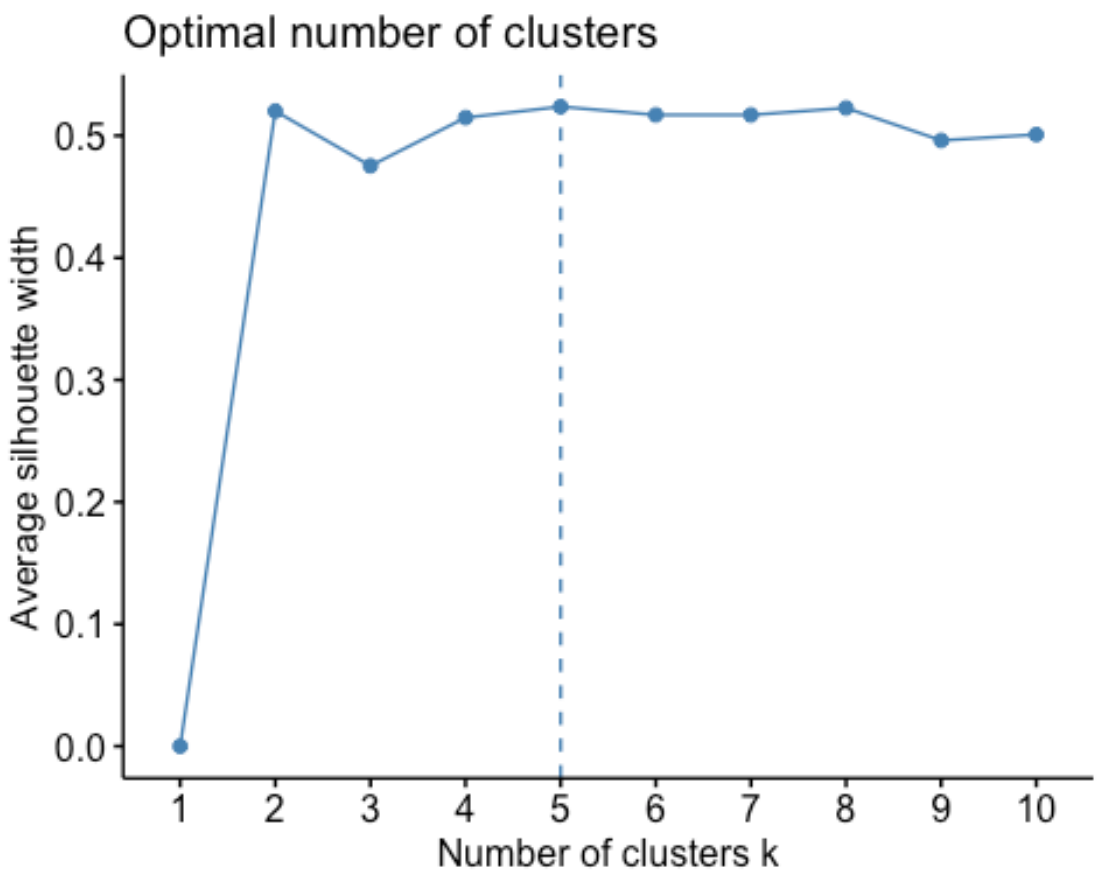
```
library(factoextra)

## Welcome! Want to learn more? See two factoextra-related books at
https://goo.gl/ve3WBa

rownames(scaled_data) = happy$Country.or.region
fviz_cluster(list(data = scaled_data, cluster = groups))
```



Based on the clusters formed we can see Pakistan being allocated to cluster with Bangladesh, Mali, Kenya, etc.

Further analysing the number of clusters again for the model based on the data

```
fviz_nbclust(happy[c(-1,-2)], FUN = hcut, method = "silhouette")
```



Based on the outputs of the plot, it cshows optimal cluster = 2, however as we need fewer countries in a cluster, I believe the use of 7 clusters as per the graph provides nearly optimal results too inline with 2 cluster scenario.

