607_Fall_2021_Tidyversecreate_Schmalfeld

Mark Schmalfeld

10/22/2021

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.4     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(latexpdf)
```

Overview of 16 personality factor (PF) test from Kagle dataset

##The dataset for this vignette uses the Kagle dataset for the 16 personality factor test.

###1)Pre-setup required using the sumarise, mutate, and rowwise functions to prepare the data prior to graphing. ###2)GGPlot is used to explore a few of the basic data variables and to demonstrate how ggplot can be used in several different ways.

What is the 16 PF data set? Questions, answers, and metadata collected from 49,159 16 Factor Personality Tests. The data was hosted on OpenPsychometrics.org a nonprofit effort to educate the public about psychology and to collect data for psychological research. Their notes on the data collected in the codebook.html.

From Wikipedia: The Sixteen Personality Factor Questionnaire (16PF) is a self-report personality test developed over several decades of empirical research by Raymond B. Cattell, Maurice Tatsuoka and Herbert Eber. The 16PF provides a measure of normal personality and can also be used by psychologists, and other mental health professionals, as a clinical instrument to help diagnose psychiatric disorders, and help with prognosis and therapy planning. The 16PF can also provide information relevant to the clinical and counseling process, such as an individual's capacity for insight, self-esteem, cognitive style, internalization of standards, openness to change, capacity for empathy, level of interpersonal trust, quality of attachments, interpersonal needs, attitude toward authority, reaction toward dynamics of power, frustration tolerance, and coping style. Thus, the 16PF instrument provides clinicians with a normal-range measurement of anxiety, adjustment, emotional stability and behavioral problems. Clinicians can use 16PF results to identify effective strategies for establishing a working alliance, to develop a therapeutic plan, and to select effective therapeutic interventions or modes of treatment. It can also be used within other areas of psychology, such as career and occupational selection.

16 factors and column letter sequence of questions it is correlated with in the table

A) Warmth: Outgoing versus reserved
B) Reasoning: Abstract versus concrete
C) Emotional stability: Calm versus high-strung
D) Dominance: Forceful versus submissive E)Liveliness: Spontaneous versus restrained F)Perfectionism: Controlled versus undisciplined
E) Social boldness: Uninhibited versus shy
F) Sensitivity: Tender-hearted versus tough-minded
G) Vigilance: Suspicious versus trusting
H) Privateness: Discreet versus open
I) Openness to change: Flexible versus attached to the familiar
J) Apprehension: Worried versus confident
K) Abstractedness: Imaginative versus practical
L) Self-reliance: Self-sufficient versus dependent
M) Rule-consciousness: Conforming versus non-conforming
N) Tension: Inpatient versus relaxed

Datasource URL

https://www.kaggle.com/lucasgreenwell/16-factor-personality-test-responses

Github download URL

https://raw.githubusercontent.com/schmalmr/607_Fall_2021_Tidyvinette_Schmalfeld/main/data.csv

https://raw.githubusercontent.com/schmalmr/607_Fall_2021_Tidyvinette_Schmalfeld/main/codebook.html

```
url="https://raw.githubusercontent.com/schmalmr/607_Fall_2021_Tidyvinette_Schmalfeld/main/data.csv"

pfdf<-read.csv(url,header=TRUE, sep="")
```

##First pass data sorting and analysis summary of the 16 categories

1) Each letter is correlated with a specific series of questions for a given personality factor 2)Initial analysis is to calculate the row wise mean for each of the personality factors to allow for data analysis on the specific factor related to the individual demographics 3)Demographics is primarily age, gender, country, set assessent of accuracy, time to take text and location where test was taken (or sourced)
2) After the first pass simple averages completed in new columns. Some graphical analysis of the data is completed.

#What is "summarise" used for in this case?

Details can be obtained from the help bar in R by typing in summarize or summarise.

#####A few of the details and examples are below from the help menu: summarise() creates a new data frame. It will have one (or more) rows for each combination of grouping variables; if there are no grouping variables, the output will have a single row summarising all observations in the input. It will contain one column for each grouping variable and one column for each of the summary statistics that you have specified.

summarise() and summarize() are synonyms.

Usage summarise(.data, ..., .groups = NULL)

summarize(.data, ..., .groups = NULL)

Examples ### A summary applied to ungrouped tbl returns a single row mtcars %>% summarise(mean = mean(disp), n = n())

Usually, you'll want to group first

mtcars %>% group_by(cyl) %>% summarise(mean = mean(disp), n = n())

dplyr 1.0.0 allows to summarise to more than one value:

mtcars %>% group_by(cyl) %>% summarise(qs = quantile(disp, c(0.25, 0.75)), prob = c(0.25, 0.75))

You use a data frame to create multiple columns so you can wrap

this up into a function:

my_quantile <- function(x, probs) { tibble(x = quantile(x, probs), probs = probs) } mtcars %>% group_by(cyl) %>% summarise(my_quantile(disp, c(0.25, 0.75)))

Each summary call removes one grouping level (since that group

is now just a single row)

mtcars %>% group_by(cyl, vs) %>% summarise(cyl_n = n()) %>% group_vars()

#What is "rowwise" function and how is it used?

##Key summary and a few examples from help menu. Find full details in the R help bar. Rowwise() allows you to compute on a data frame a row-at-a-time. This is most useful when a vectorised function doesn't exist.

Most dplyr verbs preserve row-wise grouping. The exception is summarise(), which return a grouped_df. You can explicitly ungroup with ungroup() or as_tibble(), or convert to a grouped_df with group_by().

Usage rowwise(data, . . .)

Examples of the use of Rowwise df <- tibble(x = runif(6), y = runif(6), z = runif(6)) ### Compute the mean of x, y, z in each row df %>% rowwise() %>% mutate(m = mean(c(x, y, z))) ### use c_across() to more easily select many variables df %>% rowwise() %>% mutate(m = mean(c_across(x:z)))

Compute the minimum of x and y in each row

df %>% rowwise() %>% mutate(m = min(c(x, y, z))) ###In this case you can use an existing vectorised function: df %>% mutate(m = pmin(x, y, z)) ### Where these functions exist they'll be much faster than rowwise ###so be on the lookout for them.

Here I supply variables to preserve after the summary

params %>% rowwise(sim) %>% summarise(z = rnorm(n, mean, sd))

#What is "mutate" and how is it used?

Key summary of Mutate and a few examples from the help section of R. Additional details found in the help menu in R.

Create, modify, and delete columns Description mutate() adds new variables and preserves existing ones; transmute() adds new variables and drops existing ones. New variables overwrite existing variables of the same name. Variables can be removed by setting their value to NULL.

Usage mutate(.data, ...)

S3 method for class 'data.frame'

mutate(.data, ..., .keep = c("all", "used", "unused", "none"), .before = NULL, .after = NULL)

Examples ### Newly created variables are available immediately starwars %>% select(name, mass) %>% mutate(mass2 = mass * 2, mass2_squared = mass2 * mass2)

As well as adding new variables, you can use mutate() to

remove variables and modify existing variables.

starwars %>% select(name, height, mass, homeworld) %>% mutate(mass = NULL, height = height * 0.0328084 # convert to feet)

Use across() with mutate() to apply a transformation

to multiple columns in a tibble.

starwars %>% select(name, homeworld, species) %>% mutate(across(!name, as.factor))

Below is the application examples of summarize to get the column means and standard deviations.The rowwise means and mutate function calculate the mean for each row for the associated group of questions that are of the same class. There are 16 examples that you can see and compare to what is in the data file.

```
(pfdfmean<-pfdf %>% summarise_if(is.numeric, mean))
```

```
##         A1       A2       A3       A4       A5       A6       A7       A8
## 1 3.649566 3.791513 3.799121 3.633495 3.828495 3.670376 3.869871 2.960658
##         A9      A10       B1       B2       B3       B4       B5       B6
## 1 2.140463 2.428853 3.677902 3.495148 4.143372 4.315853 3.976586 3.496389
##         B7     B8      B9      B10      B11      B12      B13      C1       C2
## 1 3.505889 3.7919 2.98812 2.609309 2.723489 1.912081 2.152648 2.80974 3.643626
##        C3       C4       C5       C6       C7       C8       C9      C10
## 1 3.37159 3.349499 2.931528 2.896804 2.750239 2.312679 2.337761 2.516731
##         D1       D2       D3       D4       D5       D6       D7       D8
## 1 3.454098 3.309262 3.549645 3.475355 3.634919 3.551252 2.652658 2.100694
##         D9      D10       E1       E2      E3       E4       E5       E6
## 1 2.861714 2.462235 2.547509 2.875669 3.78773 2.737301 3.872028 2.797128
##         E7       E8       E9      E10       F1      F2      F3       F4
## 1 2.176285 3.104294 2.077524 2.322138 3.059928 3.62782 2.55174 3.610265
```

```
##         F5       F6       F7       F8       F9      F10       G1       G2
## 1 3.341585 2.553795 2.873919 3.512419 2.563457 3.218861 3.433837 2.917736
##         G3       G4       G5       G6       G7       G8       G9     G10
## 1 3.152139 3.307289 3.379015 2.833357 2.778433 2.376248 3.435139 3.08206
##         H1      H2      H3       H4       H5     H6       H7       H8       H9
## 1 3.864013 3.93399 3.53449 2.253321 3.097988 3.6326 2.456661 2.526658 1.949592
##        H10       I1       I2       I3      I4       I5       I6      I7
## 1 2.498423 2.690291 3.331089 3.158425 2.85315 3.457515 2.091255 3.06123
##         I8       I9      I10       J1       J2       J3       J4       J5
## 1 3.272137 3.374031 3.237983 3.559938 3.830489 3.534449 3.855713 3.334181
##         J6       J7       J8       J9      J10       K1      K2       K3    K4
## 1 2.850424 3.347993 2.834537 2.375882 2.385647 3.044753 2.99583 2.712504 3.264
##        K5       K6       K7      K8       K9      K10      L1       L2       L3
## 1 3.14685 3.156268 3.030289 2.80022 3.085376 3.521898 3.35318 2.633048 3.116052
##         L4      L5       L6       L7       L8      L9      L10       M1
## 1 3.698631 3.50298 3.182998 2.996847 2.661913 2.89772 3.389715 3.829289
##         M2       M3       M4       M5       M6      M7       M8       M9
## 1 3.832381 4.207246 3.478793 3.417177 2.155943 2.12663 2.192254 2.141276
##        M10       N1       N2       N3       N4       N5       N6       N7
## 1 2.336744 2.831099 3.530768 3.881202 3.315588 3.668952 3.687199 4.131329
##         N8       N9     N10       O1       O2       O3       O4       O5
## 1 3.608352 3.451555 2.62011 3.574625 2.713013 3.589211 3.447141 3.577026
##         O6       O7       O8      O9      O10       P1       P2       P3
## 1 2.948189 2.841087 3.155943 2.99467 3.436238 2.942452 2.674729 2.978275
##         P4       P5       P6       P7      P8       P9      P10     age
## 1 3.107529 2.601518 3.095486 2.796823 2.93102 3.643829 3.463964 43713.2
##     gender accuracy   source elapsed
## 1 1.600195 47641.19 3.021502      NA
```

```
(pfdfsd<-pfdf %>% summarise_if(is.numeric, sd))
```

```
##         A1      A2       A3       A4        A5        A6        A7       A8
## 1 1.140966 1.05912 1.104136 1.067361 0.9925374 0.9760629 0.9649516 1.165763
##         A9      A10       B1       B2        B3        B4        B5       B6
## 1 1.043318 1.079312 1.006225 1.081588 0.9221223 0.8339831 0.9940223 1.069849
##         B7      B8       B9      B10      B11      B12      B13       C1
## 1 1.109821 1.00747 1.260709 1.151811 1.256889 1.019821 1.201195 1.177428
##         C2       C3       C4       C5       C6      C7       C8       C9
## 1 1.159866 1.041797 1.138612 1.213302 1.296478 1.24385 1.244499 1.188539
##       C10      D1       D2       D3       D4       D5       D6       D7      D8
## 1 1.14385 1.06666 1.121973 1.101547 1.118438 1.000152 0.980615 1.089641 0.93607
##         D9      D10       E1       E2       E3       E4        E5      E6
## 1 1.050845 1.193678 1.182456 1.316074 1.105468 1.273396 0.9581962 1.26273
##         E7       E8       E9      E10       F1       F2       F3       F4
## 1 1.116585 1.294537 1.032109 1.212824 1.138122 1.058797 1.547308 1.148554
##         F5       F6       F7      F8       F9      F10       G1       G2
## 1 1.360056 1.126102 1.170314 1.32135 1.116343 1.122671 1.116095 1.290501
##         G3       G4       G5       G6       G7       G8       G9      G10
## 1 1.254693 1.250608 1.179333 1.268525 1.234651 1.152063 1.245578 1.183644
##         H1       H2       H3       H4       H5      H6       H7       H8
## 1 1.224138 1.089703 1.314799 1.185047 1.377407 1.17974 1.292706 1.251914
##         H9     H10       I1       I2       I3       I4       I5       I6
## 1 1.045052 1.16279 1.186722 1.135489 1.096434 1.144267 1.143699 1.109643
##        I7       I8       I9      I10       J1       J2      J3       J4
```

