Data Analysis in R

Howard Nguyen

2020-01-02

LOAD PACKAGES

```
# Load packages
library(tidyverse)  # Loads the `tidyverse` collection
library(readxl)     # Reads CSV and Excel files
```

LOAD DATA

```
# Also convert several adjacent variables to factors
df <- read_csv("../data/state_trends.csv") |>
  select(region:psy_reg) |>
  mutate(across(c(psych_region, psy_reg), as_factor)) |>
  print()
```

```
## Rows: 48 Columns: 34
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (11): state, state_code, region, psych_region, psy_reg, has_nba, has_nfl...
## dbl (23): population, sq_miles, pop_density, extraversion, agreeableness, co...
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
## # A tibble: 48 x 3
##    region    psych_region                  psy_reg
##    <chr>     <fct>                         <fct>
##  1 South     Friendly and Conventional     Friendly
##  2 West      Relaxed and Creative          Creative
##  3 South     Friendly and Conventional     Friendly
##  4 West      Relaxed and Creative          Creative
##  5 West      Friendly and Conventional     Friendly
##  6 Northeast Temperamental and Uninhibited Uninhibited
##  7 South     Temperamental and Uninhibited Uninhibited
##  8 South     Friendly and Conventional     Friendly
##  9 South     Friendly and Conventional     Friendly
## 10 West      Relaxed and Creative          Creative
## # ... with 38 more rows
```

SUMMARIZE DATAFRAME

```
summary(df)  # Gives frequencies for factors only
```

```
##     region                                 psych_region        psy_reg
##  Length:48          Friendly and Conventional    :24    Friendly   :24
##  Class :character   Relaxed and Creative         :10    Creative   :10
##  Mode  :character   Temperamental and Uninhibited:14    Uninhibited:14
```

SUMMARIZE CATEGORICAL VARIABLE

```
# "region" is a character variable
# summary() not very useful
df |>
  select(region) |>
  summary()
```

```
##     region
##  Length:48
##  Class :character
##  Mode  :character
```

```
# table() works better
df |>
  select(region) |>
  table()
```

```
## region
##   Midwest Northeast     South      West
##        12         9        16        11
```

SUMMARIZE FACTOR

```
# "psych_region" is a factor
# Using summary() works best
df |>
  select(psych_region) |>
  summary()
```

```
##                         psych_region
##  Friendly and Conventional    :24
##  Relaxed and Creative         :10
##  Temperamental and Uninhibited:14
```

```
# Using table()
df |>
  select(psych_region) |>
  table()
```

```
## psych_region
##     Friendly and Conventional          Relaxed and Creative
##                            24                            10
## Temperamental and Uninhibited
##                            14
```

```
# Convert region to a factor
df <- df |>
  mutate(region = as_factor(region)) |>
  print()
```

```
## # A tibble: 48 x 3
##    region    psych_region                  psy_reg
##    <fct>     <fct>                         <fct>
##  1 South     Friendly and Conventional     Friendly
##  2 West      Relaxed and Creative          Creative
##  3 South     Friendly and Conventional     Friendly
##  4 West      Relaxed and Creative          Creative
##  5 West      Friendly and Conventional     Friendly
##  6 Northeast Temperamental and Uninhibited Uninhibited
##  7 South     Temperamental and Uninhibited Uninhibited
##  8 South     Friendly and Conventional     Friendly
##  9 South     Friendly and Conventional     Friendly
## 10 West      Relaxed and Creative          Creative
## # ... with 38 more rows
```

```
# Summarize multiple factors
summary(df)
```

```
##        region                          psych_region        psy_reg
##  South    :16   Friendly and Conventional    :24    Friendly   :24
##  West     :11   Relaxed and Creative         :10    Creative   :10
##  Northeast: 9   Temperamental and Uninhibited:14    Uninhibited:14
##  Midwest  :12
```

DESCRIPTIVES

```
# Also convert several adjacent variables to factors
df_des <- read_csv("../data/state_trends.csv") |>
  mutate(across(c(
    region, psych_region, psy_reg, has_nba:has_any
    ),
    as_factor)
  ) |>
  print()
```

```
## Rows: 48 Columns: 34
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (11): state, state_code, region, psych_region, psy_reg, has_nba, has_nfl...
## dbl (23): population, sq_miles, pop_density, extraversion, agreeableness, co...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
## # A tibble: 48 x 34
##    state  state~1 popul~2 sq_mi~3 pop_d~4 region psych~5 psy_reg extra~6 agree~7
##    <chr>  <chr>     <dbl>   <dbl>   <dbl> <fct>  <fct>   <fct>     <dbl>   <dbl>
##  1 Alaba~ AL       5.02e6   52420      96 South  Friend~ Friend~    55.5    52.7
##  2 Arizo~ AZ       7.15e6  113990      63 West   Relaxe~ Creati~    50.6    46.6
##  3 Arkan~ AR       3.01e6   53179      57 South  Friend~ Friend~    49.9    52.7
##  4 Calif~ CA       3.95e7  163695     242 West   Relaxe~ Creati~    51.4    49
##  5 Color~ CO       5.77e6  104094      55 West   Friend~ Friend~    45.3    47.5
##  6 Conne~ CT       3.61e6    5543     650 North~ Temper~ Uninhi~    57.6    38.6
##  7 Delaw~ DE       9.90e5    2489     398 South  Temper~ Uninhi~    47      38.8
##  8 Flori~ FL       2.15e7   65758     328 South  Friend~ Friend~    60.9    50.7
##  9 Georg~ GA       1.07e7   59425     180 South  Friend~ Friend~    63.2    60
## 10 Idaho  ID       1.84e6   83569      22 West   Relaxe~ Creati~    40.7    52.9
## # ... with 38 more rows, 24 more variables: conscientiousness <dbl>,
## #   neuroticism <dbl>, openness <dbl>, data_science <dbl>,
## #   artificial_intelligence <dbl>, machine_learning <dbl>, data_analysis <dbl>,
## #   business_intelligence <dbl>, spreadsheet <dbl>, statistics <dbl>,
## #   art <dbl>, dance <dbl>, museum <dbl>, basketball <dbl>, football <dbl>,
## #   baseball <dbl>, soccer <dbl>, hockey <dbl>, has_nba <fct>, has_nfl <fct>,
## #   has_mlb <fct>, has_mls <fct>, has_nhl <fct>, has_any <fct>, and ...
```