Question 3b

Furhter in order to identify the other countries included in the cluster alongside Pakistan I used with function of R to merge the 2 conditions fo extracting entries having same cluster number as the entry having country name as Pakistan

```
# getting the cluster number of focus country Pakistan
with(happy, Cluster[Country.or.region == "Pakistan"])

## [1] 6
```

Based on the output shown Pakistan belongs to cluster 6, so extracting all the countries with similar cluster number and saving them to clu_no_6 vector

```
clu_no_6 = with(happy, Country.or.region[Cluster == 6])
clu_no_6
```

```
## [1] "Pakistan"     "Nigeria"      "South Africa" "Cambodia"     "Kenya"
## [6] "Bangladesh"   "Mali"         "Ethiopia"     "India"
```

further for analysing the participants response to Pro social covid attitude, I used %in% to extract entries from cvbase having coded_country same as the one in the clu_no_6 vector.

```
# extracting rows from cvbase wherein the country_code is same as the
extracted countries in clu_no_6
three_b_data <- cvbase[cvbase$coded_country %in% clu_no_6,]

dim(three_b_data)

## [1] 792  55

dim(cvbase)

## [1] 18903    55
```

Further, just like the analysis done for 2b, I used chi square methodology to predict p values between categorical variables, on c19ProSo01, lapply for execution of chisq.test function over entire dataset and rbind to combine the results to just showcase relevant values

```
three_b_data[three_b_data== 0] <- 10
hey = lapply(three_b_data[,-c(55, 50)], function(x)
chisq.test(three_b_data[,51], x));

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect
```

```
## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
```

```
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
```

```
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect
```

```
## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect

## Warning in chisq.test(three_b_data[, 51], x): Chi-squared approximation
may be
## incorrect
```

```
do.call(rbind, hey)[,c(1,3)]
```

```
##                  statistic p.value
## affAnx          36.62065  0.04772294
## affBor          23.1974   0.5081712
## affCalm         31.99875  0.1270236
## affContent      28.21525  0.2510656
## affDepr         40.2596   0.0200536
## affEnerg        36.28614  0.05147544
## affExc          33.50315  0.09389427
## affNerv         47.94609  0.002563206
## affExh          40.7657   0.01767008
```

```
## affInsp         39.86489  0.02211217
## affRel          26.45114  0.3306785
## PLRAC19         57.05961  0.06049608
## PLRAEco         73.18379  0.002033798
## disc01          69.87008  2.287802e-06
## disc02          86.41592  5.602993e-09
## disc03          64.73734  1.330032e-05
## jbInsec01       42.63708  0.01093897
## jbInsec02       58.80202  9.410364e-05
## jbInsec03       74.84006  3.950667e-07
## jbInsec04       33.65067  0.09107168
## employstatus_1  9.995069  0.1248598
## employstatus_2  7.065096  0.3148691
## employstatus_3  2.665662  0.8494862
## employstatus_4  10.20593  0.1162434
## employstatus_5  4.56216   0.6010622
## employstatus_6  8.515459  0.2027174
## employstatus_7  16.32463  0.01211366
## employstatus_8  15.40115  0.01735597
## employstatus_9  10.12262  0.1195823
## employstatus_10 7.574628  0.2709518
## PFS01           41.09354  0.01626759
## PFS02           61.37811  4.070821e-05
## PFS03           72.74903  8.319514e-07
## fail01          36.28896  0.05144267
## fail02          62.90244  2.458929e-05
## fail03          69.46318  2.635881e-06
## happy           80.5699   0.01102421
## lifeSat         67.62869  0.0001001191
## MLQ             77.47562  7.262828e-05
## c19NormShould   101.7959  3.378913e-08
## c19NormDo       92.8122   6.564998e-07
## c19IsStrict     77.63168  4.302604e-06
## c19IsPunish     74.87846  1.047527e-05
## c19IsOrg        64.19458  0.0002778212
## trustGovCtry    62.76449  2.574334e-05
## trustGovState   58.36304  0.0001083491
## gender          22.4667   0.03261022
## age             55.78996  0.01874478
## edu             47.84572  0.08953424
## c19ProSo01      4752      0
## c19ProSo02      531.9893  1.514736e-89
## c19ProSo03      497.9564  1.214498e-82
## c19ProSo04      330.3474  2.929117e-49
```

Based on the output it seems like the might be some internal working error in the chisq.test function which might be producing error lines, as 0 values are removed from the dataset. Based on the output the predictors with lower p values signifies the factors of most significance proving lower probability (p value) against null hypothesis.