```
## 1 1.05434 1.104762 1.041411 1.077406 1.145495 1.154634 1.22297 1.127113
##         J5       J6       J7       J8       J9      J10       K1       K2
## 1 1.063431 1.079032 1.090795 1.085788 1.20569 1.186767 1.220923 1.278184
##        K3       K4       K5       K6       K7       K8      K9      K10
## 1 1.27592 1.242364 1.191471 1.280072 1.235814 1.211434 1.25694 1.147066
##         L1       L2       L3       L4       L5       L6       L7       L8
## 1 1.245947 1.170879 1.249182 1.129715 1.204617 1.260166 1.161363 1.257849
##         L9      L10       M1       M2        M3       M4       M5       M6
## 1 1.156476 1.157339 1.142065 1.038684 0.8644276 1.040082 1.157362 1.111218
##         M7       M8       M9      M10       N1       N2       N3       N4
## 1 1.152556 1.107748 1.013617 1.114499 1.205135 1.127255 1.027692 1.134726
##         N5       N6        N7       N8       N9      N10       O1       O2
## 1 1.198753 1.123167 0.9105632 1.073697 1.190972 1.204121 1.096303 1.226755
##         O3       O4       O5       O6       O7       O8       O9      O10
## 1 1.020304 1.123379 1.007843 1.219152 1.208809 1.335426 1.285741 1.175737
##         P1       P2       P3       P4       P5       P6       P7       P8
## 1 1.229446 1.240915 1.166713 1.139767 1.019074 1.155755 1.169098 1.209906
##         P9      P10     age    gender accuracy   source elapsed
## 1 1.103671 1.119474 9685640 0.5109214  9704611 12.93508      NA
```

```
pfdf<-pfdf %>% rowwise() %>% mutate(Amean=mean(c(A1,A2,A3,A4,A5,A6,A7,A8,A9,A10)))
pfdf<-pfdf %>% rowwise() %>% mutate(Bmean=mean(c(B1,B2,B3,B4,B5,B6,B7,B8,B9,B10, B11, B12,B13)))
pfdf<-pfdf %>% rowwise() %>% mutate(Cmean=mean(c(C1,C2,C3,C4,C5,C6,C7,C8,C9,C10)))
pfdf<-pfdf %>% rowwise() %>% mutate(Dmean=mean(c(D1,D2,D3,D4,D5,D6,D7,D8,D9,D10)))
pfdf<-pfdf %>% rowwise() %>% mutate(Emean=mean(c(E1,E2,E3,E4,E5,E6,E7,E8,E9,E10)))
pfdf<-pfdf %>% rowwise() %>% mutate(Fmean=mean(c(F1,F2,F3,F4,F5,F6,F7,F8,F9,F10)))
pfdf<-pfdf %>% rowwise() %>% mutate(Gmean=mean(c(G1,G2,G3,G4,G5,G6,G7,G8,G9,G10)))
pfdf<-pfdf %>% rowwise() %>% mutate(Hmean=mean(c(H1,H2,H3,H4,H5,H6,H7,H8,H9,H10)))
pfdf<-pfdf %>% rowwise() %>% mutate(Imean=mean(c(I1,I2,I3,I4,I5,I6,I7,I8,I9,I10)))
pfdf<-pfdf %>% rowwise() %>% mutate(Jmean=mean(c(J1,J2,J3,J4,J5,J6,J7,J8,J9,J10)))
pfdf<-pfdf %>% rowwise() %>% mutate(Kmean=mean(c(K1,K2,K3,K4,K5,K6,K7,K8,K9,K10)))
pfdf<-pfdf %>% rowwise() %>% mutate(Lmean=mean(c(L1,L2,L3,L4,L5,L6,L7,L8,L9,L10)))
pfdf<-pfdf %>% rowwise() %>% mutate(Mmean=mean(c(M1,M2,M3,M4,M5,M6,M7,M8,M9)))
pfdf<-pfdf %>% rowwise() %>% mutate(Nmean=mean(c(N1,N2,N3,N4,N5,N6,N7,N8,N9,N10)))
pfdf<-pfdf %>% rowwise() %>% mutate(Omean=mean(c(O1,O2,O3,O4,O5,O6,O7,O8,O9,O10)))
pfdf<-pfdf %>% rowwise() %>% mutate(Pmean=mean(c(P1,P2,P3,P4,P5,P6,P7,P8,P9,P10)))
glimpse(pfdf)
```

```
## Rows: 49,159
## Columns: 185
## Rowwise:
## $ A1       <int> 1, 4, 3, 4, 4, 3, 4, 4, 2, 5, 5, 5, 5, 5, 3, 4, 5, 5, 4, 4, 5~
## $ A2       <int> 4, 3, 4, 5, 0, 5, 2, 5, 4, 5, 5, 4, 4, 4, 3, 4, 5, 5, 4, 4, 4~
## $ A3       <int> 2, 4, 4, 4, 4, 4, 5, 4, 2, 3, 4, 4, 5, 5, 3, 4, 5, 5, 4, 4, 4~
## $ A4       <int> 3, 3, 4, 4, 4, 4, 4, 4, 5, 3, 5, 4, 4, 4, 3, 4, 5, 4, 3, 4, 4~
## $ A5       <int> 3, 4, 4, 4, 4, 4, 2, 4, 5, 5, 5, 5, 4, 4, 3, 4, 5, 5, 5, 4, 4~
## $ A6       <int> 2, 4, 4, 3, 3, 5, 4, 4, 4, 4, 5, 5, 4, 4, 3, 4, 5, 3, 4, 4, 4~
## $ A7       <int> 3, 4, 4, 3, 5, 5, 4, 5, 5, 4, 4, 5, 3, 3, 2, 4, 5, 4, 5, 5, 4~
## $ A8       <int> 4, 4, 3, 2, 1, 1, 2, 2, 4, 3, 5, 3, 3, 1, 3, 2, 4, 1, 4, 2, 4~
## $ A9       <int> 4, 2, 2, 2, 2, 1, 1, 2, 2, 1, 2, 2, 2, 2, 3, 3, 1, 2, 1, 2, 2~
## $ A10      <int> 3, 2, 2, 2, 4, 4, 3, 2, 2, 3, 3, 1, 4, 4, 3, 2, 2, 1, 2, 3, 2~
## $ B1       <int> 4, 4, 4, 4, 2, 4, 5, 3, 5, 4, 4, 4, 3, 4, 3, 3, 3, 4, 4, 3, 3~
## $ B2       <int> 4, 4, 4, 2, 4, 1, 3, 3, 3, 4, 3, 3, 3, 5, 3, 4, 3, 3, 2, 3, 1~
## $ B3       <int> 5, 4, 5, 4, 4, 4, 5, 4, 5, 5, 5, 5, 4, 5, 3, 4, 5, 5, 4, 4, 4~
```