SUMMARY

```r
# Summary for entire dataset
df_des |> summary()
```

```
##     state            state_code          population          sq_miles
##  Length:48          Length:48          Min.   :  576851   Min.   :  1545
##  Class :character   Class :character   1st Qu.: 2078518   1st Qu.: 39411
##  Mode  :character   Mode  :character   Median : 4841018   Median : 57094
##                                        Mean   : 6845231   Mean   : 65008
##                                        3rd Qu.: 7936809   3rd Qu.: 83901
##                                        Max.   :39538223   Max.   :268596
##   pop_density          region                              psych_region
##  Min.   :   6.0   South    :16   Friendly and Conventional    :24
##  1st Qu.:  52.0   West     :11   Relaxed and Creative         :10
##  Median :  93.0   Northeast: 9   Temperamental and Uninhibited:14
##  Mean   : 178.4   Midwest  :12
##  3rd Qu.: 206.8
##  Max.   :1065.0
##         psy_reg    extraversion   agreeableness   conscientiousness
##  Friendly   :24   Min.   :26.50   Min.   :29.80   Min.   :24.00
```

```
##  Creative   :10   1st Qu.:44.35   1st Qu.:45.77   1st Qu.:43.05
##  Uninhibited:14   Median :51.15   Median :52.05   Median :51.35
##                   Mean   :49.70   Mean   :50.59   Mean   :50.12
##                   3rd Qu.:56.05   3rd Qu.:56.62   3rd Qu.:56.12
##                   Max.   :69.80   Max.   :69.40   Max.   :69.60
##   neuroticism       openness      data_science   artificial_intelligence
##  Min.   :30.40   Min.   :21.80   Min.   :17.00   Min.   :18.00
##  1st Qu.:43.85   1st Qu.:42.70   1st Qu.:22.00   1st Qu.:23.00
##  Median :49.00   Median :49.85   Median :27.00   Median :26.00
##  Mean   :50.19   Mean   :49.43   Mean   :31.62   Mean   :27.94
##  3rd Qu.:56.92   3rd Qu.:56.67   3rd Qu.:37.00   3rd Qu.:30.00
##  Max.   :79.20   Max.   :65.00   Max.   :74.00   Max.   :56.00
##  machine_learning data_analysis  business_intelligence  spreadsheet
##  Min.   : 19.0    Min.   :27.0   Min.   : 24.00        Min.   :49.00
##  1st Qu.: 22.0    1st Qu.:31.0   1st Qu.: 44.50        1st Qu.:63.00
##  Median : 32.0    Median :35.0   Median : 52.50        Median :68.50
##  Mean   : 36.4    Mean   :37.4   Mean   : 52.21        Mean   :69.42
##  3rd Qu.: 42.0    3rd Qu.:40.0   3rd Qu.: 59.75        3rd Qu.:76.25
##  Max.   :100.0    Max.   :64.0   Max.   :100.00        Max.   :88.00
##    statistics         art             dance             museum
##  Min.   :41.00   Min.   : 65.00   Min.   : 59.00   Min.   :14.00
##  1st Qu.:53.00   1st Qu.: 72.00   1st Qu.: 66.75   1st Qu.:23.00
##  Median :55.00   Median : 75.00   Median : 70.00   Median :26.00
##  Mean   :56.23   Mean   : 76.75   Mean   : 70.83   Mean   :26.29
##  3rd Qu.:62.00   3rd Qu.: 80.00   3rd Qu.: 74.00   3rd Qu.:29.00
##  Max.   :73.00   Max.   :100.00   Max.   :100.00   Max.   :41.00
##    basketball        football         baseball          soccer
##  Min.   : 21.00   Min.   : 19.00   Min.   : 27.00   Min.   : 41.00
##  1st Qu.: 33.00   1st Qu.: 29.75   1st Qu.: 32.75   1st Qu.: 60.75
##  Median : 39.50   Median : 40.00   Median : 38.00   Median : 67.00
##  Mean   : 44.31   Mean   : 42.81   Mean   : 41.38   Mean   : 67.33
##  3rd Qu.: 51.50   3rd Qu.: 51.25   3rd Qu.: 43.75   3rd Qu.: 76.00
##  Max.   :100.00   Max.   :100.00   Max.   :100.00   Max.   :100.00
##      hockey       has_nba  has_nfl  has_mlb  has_mls  has_nhl  has_any
##  Min.   :  4.00   No :27   No :26   No :31   No :31   No :30   No :21
##  1st Qu.:  8.00   Yes:21   Yes:22   Yes:17   Yes:17   Yes:18   Yes:27
##  Median : 13.50
##  Mean   : 20.29
##  3rd Qu.: 22.75
##  Max.   :100.00
```

```
# Summary for one variable
df_des |>
  select(statistics) |>
  summary()
```

```
##    statistics
##  Min.   :41.00
##  1st Qu.:53.00
##  Median :55.00
##  Mean   :56.23
##  3rd Qu.:62.00
##  Max.   :73.00
```

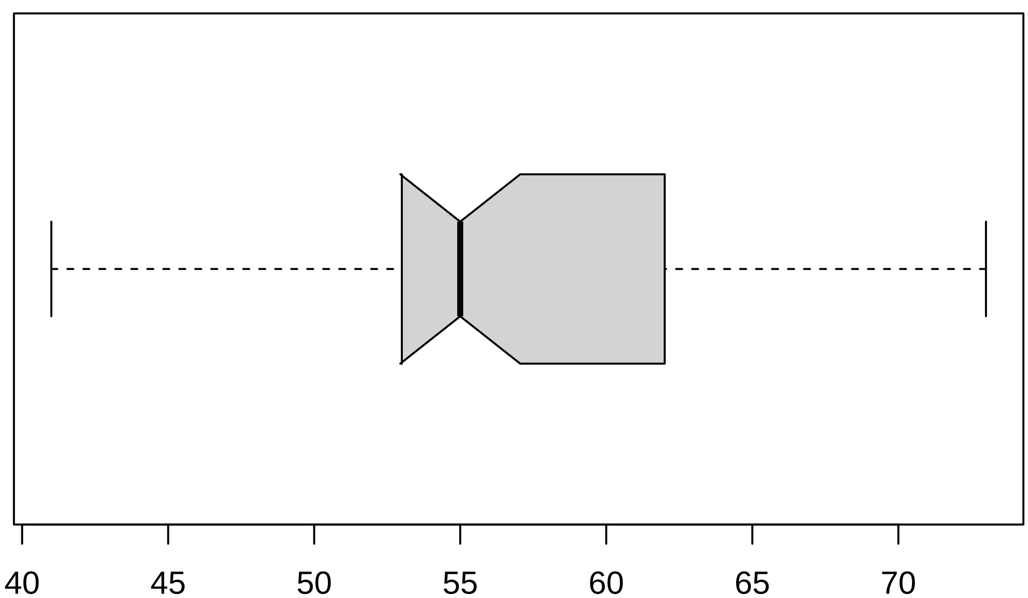
QUARTILES

```r
# Tukey's five-number summary: minimum, lower-hinge,
# median, upper-hinge, maximum. No labels.
fivenum(df_des$statistics)
```

```
## [1] 41 53 55 62 73
```

```r
# Boxplot stats: hinges, n, CI for median, and outliers
boxplot(df_des$statistics, notch = T, horizontal = T)
```

```
## Warning in (function (z, notch = FALSE, width = NULL, varwidth = FALSE, : some
## notches went outside hinges ('box'): maybe set notch=FALSE
```




```r
boxplot.stats(df_des$statistics)
```

```
## $stats
## [1] 41 53 55 62 73
##
## $n
## [1] 48
##
```

```
## $conf
## [1] 52.94752 57.05248
##
## $out
## numeric(0)
```