Thus based on the the following columns seems related to pro social covid attitude in dataset for participants from clustered countries:

trustGovCtry, c19IsStrict, c19IsPunish, disc02, and all of other disc columns, MLQ, c19NormShould, PFS03, fail03

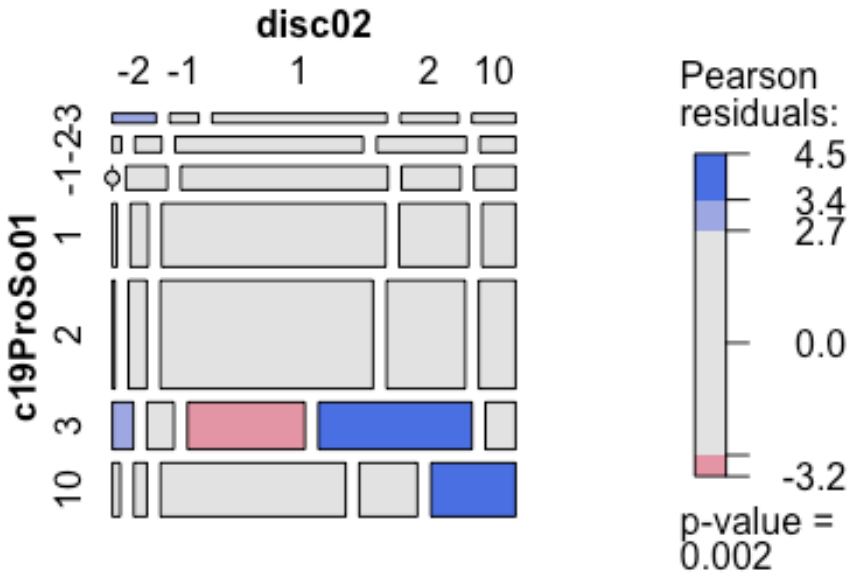
comparing the 2b output the following columns seems to be relevant to predicting pro social behaviours:

MLQ, c19NormShould, c19NormDo, c19IsOrg, c19IsStrict, c19IsPunish, trustGovCtry, trustGovState, gender, all of PFS columns, lifesat, happy, all, fo, job, insecurity, columns, affEnerg

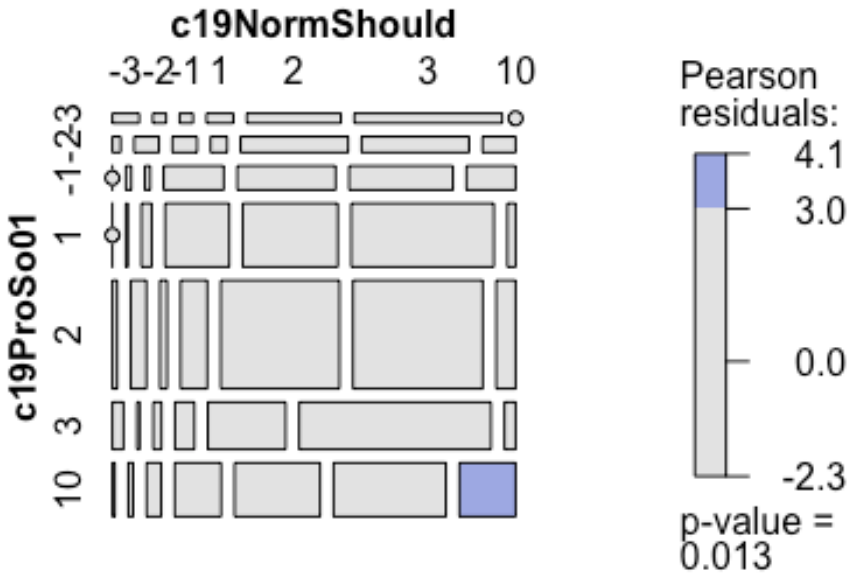
Quite many attributes, in fact most of the column values match between the datasets however, data set in 3b was better for visualization and prediction as it had p values nearly touching nill values showing strong correlation between columns which is very much lacking for the case of 3c data, the prediction values have also got p values nearby 1 too. 2c was better for having greater number of datapoints

