

Final Version of Assignment One

Amahle Nzimande

2026-02-09

Table of contents

Assignment One	3
Welcome	3
1 Project One	4
2 Day Three Practical Q3 & Q4	8
3 Day Four Practical	11

Assignment One

Welcome

Dear fellow survivor of this course

If you are reading this, we have both made it this far, congratulations to us! I hope this assignment makes sense, runs without errors, and causes minimal emotional damage.

May your marking be fair, your coffee strong, and your rubric generous.

1 Project One

QUESTION ONE

```
down <- nrow(airquality)
across <- ncol(airquality)
sum <- 0

print('The following rows have missing information: ')
```

```
[1] "The following rows have missing information: "
```

```
for(i in 1:down){
  for(j in 1:across){
    if(is.na(airquality[i,j])){
      sum <- sum + 1
      print(i)
      break}
    }
  }
```

```
[1] 5
[1] 6
[1] 10
[1] 11
[1] 25
[1] 26
[1] 27
[1] 32
[1] 33
[1] 34
[1] 35
[1] 36
[1] 37
[1] 39
```

```
[1] 42
[1] 43
[1] 45
[1] 46
[1] 52
[1] 53
[1] 54
[1] 55
[1] 56
[1] 57
[1] 58
[1] 59
[1] 60
[1] 61
[1] 65
[1] 72
[1] 75
[1] 83
[1] 84
[1] 96
[1] 97
[1] 98
[1] 102
[1] 103
[1] 107
[1] 115
[1] 119
[1] 150
```

```
print(paste0('In total there are ', sum, ' rows with missing information.' ))
```

```
[1] "In total there are 42 rows with missing information."
```

QUESTION TWO

```
my_table <- data.frame(
  Column = c('Temperature', 'Ozone'),
  Mean = c(mean(airquality[,4]), mean(airquality[,1], na.rm=TRUE)),
  SD = c(sd(airquality[,4]), sd(airquality[,1], na.rm=TRUE)),
  Min = c(min(airquality[,4]), min(airquality[,1], na.rm=TRUE)),
  Max = c(max(airquality[,4]), max(airquality[,1], na.rm=TRUE))
```

```
)
```

```
print(my_table)
```

	Column	Mean	SD	Min	Max
1	Temperature	77.88235	9.46527	56	97
2	Ozone	42.12931	32.98788	1	168

QUESTION THREE

```
data(cars)
```

```
Y <- cars$dist
```

```
X <- cbind(1, cars$speed)
```

```
my_funct <- function(design, response) {
```

```
  a <- t(design)%*%design
```

```
  b <- solve(a)
```

```
  c <- t(design)%*%response
```

```
  d <- b%*%c
```

```
  return(d)
```

```
}
```

```
print(my_funct(X, Y))
```

```
      [,1]  
[1,] -17.579095  
[2,]   3.932409
```

QUESTION FOUR

```
model <- lm(cars$dist~cars$speed, data=cars )  
summary(model)
```

Call:

```
lm(formula = cars$dist ~ cars$speed, data = cars)
```

Residuals:

Min	1Q	Median	3Q	Max
-29.069	-9.525	-2.272	9.215	43.201

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-17.5791	6.7584	-2.601	0.0123 *
cars\$speed	3.9324	0.4155	9.464	1.49e-12 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 15.38 on 48 degrees of freedom

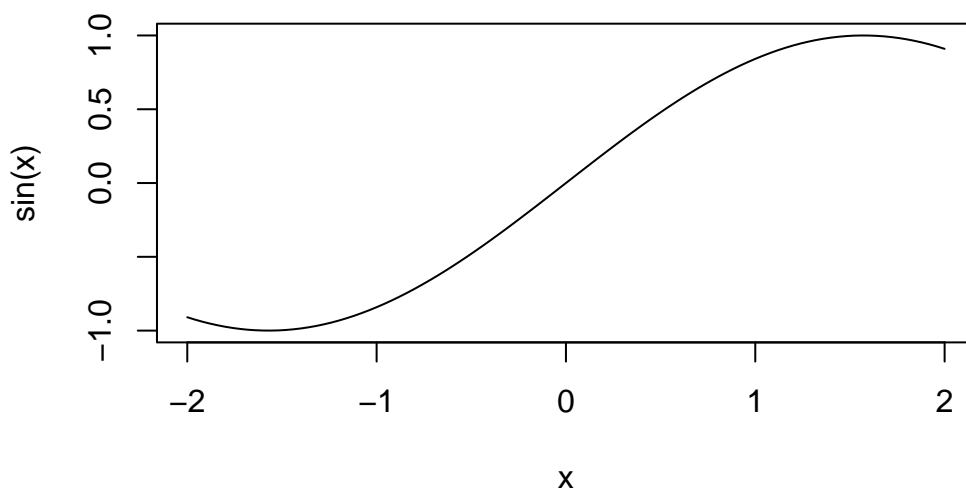
Multiple R-squared: 0.6511, Adjusted R-squared: 0.6438