CORRELATIONS

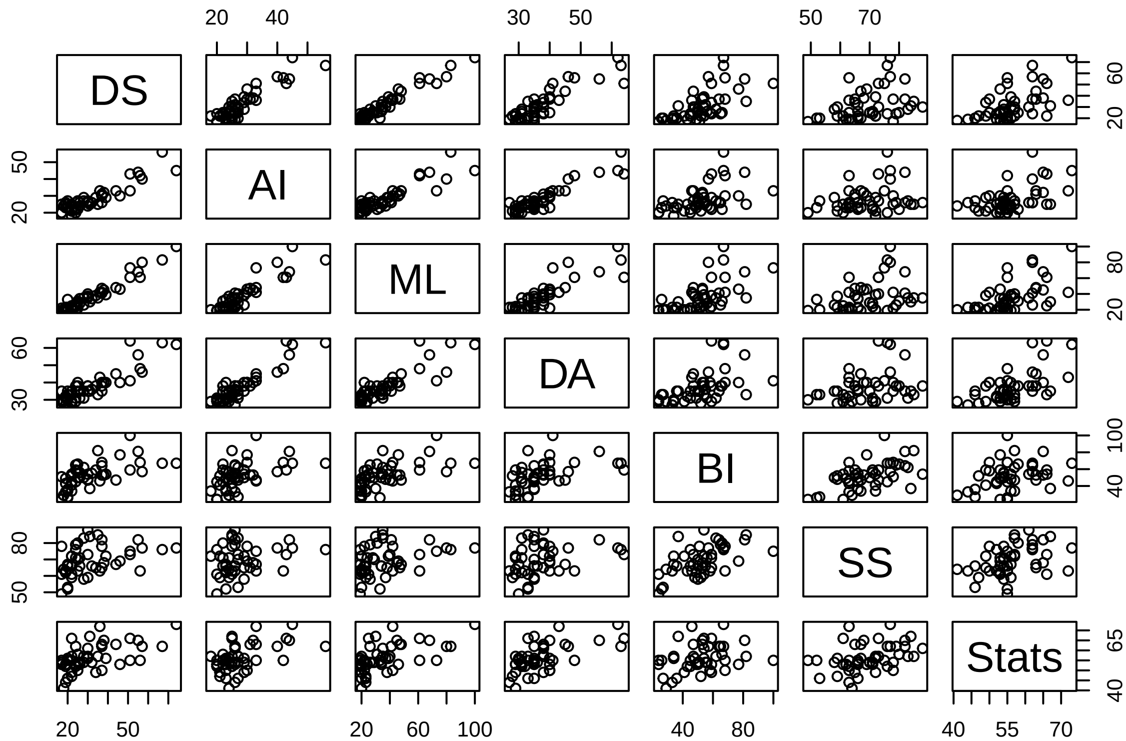
```
# Load data and convert several adjacent variables to factors
df_cor <- read_csv("../data/state_trends.csv") |>
  select(  # Rename variables with `select`
    DS = data_science,  # New = old
    AI = artificial_intelligence,
    ML = machine_learning,
    DA = data_analysis,
    BI = business_intelligence,
    SS = spreadsheet,
    Stats = statistics) |>
  print()
```

```
## Rows: 48 Columns: 34
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (11): state, state_code, region, psych_region, psy_reg, has_nba, has_nfl...
## dbl (23): population, sq_miles, pop_density, extraversion, agreeableness, co...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
## # A tibble: 48 x 7
##       DS    AI    ML    DA    BI    SS Stats
##    <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1    20    23    23    35    36    66    46
##  2    26    26    34    35    50    66    53
##  3    25    23    22    40    58    63    50
##  4    57    40    80    46    57    77    62
##  5    37    26    41    35    64    82    62
##  6    38    32    45    40    53    68    65
##  7    36    33    42    43    46    63    72
##  8    28    29    26    35    50    58    54
##  9    34    29    38    38    59    65    49
## 10    22    25    25    33    54    61    66
## # ... with 38 more rows
```

CORRELATION MATRIX

```
# Scatterplot matrix
df_cor |> plot()
```




```
# Correlation matrix
df_cor |> cor()
```

```
##              DS        AI        ML        DA        BI        SS     Stats
## DS    1.0000000 0.9074983 0.9687222 0.8836444 0.6460126 0.3781570 0.5710977
## AI    0.9074983 1.0000000 0.8973295 0.9182502 0.4998677 0.2492768 0.5088401
## ML    0.9687222 0.8973295 1.0000000 0.8655480 0.6084254 0.3481992 0.5738845
## DA    0.8836444 0.9182502 0.8655480 1.0000000 0.5024155 0.3025197 0.6008586
## BI    0.6460126 0.4998677 0.6084254 0.5024155 1.0000000 0.5657397 0.3100625
## SS    0.3781570 0.2492768 0.3481992 0.3025197 0.5657397 1.0000000 0.3825016
## Stats 0.5710977 0.5088401 0.5738845 0.6008586 0.3100625 0.3825016 1.0000000
```

```
# Rounded to 2 decimals
df_cor |> cor() |>
  round(2)
```

```
##         DS   AI   ML   DA   BI   SS Stats
## DS    1.00 0.91 0.97 0.88 0.65 0.38  0.57
## AI    0.91 1.00 0.90 0.92 0.50 0.25  0.51
## ML    0.97 0.90 1.00 0.87 0.61 0.35  0.57
## DA    0.88 0.92 0.87 1.00 0.50 0.30  0.60
## BI    0.65 0.50 0.61 0.50 1.00 0.57  0.31
## SS    0.38 0.25 0.35 0.30 0.57 1.00  0.38
## Stats 0.57 0.51 0.57 0.60 0.31 0.38  1.00
```

TEST AND CI FOR A SINGLE CORRELATION

```
# Can test one pair of variables at a time.
# Gives r, hypothesis test, and confidence interval
cor.test(df_cor$DS, df_cor$DA)
```

```
##
##  Pearson's product-moment correlation
##
## data:  df_cor$DS and df_cor$DA
## t = 12.802, df = 46, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.8004926 0.9334212
## sample estimates:
##       cor
## 0.8836444
```