Further I tried visualizing the correlation between the categorical variables plotting the mosaic graph. Which accurately depicts the relation between the countries. Using the mosaic function of the vcd library

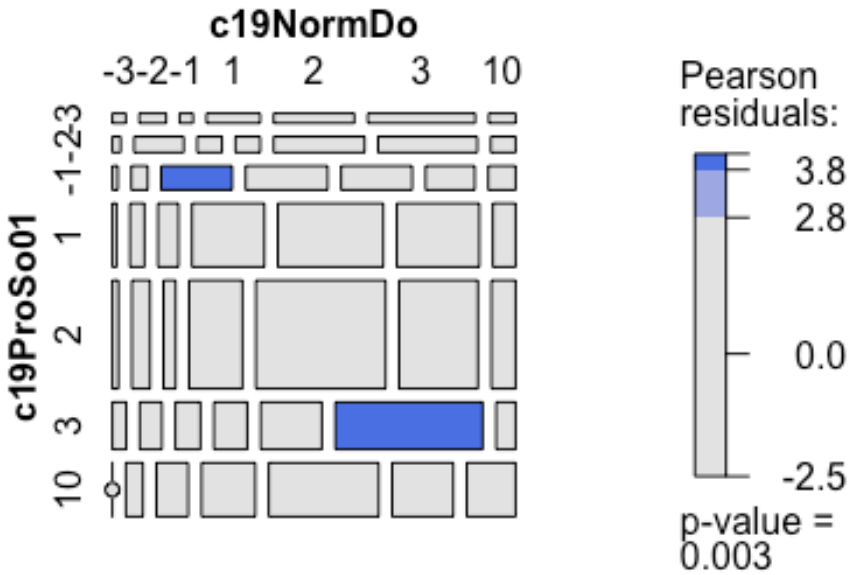
```
library(vcd)

mosaic( ~ c19ProSo01 + disc02, data = three_b_data, gp=shading_max)
```



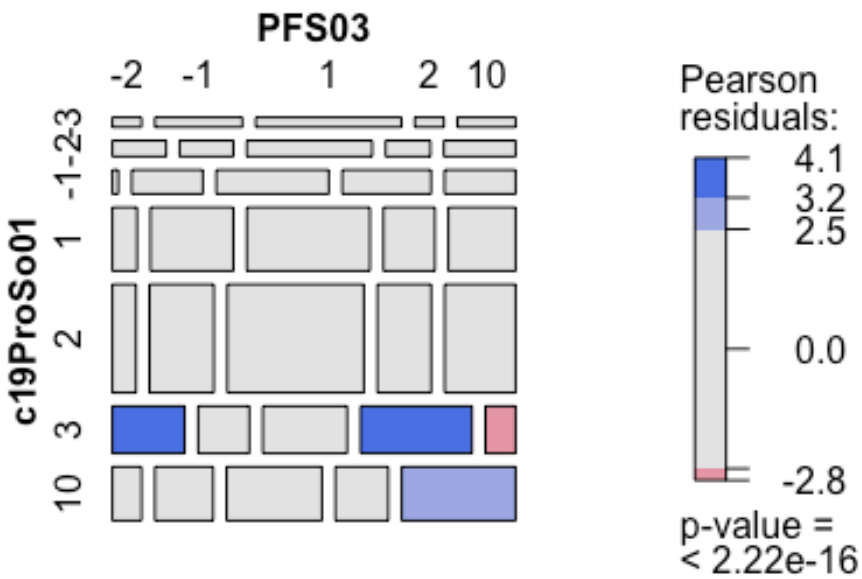

```
library(vcd)

mosaic( ~ c19ProSo01 + c19NormShould, data = three_b_data, gp=shading_max)
```