F-statistic: 89.57 on 1 and 48 DF, p-value: 1.49e-12

2 Day Three Practical Q3 & Q4

QUESTION FOUR

```
curve(sin(x), from = -2, to = 2)
```



QUESTION FIVE

```
randoms <- rt(1000, 1)

#Creating a manual QQ-plot with 95% confidence interval

sorted_randoms <- sort(randoms)

n <- length(randoms)
i <- 1:n
```

```

# Calculate plotting positions (probabilities)
p <- (i - 3/8) / (n + 1/4)

# Generate normal quantiles by simulation (since we can't use qnorm)
large_normal_sample <- rnorm(1000000) # Large normal reference
theoretical_quantiles <- quantile(large_normal_sample, probs = p)

# Calculate 95% probability envelopes by simulation
n_sim <- 1000 # Number of simulations
envelope_matrix <- matrix(NA, nrow = n_sim, ncol = n)

# Simulate many normal samples of size n
for (j in 1:n_sim) {
  sim_sample <- rnorm(n)
  sim_sorted <- sort(sim_sample)
  envelope_matrix[j, ] <- sim_sorted
}

# Calculate 2.5% and 97.5% percentiles at each position
lower_envelope <- apply(envelope_matrix, 2, quantile, probs = 0.025)
upper_envelope <- apply(envelope_matrix, 2, quantile, probs = 0.975)

# Create the QQ-plot
par(mfrow = c(1, 2)) # Split plot window

# Manual QQ-plot
plot(theoretical_quantiles, sorted_randoms,
     xlab = "Theoretical Normal Quantiles",
     ylab = "Sample Quantiles",
     main = "Manual QQ-plot with 95% Envelopes",
     pch = 19, cex = 0.6,
     ylim = c(-5, 5), xlim = c(-4, 4),
     col = rgb(0, 0, 0, 0.7))

# Add reference line (y = x)
abline(a = 0, b = 1, col = "red", lwd = 2)

# Add 95% probability bands
lines(theoretical_quantiles, lower_envelope,
     col = "blue", lty = 2, lwd = 2)
lines(theoretical_quantiles, upper_envelope,
     col = "blue", lty = 2, lwd = 2)

```

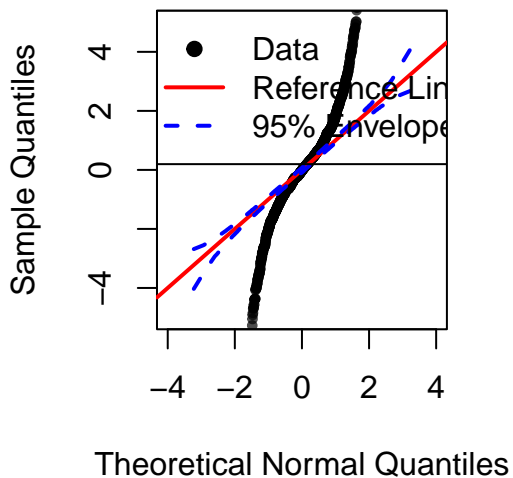
```
# Add legend
legend("topleft",
      legend = c("Data", "Reference Line", "95% Envelope"),
      col = c("black", "red", "blue"),
      pch = c(19, NA, NA),
      lty = c(NA, 1, 2),
      lwd = c(NA, 2, 2))

# Check with car::qqPlot for comparison
library(car)
```