```
## $ B4       <int> 4, 4, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 3, 5, 3, 5, 5, 5, 5, 4, 4~
## $ B5       <int> 5, 5, 4, 4, 5, 3, 5, 4, 4, 5, 4, 4, 4, 5, 3, 4, 3, 5, 4, 4, 2~
## $ B6       <int> 4, 4, 4, 5, 4, 4, 3, 4, 4, 3, 4, 3, 4, 4, 3, 4, 4, 4, 4, 4, 4~
## $ B7       <int> 5, 3, 4, 4, 4, 3, 5, 4, 5, 4, 5, 4, 2, 5, 3, 3, 5, 4, 4, 4, 2~
## $ B8       <int> 4, 2, 4, 4, 5, 3, 5, 5, 4, 5, 4, 4, 4, 4, 3, 4, 5, 4, 4, 4, 4~
## $ B9       <int> 1, 3, 2, 3, 4, 2, 2, 4, 3, 2, 5, 5, 1, 3, 3, 4, 2, 1, 4, 4, 5~
## $ B10      <int> 2, 2, 2, 3, 1, 2, 2, 2, 4, 1, 2, 2, 2, 5, 3, 2, 3, 1, 2, 4, 3~
## $ B11      <int> 1, 4, 2, 3, 5, 1, 1, 3, 1, 2, 2, 3, 1, 2, 2, 4, 3, 2, 1, 3, 2~
## $ B12      <int> 1, 1, 2, 2, 1, 2, 1, 1, 2, 1, 4, 3, 1, 3, 3, 2, 4, 1, 2, 2, 2~
## $ B13      <int> 1, 1, 2, 4, 2, 2, 1, 1, 2, 2, 2, 4, 1, 2, 3, 1, 4, 1, 3, 4, 2~
## $ C1       <int> 4, 1, 2, 3, 2, 2, 1, 3, 4, 4, 2, 3, 3, 3, 3, 2, 3, 2, 5, 3, 4~
## $ C2       <int> 5, 2, 4, 2, 4, 3, 4, 4, 4, 4, 5, 5, 5, 4, 3, 2, 5, 5, 4, 4, 5~
## $ C3       <int> 4, 3, 4, 3, 3, 4, 4, 4, 2, 4, 3, 4, 4, 3, 3, 4, 4, 5, 4, 3, 4~
## $ C4       <int> 4, 3, 3, 4, 3, 2, 2, 4, 4, 4, 5, 4, 2, 5, 3, 2, 5, 4, 4, 3, 4~
## $ C5       <int> 2, 2, 3, 3, 4, 4, 3, 2, 4, 4, 5, 2, 1, 1, 3, 2, 5, 5, 4, 2, 4~
## $ C6       <int> 4, 5, 3, 3, 4, 2, 4, 2, 2, 2, 2, 2, 4, 3, 3, 4, 2, 1, 1, 2, 1~
## $ C7       <int> 4, 4, 4, 2, 4, 2, 5, 2, 0, 2, 2, 1, 2, 3, 3, 3, 4, 2, 1, 3, 1~
## $ C8       <int> 3, 4, 2, 2, 4, 4, 1, 3, 2, 1, 1, 1, 1, 3, 3, 4, 3, 1, 1, 3, 1~
## $ C9       <int> 3, 3, 3, 3, 3, 3, 2, 2, 2, 1, 2, 1, 1, 3, 3, 3, 2, 1, 2, 1, 1~
## $ C10      <int> 2, 3, 0, 4, 2, 1, 3, 3, 4, 2, 2, 0, 2, 5, 3, 4, 1, 1, 2, 1, 2~
## $ D1       <int> 4, 4, 3, 3, 5, 4, 4, 4, 2, 3, 2, 4, 5, 3, 3, 5, 2, 5, 4, 4, 4~
## $ D2       <int> 3, 2, 3, 2, 4, 4, 3, 3, 4, 3, 5, 3, 5, 3, 3, 5, 3, 4, 4, 3, 4~
## $ D3       <int> 5, 4, 2, 3, 3, 5, 3, 3, 2, 3, 4, 4, 4, 5, 3, 4, 5, 4, 3, 3, 4~
## $ D4       <int> 5, 4, 2, 4, 4, 5, 3, 3, 2, 4, 4, 4, 5, 5, 3, 4, 5, 4, 4, 3, 4~
## $ D5       <int> 4, 4, 3, 2, 5, 4, 4, 4, 4, 3, 3, 5, 5, 4, 2, 5, 3, 4, 3, 3, 4~
## $ D6       <int> 4, 5, 4, 3, 4, 3, 3, 4, 4, 4, 4, 3, 3, 4, 3, 4, 4, 4, 4, 3, 4~
## $ D7       <int> 3, 4, 3, 3, 1, 2, 3, 2, 4, 2, 3, 1, 1, 5, 3, 2, 2, 1, 2, 3, 2~
## $ D8       <int> 2, 2, 1, 3, 1, 2, 2, 2, 2, 2, 4, 1, 3, 1, 3, 2, 1, 1, 2, 2, 2~
## $ D9       <int> 4, 3, 3, 4, 1, 2, 3, 2, 3, 3, 3, 2, 2, 2, 3, 2, 4, 2, 1, 3, 3~
## $ D10      <int> 3, 1, 3, 3, 3, 3, 3, 3, 2, 2, 4, 1, 1, 5, 3, 2, 4, 2, 1, 3, 2~
## $ E1       <int> 1, 1, 1, 3, 1, 3, 1, 2, 1, 3, 3, 4, 3, 3, 3, 3, 4, 3, 4, 3, 5~
## $ E2       <int> 1, 2, 1, 2, 1, 4, 1, 2, 2, 4, 5, 3, 2, 5, 3, 3, 2, 3, 2, 2, 4~
## $ E3       <int> 4, 4, 3, 4, 3, 3, 2, 3, 5, 5, 4, 5, 4, 2, 3, 4, 5, 3, 5, 4, 5~
## $ E4       <int> 3, 1, 1, 1, 1, 2, 1, 1, 2, 3, 2, 4, 2, 4, 3, 3, 1, 3, 1, 3, 5~
## $ E5       <int> 4, 4, 4, 4, 4, 4, 1, 4, 4, 4, 4, 5, 4, 4, 3, 4, 5, 4, 4, 4, 4~
## $ E6       <int> 5, 2, 2, 4, 2, 3, 1, 1, 2, 3, 4, 4, 2, 1, 3, 3, 5, 1, 2, 3, 4~
## $ E7       <int> 1, 2, 3, 1, 4, 2, 2, 2, 4, 1, 4, 4, 1, 3, 3, 2, 1, 2, 1, 2, 2~
## $ E8       <int> 3, 4, 4, 3, 5, 3, 5, 5, 4, 2, 2, 2, 5, 2, 3, 5, 3, 3, 4, 3, 2~
## $ E9       <int> 3, 2, 2, 2, 2, 2, 3, 1, 2, 2, 2, 1, 2, 3, 3, 3, 2, 1, 2, 2, 2~
## $ E10      <int> 4, 1, 4, 2, 3, 4, 4, 4, 2, 2, 1, 1, 3, 2, 3, 2, 1, 3, 4, 3, 2~
## $ F1       <int> 2, 3, 3, 4, 4, 2, 3, 3, 4, 3, 5, 2, 2, 1, 3, 3, 2, 4, 5, 3, 2~
## $ F2       <int> 2, 4, 4, 3, 4, 3, 4, 5, 4, 4, 4, 4, 3, 2, 3, 5, 1, 5, 5, 4, 4~
## $ F3       <int> 5, 2, 1, 1, 4, 2, 1, 5, 1, 5, 5, 5, 1, 1, 3, 2, 1, 2, 5, 1, 5~
## $ F4       <int> 1, 4, 2, 3, 4, 2, 4, 4, 5, 5, 5, 4, 3, 2, 3, 5, 1, 4, 5, 5, 5~
## $ F5       <int> 1, 5, 3, 2, 4, 1, 2, 5, 5, 5, 5, 5, 3, 1, 3, 3, 2, 5, 5, 4, 4~
## $ F6       <int> 4, 2, 3, 4, 2, 4, 1, 2, 1, 1, 3, 2, 3, 2, 3, 2, 5, 2, 1, 2, 3~
## $ F7       <int> 3, 4, 3, 3, 4, 4, 2, 4, 4, 3, 2, 4, 2, 3, 3, 3, 5, 1, 1, 2, 4~
## $ F8       <int> 5, 4, 2, 4, 4, 5, 3, 4, 2, 1, 4, 4, 4, 2, 3, 3, 5, 1, 3, 5, 4~
## $ F9       <int> 4, 2, 3, 3, 1, 4, 1, 2, 2, 2, 2, 2, 3, 4, 3, 2, 3, 2, 1, 2, 3~
## $ F10      <int> 4, 4, 4, 3, 3, 3, 2, 4, 2, 3, 3, 2, 2, 4, 3, 4, 1, 1, 4, 3, 2~
## $ G1       <int> 3, 3, 2, 2, 3, 3, 3, 4, 4, 4, 5, 4, 4, 4, 3, 4, 4, 3, 5, 4, 4~
## $ G2       <int> 1, 4, 2, 4, 1, 4, 1, 3, 2, 4, 5, 4, 2, 4, 3, 3, 3, 3, 4, 4, 4~
## $ G3       <int> 4, 2, 1, 4, 4, 4, 4, 2, 4, 3, 3, 4, 5, 3, 3, 4, 4, 4, 4, 4, 4~
## $ G4       <int> 2, 4, 3, 4, 2, 4, 3, 2, 3, 5, 4, 5, 4, 4, 3, 3, 5, 3, 5, 4, 4~
```

```
## $ G5       <int> 3, 4, 3, 3, 5, 4, 3, 4, 4, 4, 3, 4, 4, 4, 3, 4, 5, 4, 5, 4, 4~
## $ G6       <int> 4, 2, 3, 3, 2, 2, 4, 4, 2, 2, 4, 1, 2, 2, 3, 2, 1, 1, 1, 2, 2~
## $ G7       <int> 3, 3, 3, 3, 1, 4, 3, 3, 2, 1, 2, 1, 1, 4, 3, 3, 2, 1, 1, 2, 4~
## $ G8       <int> 1, 2, 3, 2, 1, 2, 2, 2, 2, 2, 4, 2, 1, 1, 3, 2, 1, 1, 2, 2, 3~
## $ G9       <int> 5, 3, 5, 3, 2, 4, 4, 3, 4, 2, 4, 2, 2, 4, 3, 3, 3, 3, 2, 3, 4~
## $ G10      <int> 1, 3, 4, 3, 2, 2, 4, 3, 4, 3, 3, 2, 1, 3, 2, 4, 4, 3, 2, 4, 4~
## $ H1       <int> 5, 5, 3, 4, 4, 5, 3, 5, 4, 3, 2, 1, 4, 5, 3, 4, 4, 5, 2, 2, 3~
## $ H2       <int> 5, 5, 4, 5, 4, 5, 5, 4, 4, 4, 4, 4, 4, 4, 4, 3, 4, 5, 5, 4, 3, 4~
## $ H3       <int> 5, 5, 3, 2, 3, 4, 3, 5, 4, 2, 2, 2, 4, 3, 3, 4, 4, 3, 1, 1, 2~
## $ H4       <int> 3, 1, 4, 1, 3, 2, 2, 5, 1, 1, 1, 3, 4, 3, 3, 4, 2, 3, 1, 2, 1~
## $ H5       <int> 3, 4, 1, 4, 1, 4, 3, 4, 2, 1, 2, 3, 4, 4, 3, 3, 2, 2, 2, 4, 2~
## $ H6       <int> 3, 3, 4, 2, 4, 4, 2, 5, 4, 3, 3, 2, 3, 5, 3, 2, 4, 4, 4, 4, 4~
## $ H7       <int> 3, 4, 2, 3, 2, 1, 2, 2, 2, 2, 2, 2, 1, 1, 3, 3, 1, 3, 3, 1, 2~
## $ H8       <int> 1, 2, 3, 2, 4, 1, 3, 2, 1, 3, 2, 2, 2, 3, 3, 4, 1, 3, 2, 3, 1~
## $ H9       <int> 2, 1, 0, 2, 1, 1, 1, 5, 2, 2, 2, 3, 1, 5, 3, 5, 2, 1, 1, 3, 3~
## $ H10      <int> 4, 2, 2, 2, 2, 2, 1, 2, 1, 1, 4, 2, 1, 5, 3, 3, 1, 1, 2, 2, 4~
## $ I1       <int> 3, 5, 3, 2, 4, 4, 4, 3, 2, 2, 4, 1, 4, 2, 3, 4, 1, 1, 1, 2, 2~
## $ I2       <int> 4, 4, 4, 2, 2, 4, 2, 4, 4, 3, 3, 2, 2, 4, 3, 5, 4, 4, 3, 3, 2~
## $ I3       <int> 4, 3, 4, 4, 4, 2, 3, 2, 4, 3, 4, 3, 2, 4, 3, 5, 5, 3, 3, 3, 3~
## $ I4       <int> 3, 2, 3, 3, 4, 3, 2, 2, 2, 3, 3, 2, 3, 4, 3, 4, 5, 2, 3, 2, 2~
## $ I5       <int> 4, 4, 4, 4, 2, 4, 2, 4, 4, 4, 5, 4, 5, 2, 3, 4, 5, 3, 3, 3, 4~
## $ I6       <int> 4, 2, 2, 2, 2, 1, 1, 2, 2, 1, 1, 1, 2, 3, 3, 3, 3, 3, 1, 1, 2~
## $ I7       <int> 2, 2, 2, 4, 3, 4, 4, 3, 4, 4, 4, 4, 2, 3, 3, 2, 3, 3, 3, 4, 2~
## $ I8       <int> 4, 2, 3, 2, 2, 4, 4, 3, 5, 4, 4, 5, 2, 4, 2, 2, 5, 3, 4, 4, 4~
## $ I9       <int> 2, 3, 4, 4, 3, 4, 3, 4, 5, 4, 4, 4, 2, 4, 3, 2, 4, 4, 5, 4, 4~
## $ I10      <int> 4, 2, 4, 1, 3, 4, 2, 4, 4, 3, 4, 4, 3, 5, 3, 2, 4, 3, 2, 3, 4~
## $ J1       <int> 5, 5, 4, 5, 3, 3, 1, 2, 4, 3, 4, 4, 4, 1, 3, 2, 5, 3, 2, 4, 4~
## $ J2       <int> 5, 4, 4, 5, 4, 3, 5, 4, 2, 3, 2, 3, 4, 5, 3, 4, 5, 3, 4, 4, 3~
## $ J3       <int> 5, 5, 4, 5, 2, 3, 4, 1, 2, 3, 3, 3, 3, 2, 3, 1, 5, 4, 4, 4, 3~
## $ J4       <int> 5, 4, 5, 5, 4, 4, 4, 2, 4, 3, 4, 4, 4, 4, 3, 2, 5, 5, 5, 4, 4~
## $ J5       <int> 5, 4, 5, 5, 4, 4, 3, 2, 2, 3, 4, 3, 3, 3, 3, 4, 5, 3, 3, 4, 2~
## $ J6       <int> 5, 2, 3, 4, 1, 3, 2, 1, 2, 4, 3, 3, 3, 4, 3, 3, 5, 2, 2, 3, 2~
## $ J7       <int> 5, 4, 3, 4, 2, 4, 3, 2, 4, 3, 4, 3, 4, 4, 3, 4, 5, 3, 3, 4, 4~
## $ J8       <int> 2, 3, 3, 1, 5, 2, 3, 4, 4, 4, 4, 3, 2, 3, 3, 4, 1, 2, 5, 3, 2~
## $ J9       <int> 1, 2, 1, 1, 4, 3, 1, 4, 4, 2, 4, 4, 1, 5, 3, 4, 1, 1, 1, 2, 3~
## $ J10      <int> 1, 2, 1, 1, 2, 3, 1, 2, 4, 2, 4, 4, 1, 5, 3, 3, 1, 2, 3, 3, 3~
## $ K1       <int> 2, 3, 3, 3, 4, 4, 1, 4, 2, 3, 4, 1, 1, 2, 3, 3, 1, 2, 3, 2, 4~
## $ K2       <int> 3, 4, 4, 3, 4, 4, 2, 4, 2, 1, 2, 3, 2, 5, 3, 3, 5, 2, 1, 3, 1~
## $ K3       <int> 2, 2, 4, 1, 1, 3, 3, 2, 2, 3, 4, 2, 1, 2, 3, 2, 2, 3, 2, 2, 4~
## $ K4       <int> 4, 4, 4, 4, 4, 2, 2, 4, 4, 3, 5, 2, 2, 0, 3, 4, 1, 2, 2, 3, 4~
## $ K5       <int> 4, 3, 4, 3, 4, 3, 3, 4, 4, 3, 4, 1, 2, 2, 3, 4, 1, 2, 2, 2, 4~
## $ K6       <int> 5, 2, 3, 2, 2, 4, 4, 4, 4, 3, 5, 4, 5, 5, 3, 2, 5, 4, 4, 4, 4~
## $ K7       <int> 2, 2, 2, 2, 3, 3, 4, 4, 4, 2, 2, 4, 5, 5, 3, 2, 5, 3, 4, 4, 2~
## $ K8       <int> 5, 2, 2, 2, 2, 3, 4, 2, 4, 4, 4, 4, 4, 5, 3, 1, 5, 2, 3, 3, 3~
## $ K9       <int> 2, 3, 3, 3, 2, 5, 4, 4, 4, 1, 2, 4, 5, 5, 3, 2, 5, 3, 4, 4, 4~
## $ K10      <int> 5, 4, 4, 3, 4, 4, 5, 4, 4, 3, 3, 3, 5, 4, 3, 4, 5, 3, 4, 4, 4~
## $ L1       <int> 3, 3, 4, 3, 4, 2, 4, 4, 4, 3, 4, 2, 3, 4, 3, 5, 3, 1, 1, 4, 4~
## $ L2       <int> 4, 4, 3, 2, 2, 2, 2, 4, 2, 2, 3, 1, 3, 5, 3, 4, 2, 1, 2, 2, 2~
## $ L3       <int> 3, 4, 3, 4, 4, 5, 4, 3, 2, 2, 4, 2, 4, 5, 3, 3, 4, 1, 3, 4, 2~
## $ L4       <int> 3, 4, 4, 3, 4, 4, 5, 4, 4, 3, 4, 3, 5, 5, 3, 4, 4, 1, 4, 4, 4~
## $ L5       <int> 5, 2, 4, 4, 4, 4, 4, 4, 4, 4, 3, 5, 3, 2, 4, 3, 5, 2, 1, 4, 4, 4~
## $ L6       <int> 3, 1, 4, 4, 4, 4, 5, 5, 2, 2, 4, 2, 3, 5, 3, 4, 5, 4, 3, 4, 2~
## $ L7       <int> 4, 4, 2, 4, 4, 2, 4, 3, 4, 2, 4, 4, 2, 3, 3, 5, 3, 1, 2, 3, 5~
## $ L8       <int> 4, 2, 3, 4, 1, 1, 1, 3, 2, 3, 2, 4, 4, 0, 3, 2, 4, 5, 4, 2, 2~
```