PACKAGES TO GET P-VALUES FOR MATRIX

```
# The `Hmisc` package can get p-values for matrix
#browseURL("https://cran.r-project.org/web/packages/Hmisc/")

# The `rstatix` package is another option (with graphs)
#browseURL("https://cran.r-project.org/web/packages/rstatix/")
```

REGRESSION

LOAD DATA

```
# Select the personality and Google Trends variables
df_reg <- read_csv("../data/state_trends.csv") |>
  select(extraversion:hockey) |>
  print()
```

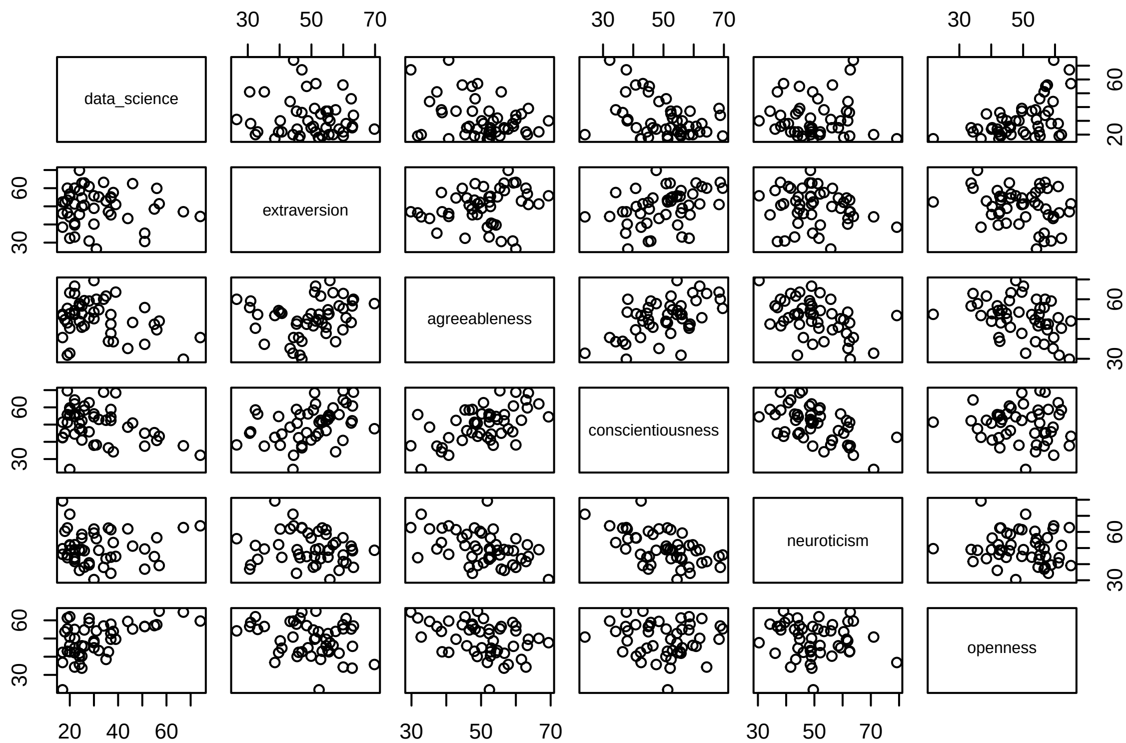
```
## Rows: 48 Columns: 34
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (11): state, state_code, region, psych_region, psy_reg, has_nba, has_nfl...
## dbl (23): population, sq_miles, pop_density, extraversion, agreeableness, co...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
## # A tibble: 48 x 20
##    extraversion agreea~1 consc~2 neuro~3 openn~4 data_~5 artif~6 machi~7 data_~8
```

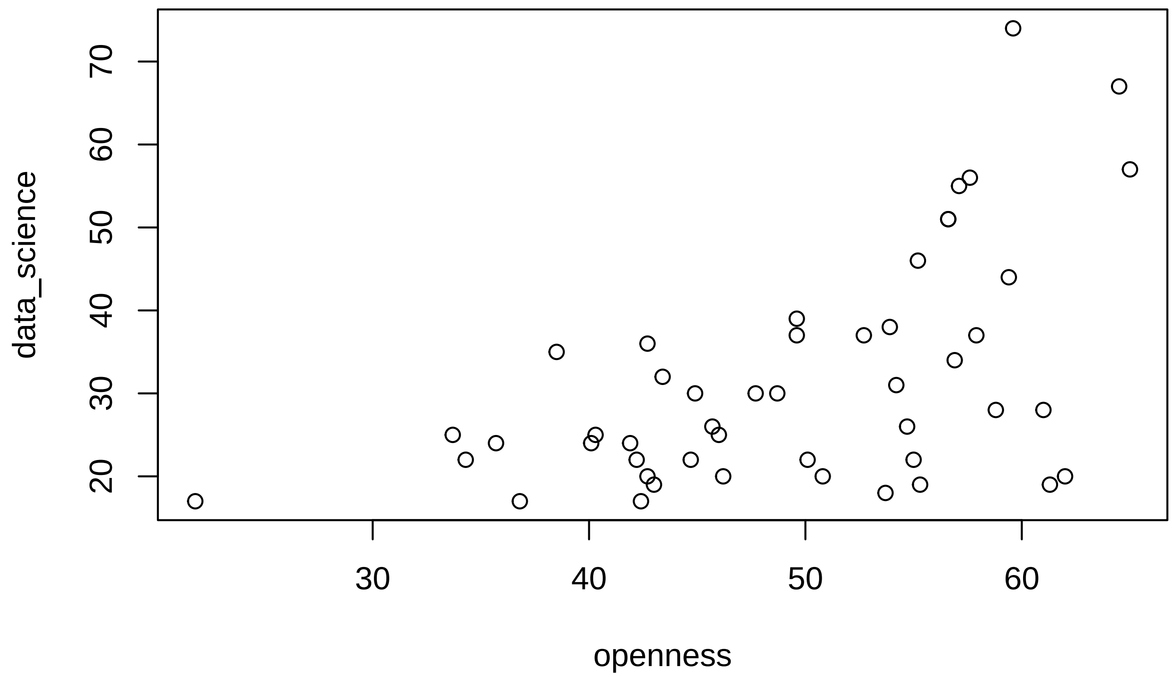
```
##           <dbl>    <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1          55.5     52.7    55.5    48.7    42.7      20      23      23      35
##  2          50.6     46.6    58.4    38.1    54.7      26      26      34      35
##  3          49.9     52.7    41      56.2    40.3      25      23      22      40
##  4          51.4     49      43.2    39.1    65        57      40      80      46
##  5          45.3     47.5    58.8    34.3    57.9      37      26      41      35
##  6          57.6     38.6    34.2    53.4    53.9      38      32      45      40
##  7          47       38.8    36.5    62.4    42.7      36      33      42      43
##  8          60.9     50.7    62.7    40.8    61        28      29      26      35
##  9          63.2     60      68.8    38      56.9      34      29      38      38
## 10          40.7     52.9    44.5    44.2    44.7      22      25      25      33
## # ... with 38 more rows, 11 more variables: business_intelligence <dbl>,
## #   spreadsheet <dbl>, statistics <dbl>, art <dbl>, dance <dbl>, museum <dbl>,
## #   basketball <dbl>, football <dbl>, baseball <dbl>, soccer <dbl>,
## #   hockey <dbl>, and abbreviated variable names 1: agreeableness,
## #   2: conscientiousness, 3: neuroticism, 4: openness, 5: data_science,
## #   6: artificial_intelligence, 7: machine_learning, 8: data_analysis
```