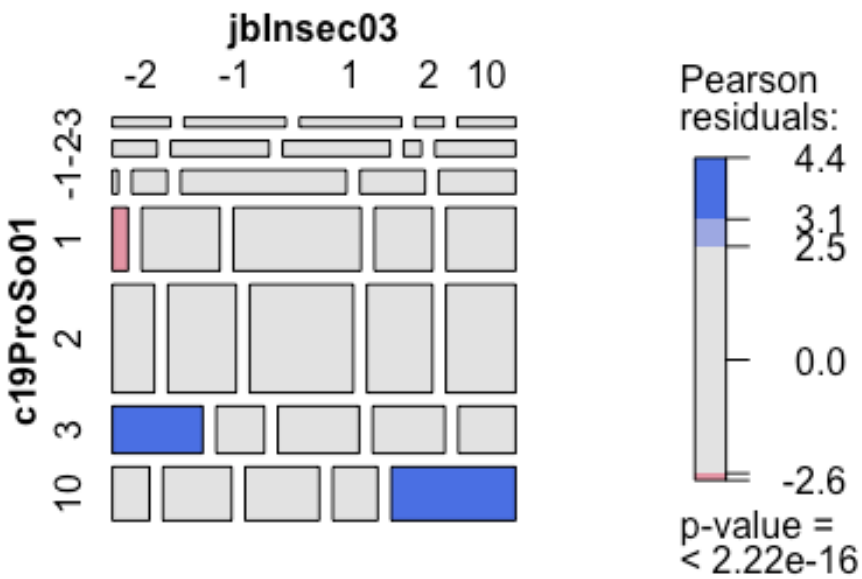

```
library(vcd)

mosaic( ~ c19ProSo01 + c19NormDo, data = three_b_data, gp=shading_max)
```



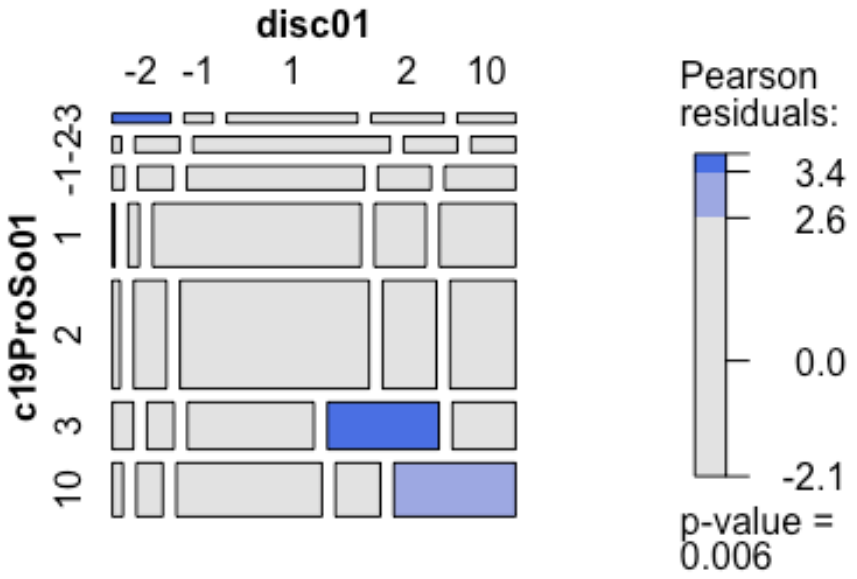

```
library(vcd)

mosaic( ~ c19ProSo01 + PFS03, data = three_b_data, gp=shading_max)
```




```
library(vcd)

mosaic( ~ c19ProSo01 + jbInsec03, data = three_b_data, gp=shading_max)
```




```
library(vcd)

mosaic( ~ c19ProSo01 + disc01, data = three_b_data, gp=shading_max)
```


disc01
-2
-1
1
2
10
c19ProSo01
-2-3
-1
1
2
3
10
Pearson residuals:
3.4
2.6
0.0
-2.1
p-value = 0.006