```
## $ L9       <int> 2, 2, 3, 3, 3, 4, 2, 2, 2, 4, 4, 4, 2, 3, 3, 2, 5, 5, 3, 2, 4~
## $ L10      <int> 4, 3, 4, 2, 3, 3, 3, 3, 4, 4, 4, 2, 3, 2, 3, 2, 3, 5, 4, 3, 4~
## $ M1       <int> 5, 4, 3, 4, 2, 4, 4, 4, 4, 3, 3, 4, 5, 5, 3, 2, 4, 4, 4, 4, 4~
## $ M2       <int> 4, 2, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 2, 5, 3, 3, 5, 4, 5, 4, 4~
## $ M3       <int> 4, 4, 5, 5, 4, 4, 5, 4, 4, 4, 5, 4, 4, 5, 3, 4, 5, 5, 5, 4, 4~
## $ M4       <int> 4, 4, 4, 3, 4, 3, 4, 4, 2, 4, 2, 3, 3, 4, 3, 4, 3, 5, 4, 3, 2~
## $ M5       <int> 4, 2, 4, 2, 5, 4, 4, 4, 4, 3, 4, 3, 1, 5, 3, 1, 5, 3, 3, 3, 4~
## $ M6       <int> 1, 1, 2, 1, 1, 2, 2, 2, 2, 2, 3, 2, 2, 1, 3, 3, 1, 1, 2, 3, 2~
## $ M7       <int> 1, 1, 1, 4, 2, 5, 1, 1, 1, 2, 4, 2, 2, 5, 3, 3, 1, 1, 1, 2, 2~
## $ M8       <int> 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 4, 3, 3, 1, 3, 3, 1, 1, 2, 3, 2~
## $ M9       <int> 3, 2, 1, 2, 1, 2, 1, 1, 2, 2, 2, 3, 3, 1, 3, 4, 1, 1, 2, 3, 2~
## $ M10      <int> 3, 2, 2, 2, 2, 1, 2, 2, 2, 2, 3, 2, 2, 3, 3, 2, 1, 2, 2, 2, 1~
## $ N1       <int> 4, 4, 4, 4, 3, 4, 2, 4, 2, 3, 4, 2, 2, 3, 3, 3, 4, 2, 1, 2, 3~
## $ N2       <int> 4, 2, 4, 5, 3, 4, 4, 3, 2, 2, 5, 4, 4, 4, 3, 5, 3, 4, 3, 3, 2~
## $ N3       <int> 4, 4, 5, 5, 4, 5, 4, 4, 4, 3, 4, 2, 4, 5, 3, 4, 5, 2, 2, 4, 4~
## $ N4       <int> 4, 5, 4, 5, 4, 4, 4, 4, 2, 3, 4, 1, 2, 4, 3, 4, 2, 4, 2, 3, 2~
## $ N5       <int> 4, 5, 4, 5, 5, 4, 2, 4, 4, 4, 4, 5, 2, 4, 3, 4, 5, 1, 1, 4, 4~
## $ N6       <int> 5, 5, 5, 5, 4, 5, 3, 5, 4, 4, 4, 2, 1, 5, 3, 4, 5, 4, 3, 4, 3~
## $ N7       <int> 5, 5, 4, 5, 5, 4, 5, 5, 4, 4, 5, 4, 4, 4, 3, 4, 5, 5, 4, 4, 4~
## $ N8       <int> 4, 2, 3, 1, 4, 4, 2, 3, 0, 4, 4, 3, 4, 4, 3, 4, 3, 4, 4, 4, 4~
## $ N9       <int> 4, 4, 4, 3, 4, 4, 2, 4, 4, 4, 5, 4, 2, 2, 3, 4, 5, 4, 5, 4, 4~
## $ N10      <int> 3, 1, 1, 3, 2, 2, 2, 2, 4, 1, 2, 3, 3, 1, 2, 4, 1, 1, 4, 2, 2~
## $ O1       <int> 4, 3, 3, 2, 3, 2, 4, 4, 4, 4, 4, 4, 5, 4, 3, 5, 4, 3, 4, 4, 4~
## $ O2       <int> 2, 2, 2, 2, 3, 1, 3, 3, 4, 2, 5, 2, 2, 4, 3, 2, 5, 4, 3, 3, 2~
## $ O3       <int> 3, 4, 3, 2, 5, 4, 4, 4, 4, 4, 5, 4, 4, 4, 3, 4, 4, 4, 5, 3, 3~
## $ O4       <int> 4, 2, 3, 2, 3, 4, 4, 4, 5, 4, 4, 4, 4, 5, 3, 5, 2, 4, 3, 4, 4~
## $ O5       <int> 3, 3, 3, 3, 4, 4, 4, 4, 4, 4, 4, 3, 4, 4, 3, 5, 4, 4, 4, 3, 3~
## $ O6       <int> 2, 2, 2, 4, 2, 4, 3, 4, 4, 1, 2, 4, 1, 4, 3, 4, 3, 3, 2, 4, 4~
## $ O7       <int> 2, 2, 3, 5, 1, 2, 3, 2, 4, 3, 4, 4, 2, 5, 3, 4, 5, 1, 2, 3, 3~
## $ O8       <int> 4, 3, 4, 4, 2, 5, 3, 4, 4, 1, 2, 4, 2, 4, 3, 4, 2, 1, 2, 4, 2~
## $ O9       <int> 4, 4, 3, 5, 1, 3, 4, 4, 4, 2, 2, 4, 2, 4, 3, 4, 4, 1, 4, 4, 2~
## $ O10      <int> 4, 4, 3, 4, 4, 4, 3, 3, 4, 2, 2, 2, 4, 4, 3, 4, 2, 1, 2, 3, 4~
## $ P1       <int> 5, 4, 3, 4, 4, 2, 2, 2, 2, 2, 3, 1, 3, 4, 3, 4, 2, 1, 3, 2, 2~
## $ P2       <int> 5, 4, 3, 2, 3, 4, 2, 3, 2, 1, 4, 2, 4, 4, 3, 5, 2, 2, 1, 2, 2~
## $ P3       <int> 5, 4, 4, 2, 4, 5, 4, 2, 1, 1, 3, 1, 4, 5, 3, 4, 2, 4, 1, 3, 2~
## $ P4       <int> 4, 2, 2, 2, 3, 2, 4, 2, 2, 3, 4, 1, 4, 5, 3, 4, 5, 2, 3, 3, 3~
## $ P5       <int> 4, 3, 4, 3, 4, 3, 2, 3, 2, 2, 3, 3, 2, 4, 3, 3, 4, 1, 1, 3, 2~
## $ P6       <int> 5, 2, 2, 4, 3, 3, 2, 2, 3, 3, 4, 4, 4, 4, 3, 2, 1, 1, 4, 3, 4~
## $ P7       <int> 1, 3, 2, 2, 4, 4, 4, 1, 2, 2, 4, 2, 4, 4, 3, 4, 1, 1, 3, 3, 3~
## $ P8       <int> 2, 2, 3, 3, 2, 4, 2, 4, 4, 4, 3, 4, 2, 3, 3, 2, 5, 5, 4, 3, 4~
## $ P9       <int> 5, 2, 3, 4, 3, 2, 3, 4, 4, 4, 4, 4, 2, 5, 2, 2, 3, 4, 4, 4, 4~
## $ P10      <int> 2, 2, 3, 4, 2, 5, 4, 3, 4, 3, 5, 4, 2, 5, 3, 2, 5, 5, 5, 4, 4~
## $ age      <int> 17, 37, 31, 32, 46, 36, 35, 61, 17, 19, 21, 23, 17, 24, 21, 3~
## $ gender   <int> 1, 1, 1, 1, 2, 2, 1, 2, 2, 1, 1, 2, 2, 2, 2, 2, 2, 1, 1, 2, 1~
## $ accuracy <int> 92, 100, 80, 93, 87, 80, 80, 100, 78, 95, 95, 82, 90, 90, 54,~
## $ country  <chr> "US", "US", "US", "US", "NZ", "IT", "US", "US", "US", "US", "~
## $ source   <int> 6, 1, 6, 1, 1, 3, 1, 1, 6, 6, 6, 6, 1, 1, 6, 1, 5, 1, 6, 6, 6~
## $ elapsed  <int> 914, 891, 903, 806, 1826, 669, 717, 567, 651, 1835, 770, 922,~
## $ Amean    <dbl> 2.9, 3.4, 3.4, 3.3, 3.1, 3.6, 3.1, 3.6, 3.5, 3.6, 4.3, 3.8, 3~
## $ Bmean    <dbl> 3.153846, 3.153846, 3.384615, 3.615385, 3.538462, 2.769231, 3~
## $ Cmean    <dbl> 3.5, 3.0, 2.8, 2.9, 3.3, 2.7, 2.9, 2.9, 2.8, 2.8, 2.9, 2.3, 2~
## $ Dmean    <dbl> 3.7, 3.3, 2.7, 3.0, 3.1, 3.4, 3.1, 3.0, 2.9, 2.9, 3.6, 2.8, 3~
## $ Emean    <dbl> 2.9, 2.3, 2.5, 2.6, 2.6, 3.0, 2.1, 2.5, 2.8, 2.9, 3.1, 3.3, 2~
## $ Fmean    <dbl> 3.1, 3.4, 2.8, 3.0, 3.4, 3.0, 2.3, 3.8, 3.0, 3.2, 3.8, 3.4, 2~
```