SCATTERPLOTS

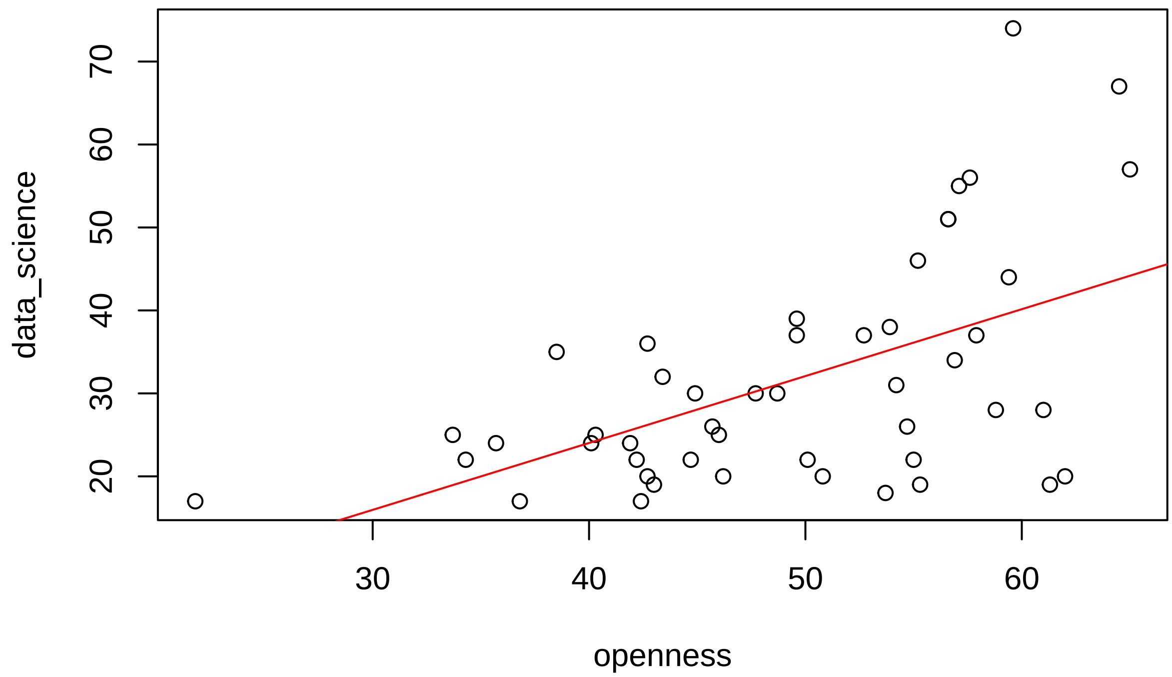
```r
# Scatterplot of "data_science" and personality variables
df_reg |>
  select(data_science, extraversion:openness) |>
  plot()
```




```r
# Quick graphical check on bivariate association
df_reg |>
  select(openness, data_science) |>
  plot()
```




```
# Add regression line with lm(); usage: y ~ X
# Note different variable order (vs plot)
df_reg |>
  select(openness, data_science) |>
  plot()
lm(df_reg$data_science ~ df_reg$openness) |> abline(col = "red")
```