```
## $ Gmean    <dbl> 2.7, 3.0, 2.9, 3.1, 2.3, 3.3, 3.1, 3.0, 3.1, 3.0, 3.7, 2.9, 2~
## $ Hmean    <dbl> 3.4, 3.2, 2.6, 2.7, 2.8, 2.9, 2.5, 3.9, 2.5, 2.2, 2.4, 2.4, 2~
## $ Imean    <dbl> 3.4, 2.9, 3.3, 2.8, 2.9, 3.4, 2.7, 3.1, 3.6, 3.1, 3.6, 3.0, 2~
## $ Jmean    <dbl> 3.9, 3.5, 3.3, 3.6, 3.1, 3.2, 2.7, 2.4, 3.2, 3.0, 3.6, 3.4, 2~
## $ Kmean    <dbl> 3.4, 2.9, 3.3, 2.6, 3.0, 3.5, 3.2, 3.6, 3.4, 2.6, 3.5, 2.8, 3~
## $ Lmean    <dbl> 3.5, 2.9, 3.4, 3.3, 3.3, 3.1, 3.4, 3.5, 3.0, 2.8, 3.8, 2.7, 3~
## $ Mmean    <dbl> 3.000000, 2.333333, 2.888889, 3.000000, 2.777778, 3.333333, 3~
## $ Nmean    <dbl> 4.1, 3.7, 3.8, 4.1, 3.8, 4.0, 3.0, 3.8, 3.0, 3.2, 4.1, 3.0, 2~
## $ Omean    <dbl> 3.2, 2.9, 2.9, 3.3, 2.8, 3.3, 3.5, 3.6, 4.1, 2.7, 3.4, 3.5, 3~
## $ Pmean    <dbl> 3.8, 2.8, 2.9, 3.0, 3.2, 3.4, 2.9, 2.6, 2.6, 2.5, 3.7, 2.6, 3~
```

Vignette Info

Note the various macros within the `vignette` section of the metadata block above. These are required in order to instruct R how to build the vignette. Note that you should change the `title` field and the `\VignetteIndexEntry` to match the title of your vignette.

##What is ggplot2 and provide a couple of basic examples of the use of scatter plots and an example of a box plot?

Details on ggplot2 can be found at the following link: https://github.com/tidyverse/ggplot2

Cheat sheet for GGPLOT2 https://github.com/rstudio/cheatsheets/blob/master/data-visualization-2.1.pdf

A short summary of the basic elements explored below are as follows described and from the original site on ggplot in tidyverse:

"It's hard to succinctly describe how ggplot2 works because it embodies a deep philosophy of visualisation. However, in most cases you start with ggplot(), supply a dataset and aesthetic mapping (with aes()). You then add on layers (like geom_point() or geom_histogram()), scales (like scale_colour_brewer()), faceting specifications (like facet_wrap()) and coordinate systems (like coord_flip())."

ggplot2 is a vast ecosystem of visualization tools. This will focus on a couple of basic plots to provide exposured to ggplot. The scatter plot and a box plot.

#Part 1: This is basic scatter plot format. pfdf is the dataframe and the x and y values are the two axis. geom_point ensures we plot a point and you can set the size and also the shape. There are no axis limits, no tittles and no special features.

"ggplot(pfdf,aes(y=elapsed, x=age))+geom_point(size=2)"

Try this to see what happens.

#Part 2: Adjust your basic scatter plot to control the axis ranges.

In this case you can and should add a limit to the respondents age. (over 100 years old is likely too high and also a time limit in seconds which is the y axis in this case) Adding limits is preferred in exploring the data as it does not change the dataset to start but makes it more useable or transparent what is happening with the data.

ggplot(pfdf,aes(y=elapsed, x=age))+geom_point(size=2)+ylim(0,250000)+xlim(0,100)

#Part 3: Extending the scatter plot to include more features with a layer(color)

You can then add another feature to the graph by adding a color feature. In this case we added gender. We also decided to look at another variable the accuracy of the assessment which we limited to 100% max accuracy and 0% min accuracy.

We still have no graph title included yet.

ggplot(pfdf,aes(x=accuracy, y=elapsed, color=gender))+geom_point(size=2)+xlim(0,100)+ylim(0,250000)

#Part 4: Extend to add titles and change to a box plot

You can now add a title to the graph as shown below with ggtitle and change to a boxplot of a variable with geom_boxplot() (instead of geom_point())

ggplot(pfdf,aes(x=gender,y=Imean))+geom_boxplot()+ggtitle("Average Vigilance boxplot")

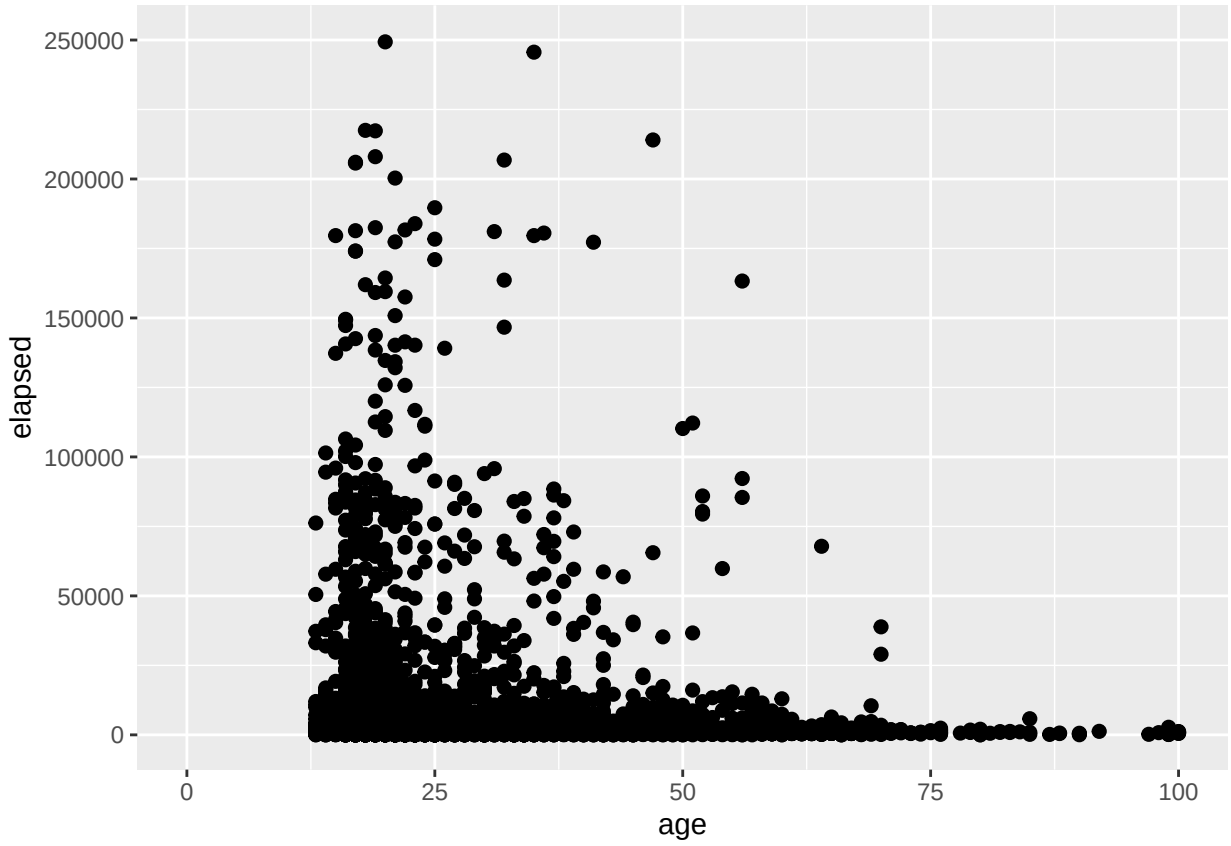
Also a included is the simple first pass attempt to overlap the country into the mix. It appears too messy to be of much value. So, we need to rethink this and look for an approach. What do you see as the next potential approach to analyze by country and would you expect this to be important? Likley the country analysis or grouping of the countrys by specific cultural indicators and environment would be interesting.

A next step is to plot individual country scores and evaluate this or to group by similar country culture and demographic profiles to see if there are statistically significant differences.

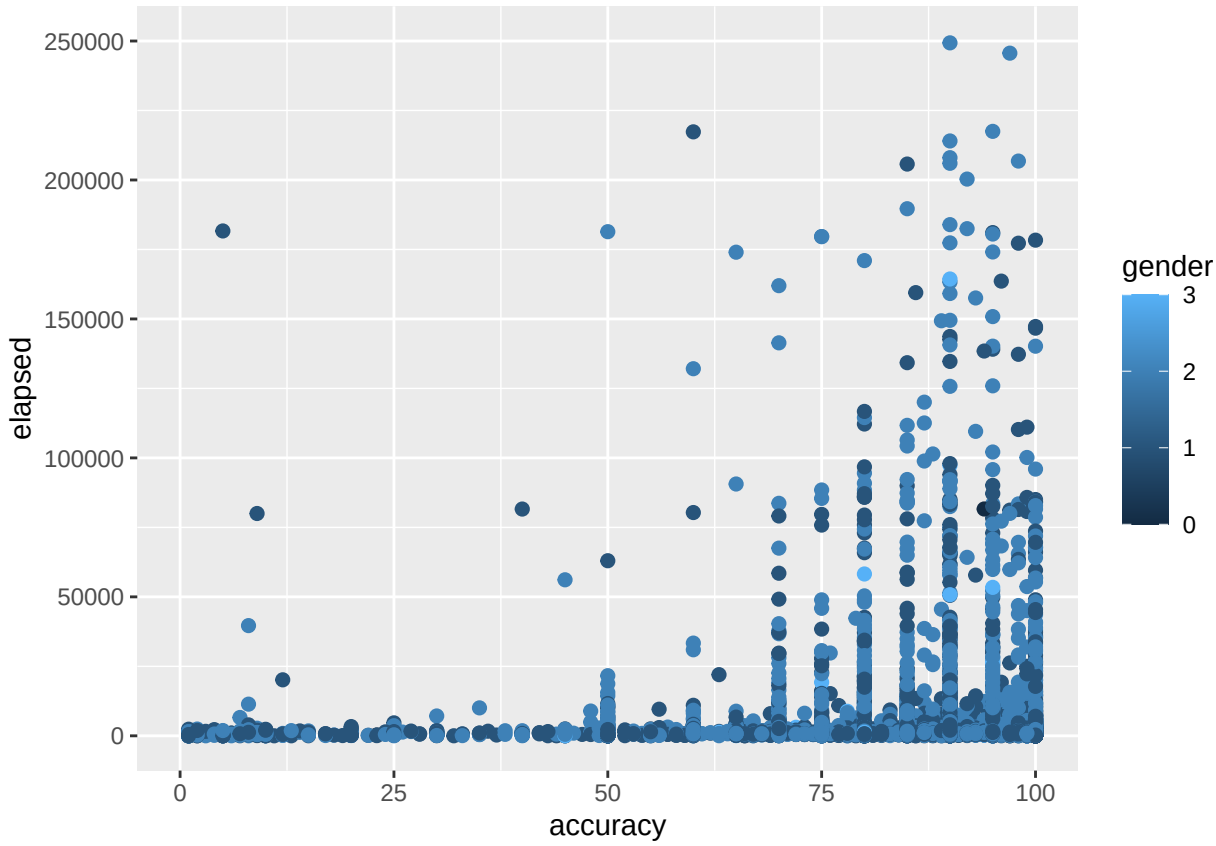
Figures for simple GGPLOT2 demonstration of scatter plot and a box plot

The figure sizes have been customised so that you can easily put two images side-by-side.

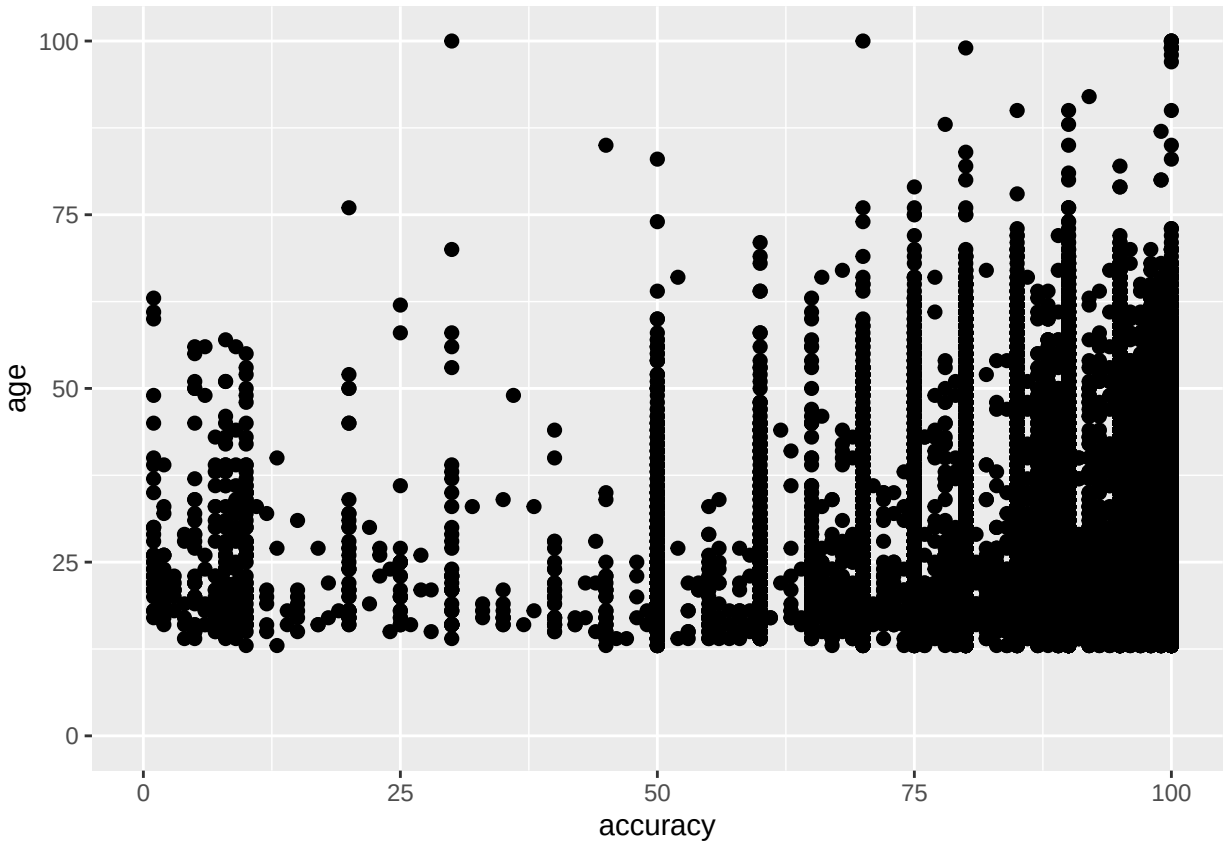
```
## Warning: Removed 130 rows containing missing values (geom_point).
```



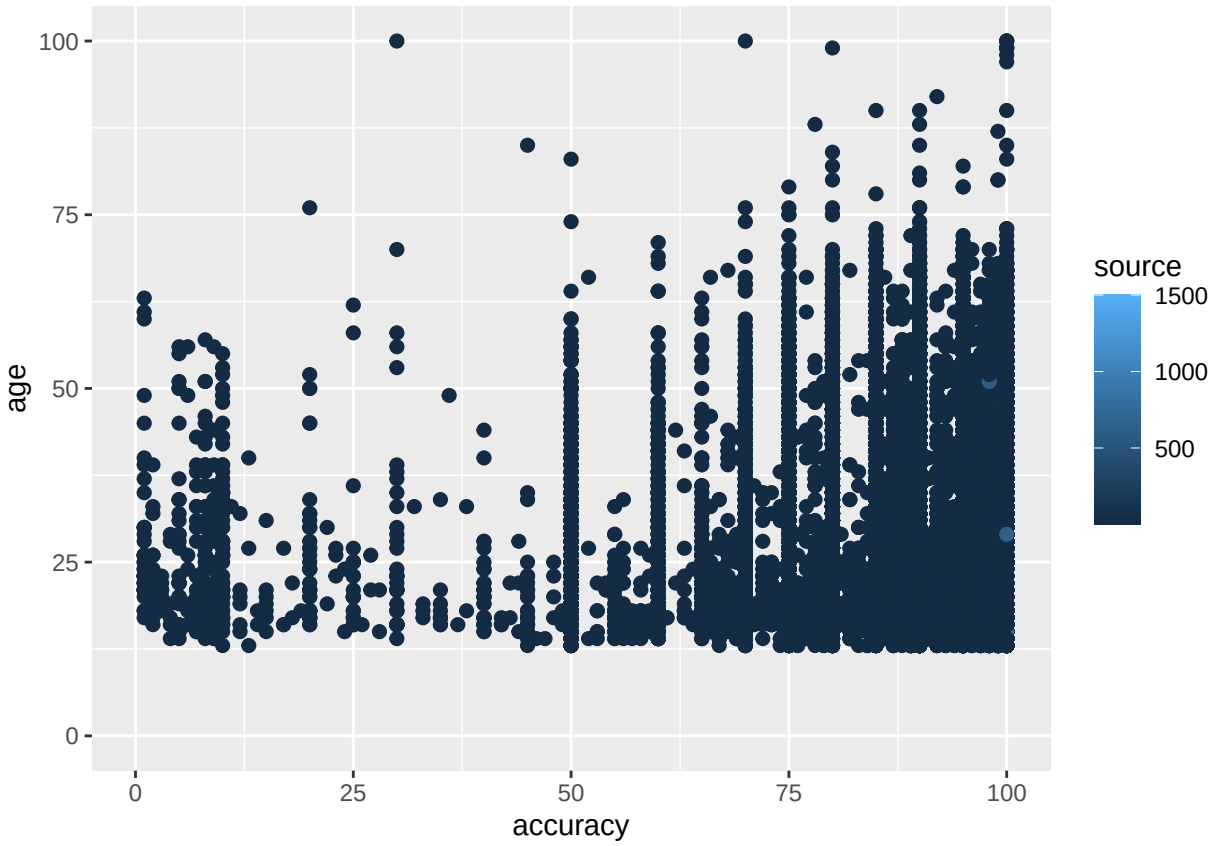

```
## Warning: Removed 85 rows containing missing values (geom_point).
```



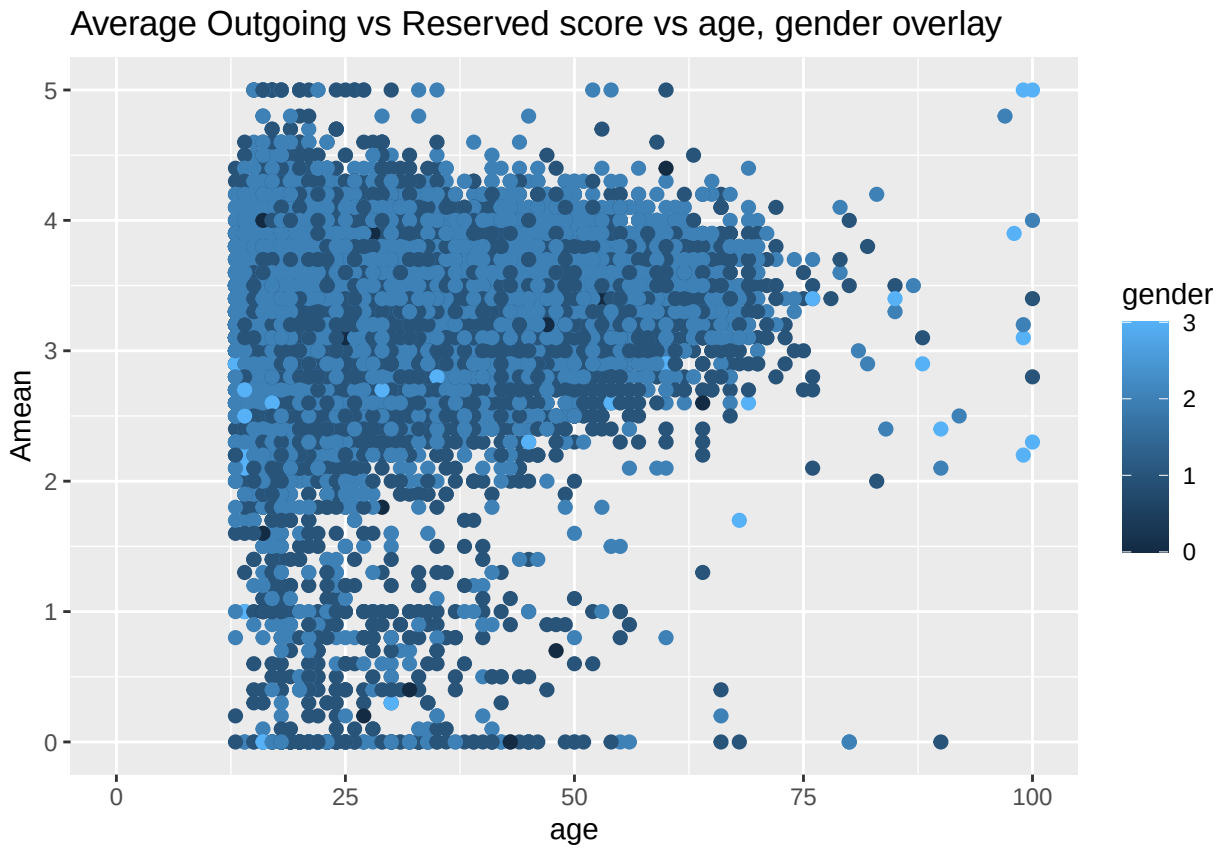

```
## Warning: Removed 90 rows containing missing values (geom_point).
```



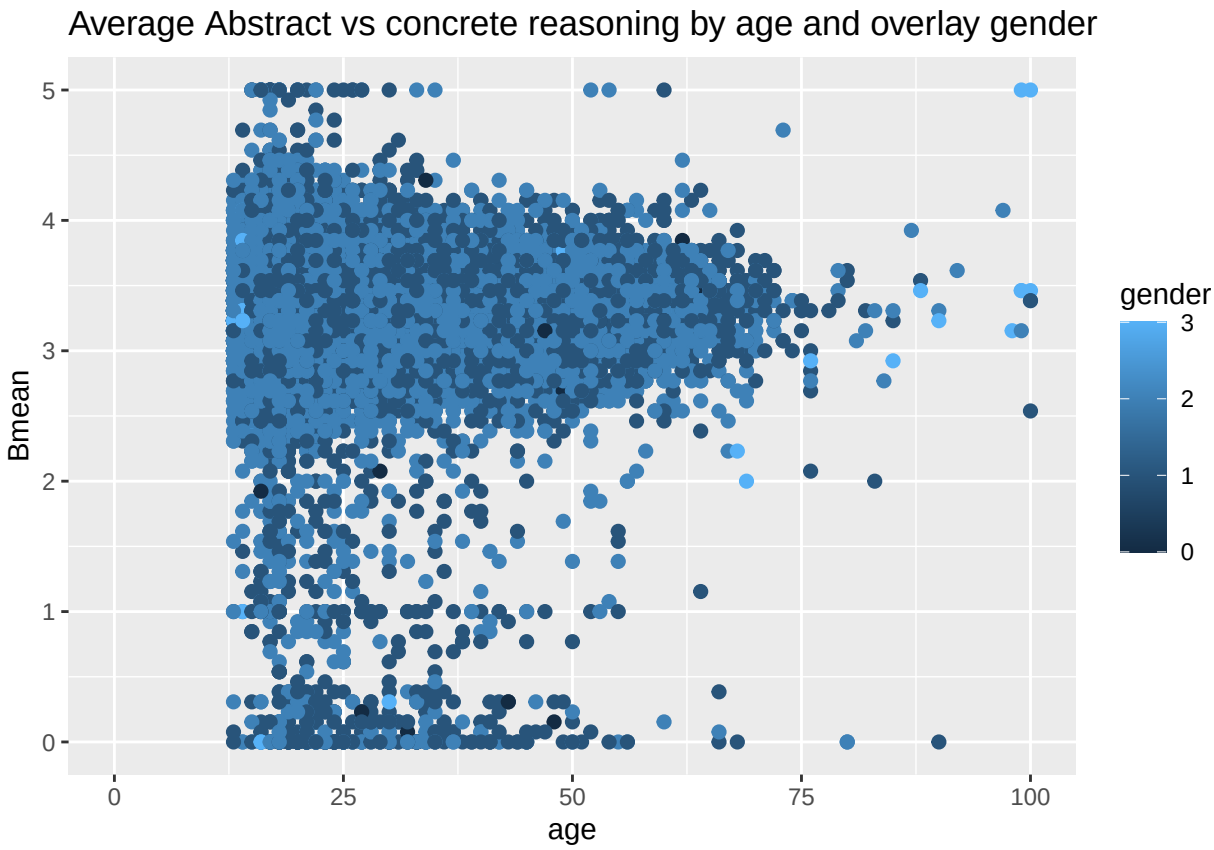