BIVARIATE REGRESSION

```
# Compute and save bivariate regression
fit1 <- lm(df_reg$data_science ~ df_reg$openness)
```

```
# Show model
fit1
```

```
##
## Call:
## lm(formula = df_reg$data_science ~ df_reg$openness)
##
## Coefficients:
##     (Intercept)  df_reg$openness
##         -8.2243           0.8062
```

```
# Summarize regression model
summary(fit1)
```

```
##
## Call:
## lm(formula = df_reg$data_science ~ df_reg$openness)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -22.197  -7.822  -0.636   6.350  34.173
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -8.2243     9.2229  -0.892    0.377
## df_reg$openness   0.8062     0.1835   4.394 6.49e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.66 on 46 degrees of freedom
## Multiple R-squared:  0.2957, Adjusted R-squared:  0.2804
## F-statistic: 19.31 on 1 and 46 DF,  p-value: 6.488e-05
```

```
# Confidence intervals for coefficients
confint(fit1)
```

```
##                       2.5 %    97.5 %
## (Intercept)     -26.7890606 10.340520
## df_reg$openness   0.4369258  1.175521
```

```
# Predict values of "data science"
predict(fit1)
```

```
##         1         2         3         4         5         6         7         8
## 26.201468 35.876149 24.266532 44.180250 38.456064 35.231170 26.201468 40.955356
##         9        10        11        12        13        14        15        16
## 37.649840 27.813915 36.279260 27.975160 18.945458 24.105287 26.443335 35.069925
##        17        18        19        20        21        22        23        24
## 32.731878 37.407973 39.826643 26.765824 22.815330 29.023250 28.620138 36.118016
##        25        26        27        28        29        30        31        32
## 19.429192 41.197223 31.038808 38.214197 41.761579 43.777138 31.764409  9.351399
##        33        34        35        36        37        38        39        40
## 28.862005 25.798356 39.181665 31.764409 39.665399 36.359883 25.556489 32.167521
##        41        42        43        44        45        46        47        48
## 34.263702 30.232585 35.473037 37.811085 37.407973 21.444750 20.557904 25.959601
```

```
# Prediction intervals for values of "data_science"
predict(fit1, interval = "prediction")
```

```
## Warning in predict.lm(fit1, interval = "prediction"): predictions on current data refer to _future_ r
```

```
##          fit        lwr      upr
## 1  26.201468   2.3662348 50.03670
## 2  35.876149  12.0908906 59.66141
## 3  24.266532   0.3226886 48.21038
## 4  44.180250  19.7871953 68.57330
## 5  38.456064  14.5450318 62.36710
## 6  35.231170  11.4682748 58.99407
## 7  26.201468   2.3662348 50.03670
```

```
## 8  40.955356  16.8677571 65.04295
## 9  37.649840  13.7843267 61.51535
## 10 27.813915   4.0443105 51.58352
## 11 36.279260  12.4781738 60.08035
## 12 27.975160   4.2108654 51.73945
## 13 18.945458  -5.4610823 43.35200
## 14 24.105287   0.1509350 48.05964
## 15 26.443335   2.6193943 50.26728
## 16 35.069925  11.3120501 58.82780
## 17 32.731878   9.0210428 56.44271
## 18 37.407973  13.5550171 61.26093
## 19 39.826643  15.8253845 63.82790
## 20 26.765824   2.9561468 50.57550
## 21 22.815330  -1.2310968 46.86176
## 22 29.023250   5.2878987 52.75860
## 23 28.620138   4.8747996 52.36548
## 24 36.118016  12.3234312 59.91260
## 25 19.429192  -4.9255705 43.78395
## 26 41.197223  17.0897203 65.30473
## 27 31.038808   7.3318744 54.74574
## 28 38.214197  14.3174104 62.11098
## 29 41.761579  17.6057220 65.91744
## 30 43.777138  19.4269563 68.12732
## 31 31.764409   8.0589101 55.46991
## 32  9.351399 -16.4563515 35.15915
## 33 28.862005   5.1228309 52.60118
## 34 25.798356   1.9431707 49.65354
## 35 39.181665  15.2248726 63.13846
## 36 31.764409   8.0589101 55.46991
## 37 39.665399  15.6755899 63.65521
## 38 36.359883  12.5554599 60.16431
## 39 25.556489   1.6886545 49.42432
## 40 32.167521   8.4608053 55.87424
## 41 34.263702  10.5274948 57.99991
## 42 30.232585   6.5185930 53.94658
## 43 35.473037  11.7021836 59.24389
## 44 37.811085  13.9369178 61.68525
## 45 37.407973  13.5550171 61.26093
## 46 21.444750  -2.7149603 45.60446
## 47 20.557904  -3.6834905 44.79930
## 48 25.959601   2.1125659 49.80664
```

```
# Regression diagnostics
lm.influence(fit1)
```

```
## $hat
##          1          2          3          4          5          6          7
## 0.03204490 0.02772167 0.04147177 0.08091649 0.03861935 0.02579005 0.03204490
##          8          9         10         11         12         13         14
## 0.05401502 0.03466878 0.02636937 0.02908997 0.02591083 0.08211198 0.04238617
##         15         16         17         18         19         20         21
## 0.03106722 0.02535670 0.02130032 0.03358023 0.04647249 0.02983302 0.05041491
##         22         23         24         25         26         27         28
## 0.02341341 0.02427486 0.02852779 0.07752552 0.05575763 0.02096431 0.03738215
```

```
##         29         30         31         32         33         34         35
## 0.05999714 0.07712024 0.02084074 0.20992957 0.02374313 0.03377347 0.04259852
##         36         37         38         39         40         41         42
## 0.02084074 0.04547427 0.02937850 0.03487007 0.02094552 0.02348722 0.02157232
##         43         44         45         46         47         48
## 0.02647725 0.03541925 0.03358023 0.06033532 0.06751747 0.03306718
##
## $coefficients
##     (Intercept) df_reg$openness
## 1  -0.661243471    0.0106777299
## 2   0.444262476   -0.0132696890
## 3   0.101465339   -0.0017302988
## 4  -2.369351364    0.0538154992
## 5   0.125590014   -0.0031792941
## 6  -0.096461644    0.0031495416
## 7   1.044787321   -0.0168711787
## 8   1.655507634   -0.0392663702
## 9   0.267222337   -0.0070000389
## 10 -0.470060791    0.0069932736
## 11 -0.499188974    0.0143195190
## 12  0.158542334   -0.0023314355
## 13  1.407753236   -0.0257011543
## 14 -0.014848249    0.0002540647
## 15 -0.764638068    0.0122320902
## 16  0.551530983   -0.0185405734
## 17 -0.052312107   -0.0044248647
## 18 -0.942349197    0.0249934967
## 19 -3.717906815    0.0903259877
## 20  0.510585247   -0.0080560436
## 21  1.984291426   -0.0347373723
## 22 -0.557615112    0.0073871213
## 23 -0.178502804    0.0024795829
## 24  0.688990775   -0.0200649761
## 25  0.574295673   -0.0104443967
## 26  2.928080007   -0.0691489198
## 27 -0.031552372    0.0001911322
## 28 -1.464232864    0.0374118744
## 29  3.082003742   -0.0721124239
## 30 -4.120333190    0.0939681638
## 31  0.138302747    0.0003165704
## 32  3.476819223   -0.0662619232
## 33 -0.248431728    0.0033588162
## 34 -0.429800977    0.0070387030
## 35  1.097177894   -0.0271206404
## 36  0.100074286    0.0002290667
## 37 -0.459970218    0.0112200949
## 38  0.913648305   -0.0260233701
## 39 -0.182248441    0.0030074605
## 40 -0.130779946   -0.0017313399
## 41 -0.053927444    0.0022721303
## 42 -0.009979763    0.0001017136
## 43  0.172822854   -0.0054331671
## 44 -1.303105522    0.0338753022
## 45 -0.942349197    0.0249934967
```

```
## 46 -0.829945259    0.0147975669
## 47  0.697397610   -0.0125537453
## 48 -0.990384870    0.0161317059
##
## $sigma
##        1        2        3        4        5        6        7        8
## 11.74726 11.68981 11.78425 11.61496 11.78270 11.77736 11.69089 11.61629
##        9       10       11       12       13       14       15       16
## 11.77176 11.75200 11.69266 11.78081 11.74707 11.78477 11.73075 11.49945
##       17       18       19       20       21       22       23       24
## 11.62757 11.60315 10.56715 11.75813 11.63644 11.70591 11.77815 11.58972
##       25       26       27       28       29       30       31       32
## 11.77802 11.28208 11.78374 11.47076 11.29981 11.22030 11.73426 11.71476
##       33       34       35       36       37       38       39       40
## 11.77037 11.77069 11.66100 11.75836 11.76621 11.48831 11.78241 11.68480
##       41       42       43       44       45       46       47       48
## 11.77755 11.78473 11.76539 11.49235 11.60315 11.76494 11.77280 11.70625
##
## $wt.res
##           1           2           3           4           5           6
##  -6.2014681  -9.8761487   0.7334680  12.8197504  -1.4560636   2.7688300
##           7           8           9          10          11          12
##   9.7985319 -12.9553561  -3.6498402  -5.8139149   9.7207396   2.0248405
##          13          14          15          16          17          18
##   6.0545424  -0.1052873  -7.4433351 -17.0699253 -12.7318775  13.5920268
##          19          20          21          22          23          24
##  34.1733567   5.2341755  12.1846701  -9.0232500  -2.6201383 -14.1180157
##          25          26          27          28          29          30
##   2.5708083 -22.1972231  -1.0388084  17.7858035 -21.7615794  23.2228621
##          31          32          33          34          35          36
##   7.2355905   7.6486007  -3.8620053  -3.7983564 -11.1816646   5.2355905
##          37          38          39          40          41          42
##   4.3346014 -17.3598828  -1.5564894 -10.1675212   2.7362980  -0.2325850
##          43          44          45          46          47          48
##  -4.4730370  17.1889151  13.5920268  -4.4447501   3.4420956  -8.9596011
```

```
influence.measures(fit1)
```

```
## Influence measures of
##   lm(formula = df_reg$data_science ~ df_reg$openness) :
##
##      dfb.1_  dfb.df_.    dffit cov.r   cook.d    hat inf
## 1  -0.07114  0.057748 -0.09763 1.066 4.84e-03 0.0320
## 2   0.04803 -0.072119 -0.14468 1.041 1.05e-02 0.0277
## 3   0.01088 -0.009329  0.01322 1.090 8.94e-05 0.0415
## 4  -0.25781  0.294363  0.34161 1.073 5.79e-02 0.0809
## 5   0.01347 -0.017143 -0.02526 1.086 3.26e-04 0.0386
## 6  -0.01035  0.016990  0.03875 1.070 7.67e-04 0.0258
## 7   0.11294 -0.091684  0.15500 1.046 1.21e-02 0.0320
## 8   0.18011 -0.214757 -0.27400 1.043 3.73e-02 0.0540
## 9   0.02869 -0.037779 -0.05980 1.078 1.82e-03 0.0347
## 10 -0.05055  0.037806 -0.08251 1.061 3.46e-03 0.0264
## 11 -0.05396  0.077805  0.14604 1.043 1.07e-02 0.0291
## 12  0.01701 -0.012573  0.02840 1.071 4.12e-04 0.0259
```

```
## 13  0.15145 -0.139001  0.16090 1.124 1.31e-02 0.0821
## 14 -0.00159  0.001370 -0.00192 1.091 1.89e-06 0.0424
## 15 -0.08238  0.066247 -0.11542 1.059 6.75e-03 0.0311
## 16  0.06061 -0.102433 -0.24252 0.972 2.86e-02 0.0254
## 17 -0.00569 -0.024177 -0.16329 1.012 1.33e-02 0.0213
## 18 -0.10264  0.136850  0.22212 1.016 2.44e-02 0.0336
## 19 -0.44465  0.543061  0.73113 0.708 2.20e-01 0.0465   *
## 20  0.05488 -0.043529  0.07925 1.067 3.20e-03 0.0298
## 21  0.21551 -0.189658  0.24759 1.046 3.05e-02 0.0504
## 22 -0.06020  0.040093 -0.12078 1.042 7.36e-03 0.0234
## 23 -0.01915  0.013375 -0.03552 1.069 6.44e-04 0.0243
## 24  0.07513 -0.109992 -0.21179 1.006 2.22e-02 0.0285
## 25  0.06162 -0.056338  0.06588 1.130 2.22e-03 0.0775
## 26  0.32800 -0.389395 -0.49201 0.930 1.13e-01 0.0558
## 27 -0.00338  0.001030 -0.01304 1.067 8.69e-05 0.0210
## 28 -0.16132  0.207210  0.31143 0.974 4.70e-02 0.0374
## 29  0.34470 -0.405446 -0.50183 0.940 1.18e-01 0.0600
## 30 -0.46410  0.532071  0.62280 0.930 1.80e-01 0.0771
## 31  0.01490  0.001714  0.09091 1.049 4.19e-03 0.0208
## 32  0.37508 -0.359355  0.37863 1.291 7.24e-02 0.2099   *
## 33 -0.02667  0.018130 -0.05179 1.065 1.37e-03 0.0237
## 34 -0.04615  0.037991 -0.06138 1.076 1.92e-03 0.0338
## 35  0.11891 -0.147760 -0.20672 1.046 2.14e-02 0.0426
## 36  0.01076  0.001238  0.06565 1.058 2.19e-03 0.0208
## 37 -0.04941  0.060583  0.08230 1.088 3.45e-03 0.0455
## 38  0.10051 -0.143913 -0.26684 0.972 3.46e-02 0.0294
## 39 -0.01955  0.016217 -0.02556 1.082 3.34e-04 0.0349
## 40 -0.01414 -0.009414 -0.12863 1.032 8.31e-03 0.0209
## 41 -0.00579  0.012257  0.03646 1.067 6.79e-04 0.0235
## 42 -0.00107  0.000548 -0.00296 1.068 4.49e-06 0.0216
## 43  0.01856 -0.029339 -0.06355 1.066 2.06e-03 0.0265
## 44 -0.14330  0.187270  0.29182 0.980 4.14e-02 0.0354
## 45 -0.10264  0.136850  0.22212 1.016 2.44e-02 0.0336
## 46 -0.08915  0.079909 -0.09876 1.105 4.97e-03 0.0603
## 47  0.07487 -0.067747  0.08147 1.116 3.39e-03 0.0675
## 48 -0.10692  0.087550 -0.14394 1.052 1.04e-02 0.0331
```