```
## Warning: Removed 90 rows containing missing values (geom_point).
```



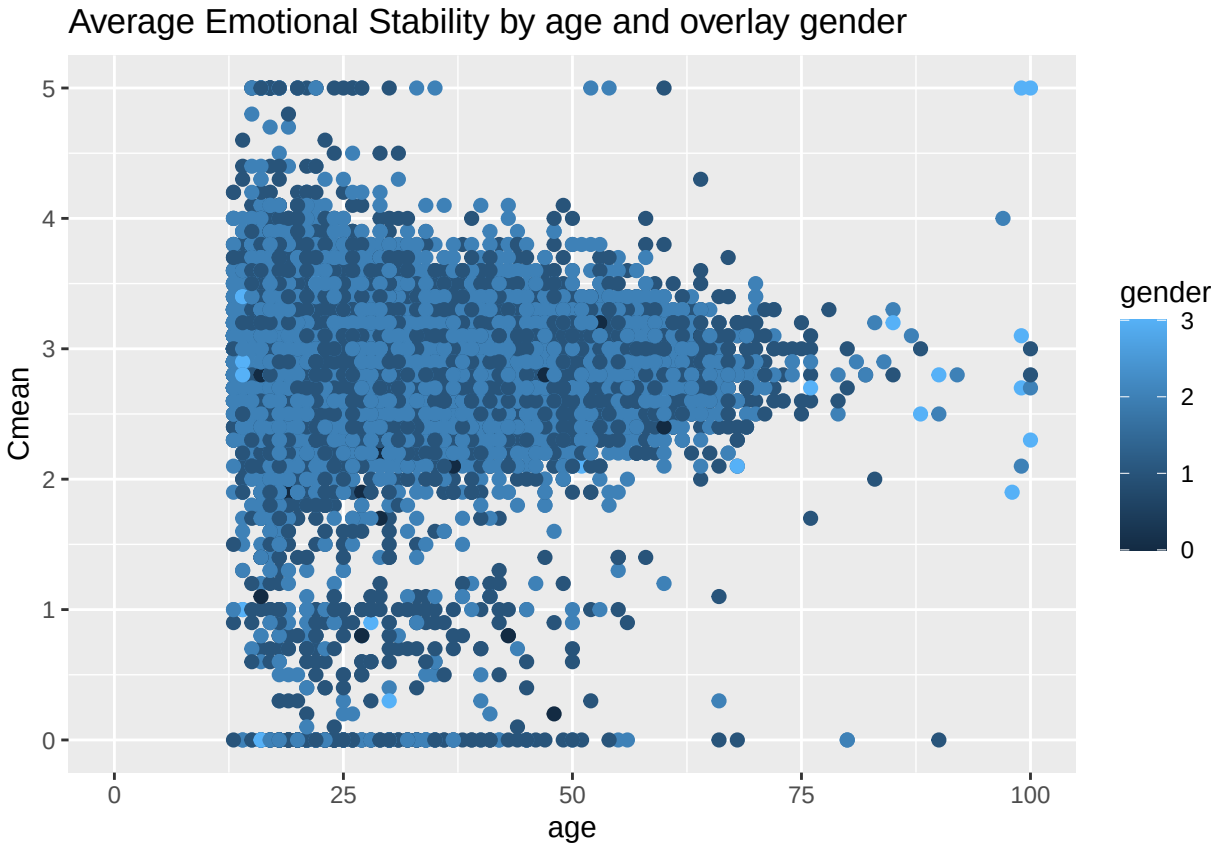

```
## Warning: Removed 68 rows containing missing values (geom_point).
```



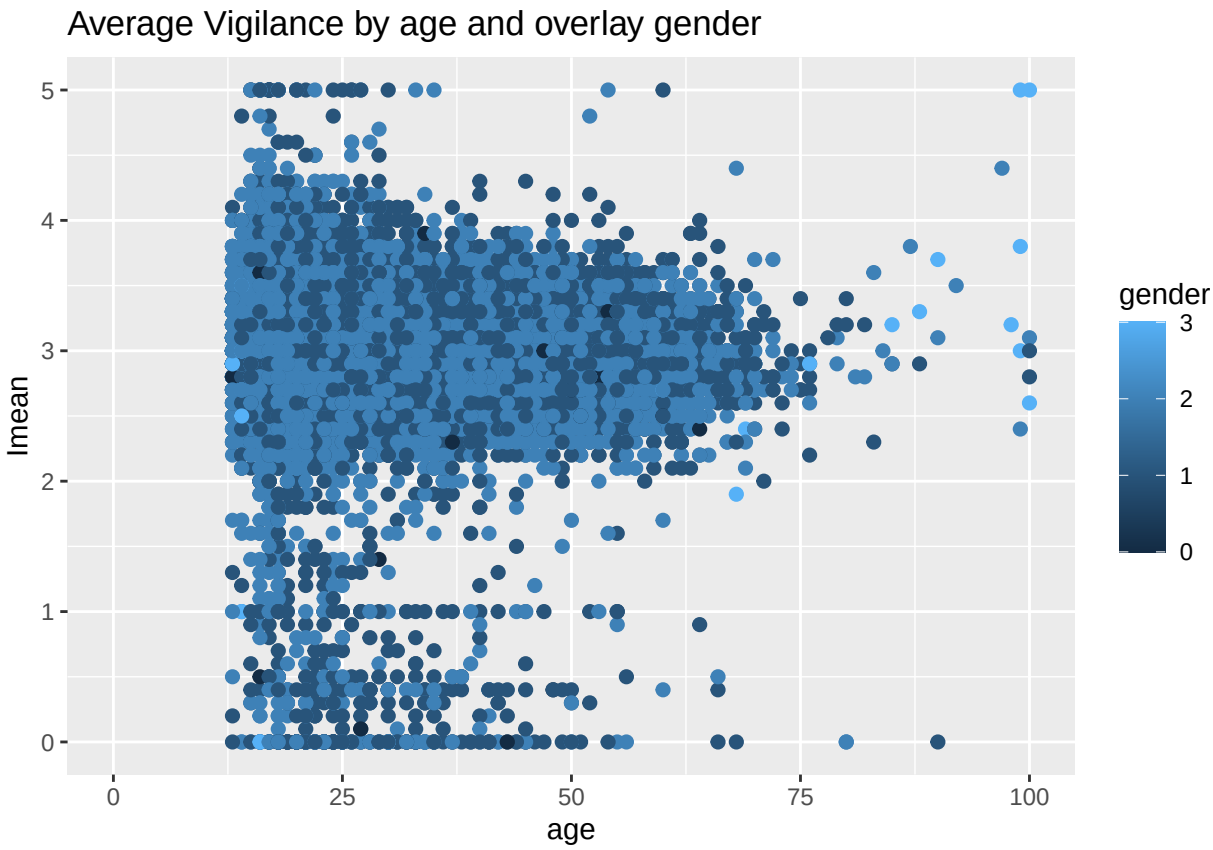

```
## Warning: Removed 68 rows containing missing values (geom_point).
```



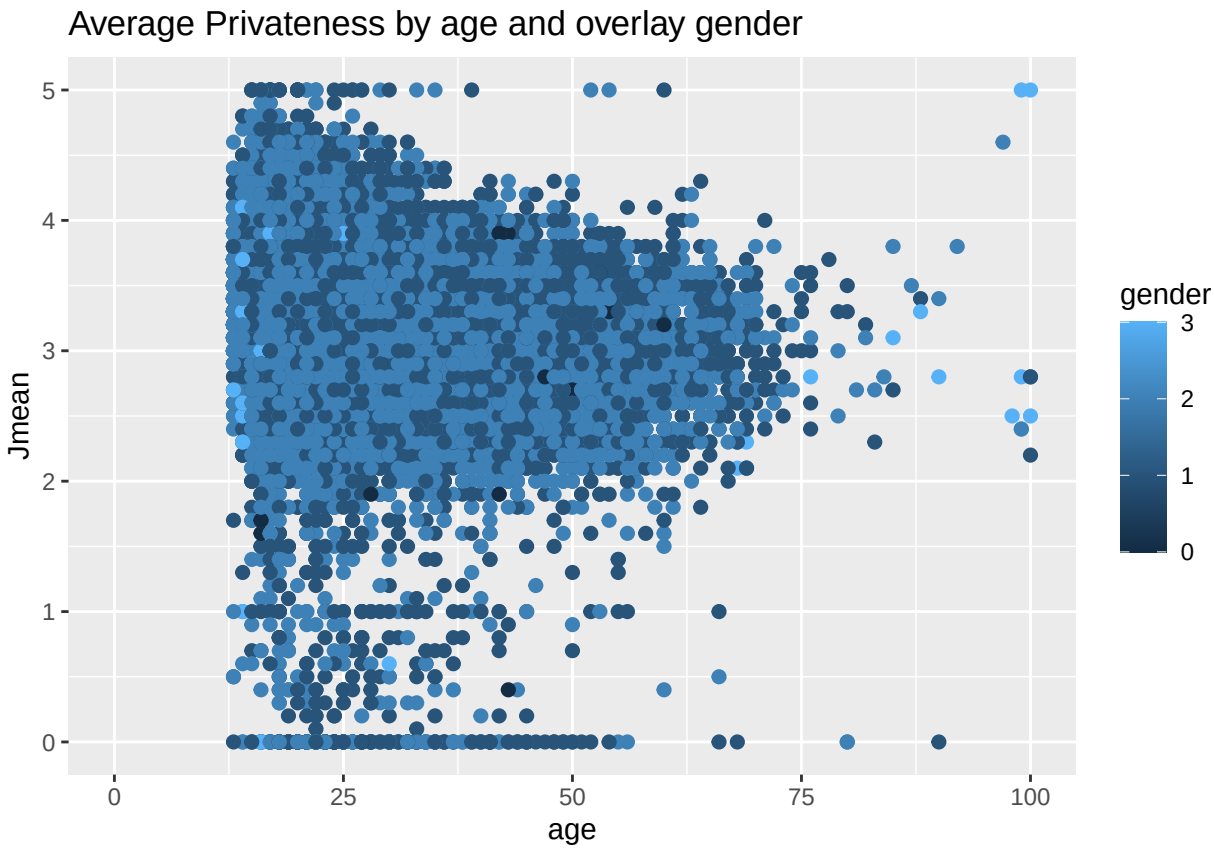

```
## Warning: Removed 68 rows containing missing values (geom_point).
```



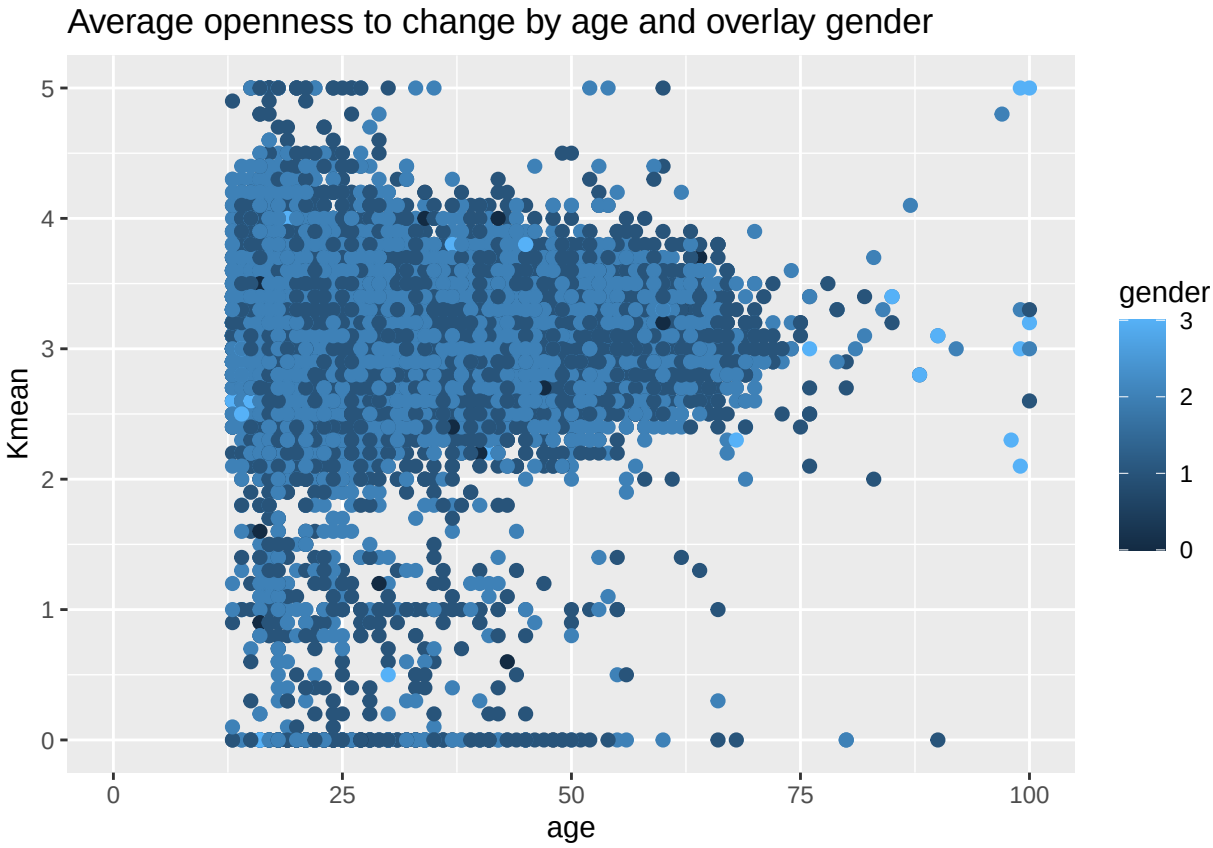