MULTIPLE REGRESSION

```
# Moving the outcome, y, to the front and having nothing
# else but predictor variables, X, can make things easier
df_reg <- df_reg |>
  select(data_science, extraversion:openness) |>
  print()
```

```
## # A tibble: 48 x 6
##    data_science extraversion agreeableness conscientiousness neuroticism openn~1
##           <dbl>        <dbl>         <dbl>             <dbl>       <dbl>   <dbl>
##  1           20         55.5          52.7              55.5        48.7    42.7
##  2           26         50.6          46.6              58.4        38.1    54.7
##  3           25         49.9          52.7              41          56.2    40.3
```

```
##  4           57          51.4          49                 43.2        39.1    65
##  5           37          45.3          47.5               58.8        34.3    57.9
##  6           38          57.6          38.6               34.2        53.4    53.9
##  7           36          47            38.8               36.5        62.4    42.7
##  8           28          60.9          50.7               62.7        40.8    61
##  9           34          63.2          60                 68.8        38      56.9
## 10           22          40.7          52.9               44.5        44.2    44.7
## # ... with 38 more rows, and abbreviated variable name 1: openness
```

```
# Note that if you want to just move one variable to the
# front and keep everything else in the same order, you can
# do this: select(data_analysis, everything()) |>

# Three ways to specify model

# Most concise
lm(df_reg)
```

```
##
## Call:
## lm(formula = df_reg)
##
## Coefficients:
##       (Intercept)       extraversion      agreeableness  conscientiousness
##            7.1013             0.3570             0.1183            -0.7037
##       neuroticism           openness
##           -0.1442             0.8761
```

```
# Identify outcome, infer rest
lm(data_science ~ ., data = df_reg)
```

```
##
## Call:
## lm(formula = data_science ~ ., data = df_reg)
##
## Coefficients:
##       (Intercept)       extraversion      agreeableness  conscientiousness
##            7.1013             0.3570             0.1183            -0.7037
##       neuroticism           openness
##           -0.1442             0.8761
```

```
# Identify entire model
lm(data_science ~ extraversion + agreeableness +
   conscientiousness + neuroticism + openness, data = df_reg)
```

```
##
## Call:
## lm(formula = data_science ~ extraversion + agreeableness + conscientiousness +
##     neuroticism + openness, data = df_reg)
##
## Coefficients:
##       (Intercept)       extraversion      agreeableness  conscientiousness
```

```
##           7.1013            0.3570            0.1183           -0.7037
##       neuroticism          openness
##          -0.1442            0.8761
```

```r
# Save model
fit2 <- lm(df_reg)
```

```r
# Show model
fit2
```

```
##
## Call:
## lm(formula = df_reg)
##
## Coefficients:
##       (Intercept)       extraversion      agreeableness  conscientiousness
##           7.1013            0.3570            0.1183           -0.7037
##       neuroticism          openness
##          -0.1442            0.8761
```

```r
# Summarize regression model
summary(fit2)
```

```
##
## Call:
## lm(formula = df_reg)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -24.1425  -5.8822  -0.8419   7.3259  25.8733
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)         7.1013    27.7356   0.256  0.79917
## extraversion        0.3570     0.1789   1.996  0.05250 .
## agreeableness       0.1183     0.2384   0.496  0.62226
## conscientiousness  -0.7037     0.2161  -3.256  0.00224 **
## neuroticism        -0.1442     0.2009  -0.718  0.47683
## openness            0.8761     0.2033   4.309 9.68e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.67 on 42 degrees of freedom
## Multiple R-squared:  0.4612, Adjusted R-squared:  0.3971
## F-statistic:  7.19 on 5 and 42 DF,  p-value: 6.125e-05
```

CONTINGENCY

LOAD DATA

```r
# Also convert all variables to factors
df_cont <- read_csv("../data/state_trends.csv") |>
  select(region, psy_reg) |>
  mutate(across(everything(), as_factor)) |>
  print()
```

```
## Rows: 48 Columns: 34
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (11): state, state_code, region, psych_region, psy_reg, has_nba, has_nfl...
## dbl (23): population, sq_miles, pop_density, extraversion, agreeableness, co...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
## # A tibble: 48 x 2
##    region    psy_reg
##    <fct>     <fct>
##  1 South     Friendly
##  2 West      Creative
##  3 South     Friendly
##  4 West      Creative
##  5 West      Friendly
##  6 Northeast Uninhibited
##  7 South     Uninhibited
##  8 South     Friendly
##  9 South     Friendly
## 10 West      Creative
## # ... with 38 more rows
```

ANALYZE DATA

```r
# Create contingency table
ct <- table(df_cont$region, df_cont$psy_reg)
ct
```

```
##
##             Friendly Creative Uninhibited
##   South           10        2           4
##   West             3        8           0
##   Northeast        0        0           9
##   Midwest         11        0           1
```

```r
# Call also get cell, row, and column %
# With rounding to get just 2 decimal places
# Multiplied by 100 to make %

# Row percentages
ct |>
```

```r
  prop.table(1) |>  # 1 is for row percentages
  round(2) * 100
```

```
##
##              Friendly Creative Uninhibited
##   South            62       12          25
##   West             27       73           0
##   Northeast         0        0         100
##   Midwest          92        0           8
```

```r
# Column percentages
ct |>
  prop.table(2) |>  # 2 is for columns percentages
  round(2) * 100
```

```
##
##              Friendly Creative Uninhibited
##   South            42       20          29
##   West             12       80           0
##   Northeast         0        0          64
##   Midwest          46        0           7
```

```r
# Total percentages
ct |>
  prop.table() |>  # No argument for total percentages
  round(2) * 100
```

```
##
##              Friendly Creative Uninhibited
##   South            21        4           8
##   West              6       17           0
##   Northeast         0        0          19
##   Midwest          23        0           2
```

```r
# Chi-squared test (but n is small)
tchi <- chisq.test(ct)
```

```
## Warning in chisq.test(ct): Chi-squared approximation may be incorrect
```

```r
tchi
```

```
##
##  Pearson's Chi-squared test
##
## data:  ct
## X-squared = 50.002, df = 6, p-value = 4.697e-09
```

```r
# Get p-value  in one step
table(df_cont$region, df_cont$psy_reg) |> chisq.test()
```

```
## Warning in chisq.test(table(df_cont$region, df_cont$psy_reg)): Chi-squared
## approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  table(df_cont$region, df_cont$psy_reg)
## X-squared = 50.002, df = 6, p-value = 4.697e-09
```

```
# Additional tables
tchi$observed   # Observed frequencies (same as ct)
```

```
##
##             Friendly Creative Uninhibited
##   South           10        2           4
##   West             3        8           0
##   Northeast        0        0           9
##   Midwest         11        0           1
```

```
tchi$expected   # Expected frequencies
```

```
##
##             Friendly Creative Uninhibited
##   South          8.0 3.333333    4.666667
##   West           5.5 2.291667    3.208333
##   Northeast      4.5 1.875000    2.625000
##   Midwest        6.0 2.500000    3.500000
```

```
tchi$residuals  # Pearson's residual
```

```
##
##               Friendly   Creative Uninhibited
##   South      0.7071068 -0.7302967  -0.3086067
##   West      -1.0660036  3.7708009  -1.7911821
##   Northeast -2.1213203 -1.3693064   3.9347354
##   Midwest    2.0412415 -1.5811388  -1.3363062
```

```
tchi$stdres     # Standardized residual
```

```
##
##               Friendly   Creative Uninhibited
##   South      1.2247449 -1.0052494  -0.4490887
##   West      -1.7170914  4.8270542  -2.4240449
##   Northeast -3.3282012 -1.7073312   5.1866269
##   Midwest    3.3333333 -2.0519567  -1.8333970
```