```
## Warning: Removed 68 rows containing missing values (geom_point).
```



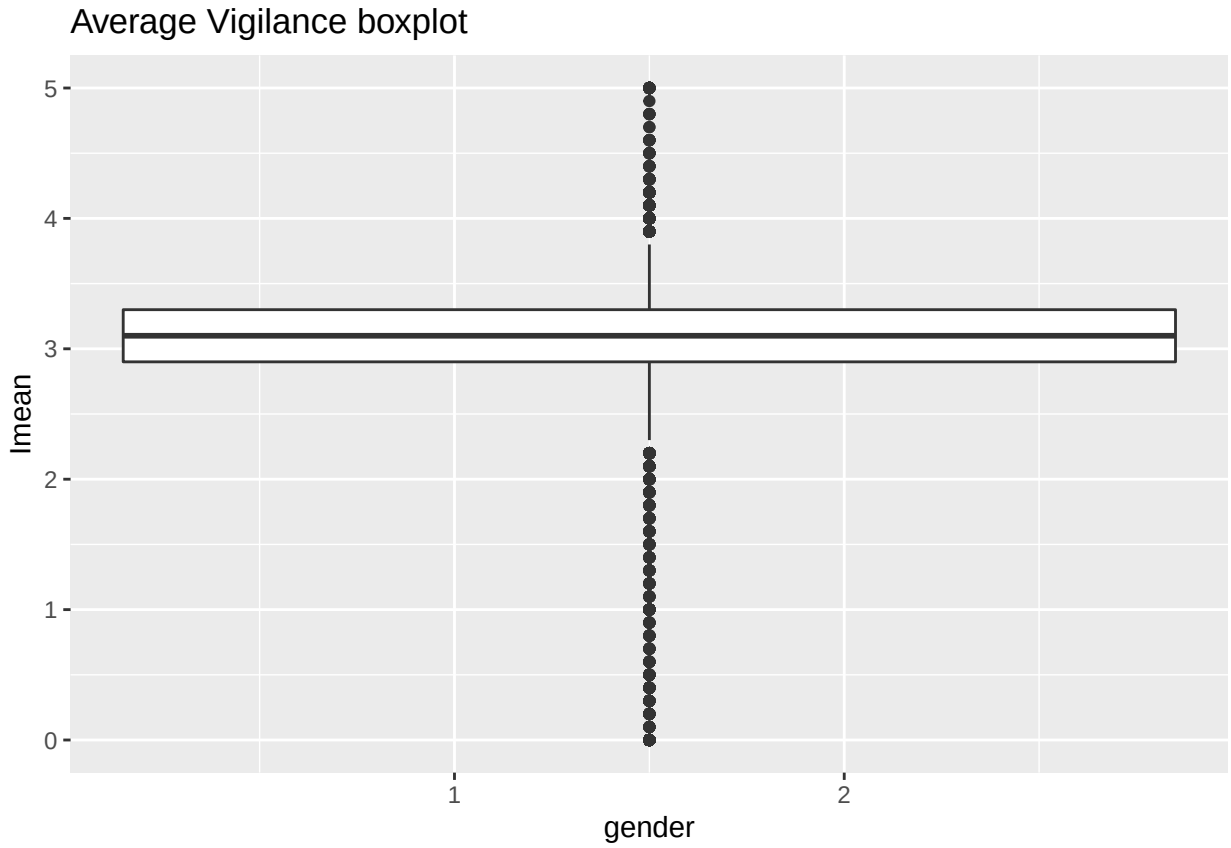

```
## Warning: Removed 68 rows containing missing values (geom_point).
```



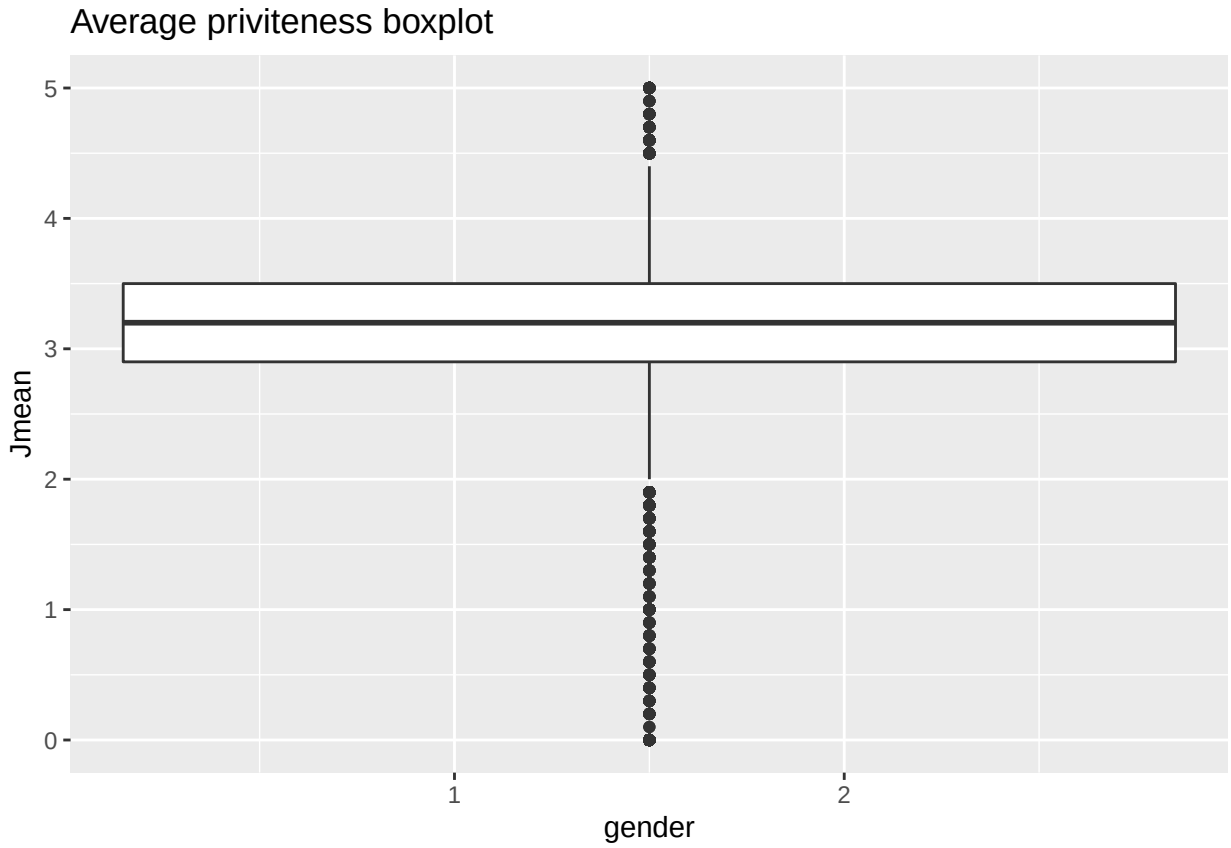

```
## Warning: Removed 68 rows containing missing values (geom_point).
```



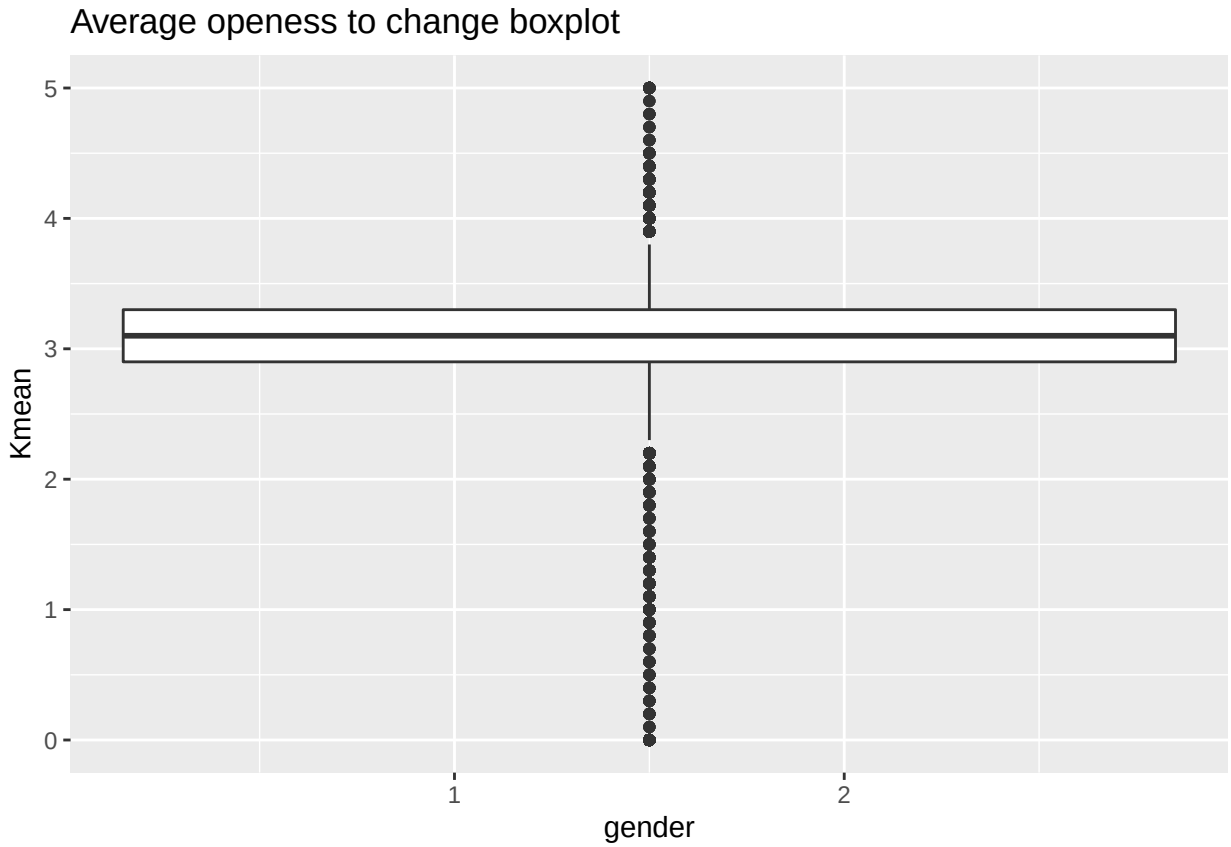

```
## Warning: Continuous x aesthetic -- did you forget aes(group=...)?
```




```
## Warning: Continuous x aesthetic -- did you forget aes(group=...)?
```




```
## Warning: Continuous x aesthetic -- did you forget aes(group=...)?
```


Average openess to change boxplot
Kmean
5
4
3
2
1
0
1
2
gender