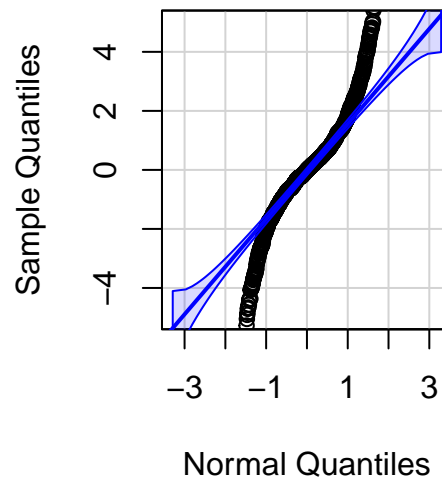
Loading required package: carData

```
qqPlot(randoms, envelope = 0.95, ylim = c(-5, 5),
      main = "car::qqPlot for Comparison",
      xlab = "Normal Quantiles",
      ylab = "Sample Quantiles")
```

Manual QQ-plot with 95% Envelope



car::qqPlot for Comparison



[1] 806 426

3 Day Four Practical

QUESTION ONE

```
library(tidyverse)
```

Warning: package 'tidyverse' was built under R version 4.3.3

Warning: package 'tidyr' was built under R version 4.3.3

Warning: package 'purrr' was built under R version 4.3.3

Warning: package 'dplyr' was built under R version 4.3.3

Warning: package 'forcats' was built under R version 4.3.3

Warning: package 'lubridate' was built under R version 4.3.3

```
-- Attaching core tidyverse packages ----- tidyverse 2.0.0 --
```

```
v dplyr      1.1.4      v readr      2.1.5
```

```
v forcats    1.0.0      v stringr    1.6.0
```

```
v ggplot2    4.0.2      v tibble     3.2.1
```

```
v lubridate  1.9.4      v tidyr      1.3.1
```

```
v purrr      1.0.4
```

```
-- Conflicts ----- tidyverse_conflicts() --
```

```
x dplyr::filter() masks stats::filter()
```

```
x dplyr::lag()     masks stats::lag()
```

```
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become
```

```
library(nycflights13)
```

Warning: package 'nycflights13' was built under R version 4.3.3

```
#The line below renames the first 10 columns of flights and prints the flights data set with
flights |> rename(Year = year, Month = month, Day = day, Dep_Time = dep_time, Sched_Dep_Time
```

```
# A tibble: 336,776 x 19
```

	Year	Month	Day	Dep_Time	Sched_Dep_Time	Dep_Delay	Arr_Time	Sched_Arr_Time
	<int>	<int>	<int>	<int>	<int>	<dbl>	<int>	<int>
1	2013	1	2	2130	2130	0	1	18
2	2013	1	11	2157	2000	117	1	2208
3	2013	1	11	2253	2249	4	1	2357
4	2013	1	14	2122	2130	-8	1	2
5	2013	1	14	2246	2250	-4	1	7
6	2013	1	15	2304	2245	19	1	2357
7	2013	1	16	2018	2025	-7	1	2329
8	2013	1	16	2303	2245	18	1	2357
9	2013	1	19	2107	2110	-3	1	2355
10	2013	1	22	2246	2249	-3	1	2357

```
# i 336,766 more rows
```

```
# i 11 more variables: Arr_Delay <dbl>, Carrier <chr>, flight <int>,
#   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
#   hour <dbl>, minute <dbl>, time_hour <dtm>
```

QUESTION TWO

```
#Creates flight1 tibble that contains flights that occurred in month 1 ONLY
flight1 <- flights |> filter(month == 1)
```

```
#Gets the average distance of each carrier
```

```
carrier_dist_vec_mean <- flight1 |> group_by(carrier) |> summarise(MEAN = mean(distance, na.rm = TRUE))
```

```
#Gets the standard deviation of the distance for each carrier
```

```
carrier_dist_vec_sd <- flight1 |> group_by(carrier) |> summarise(SD = sd(distance, na.rm = TRUE))
```

```
#Creates one tibble with both the mean and sd distance for each carrier
```

```
dist_tbl <- carrier_dist_vec_mean |> left_join(carrier_dist_vec_sd)
```

```
Joining with `by = join_by(carrier)`
```

```
#Arranges the carrier rows from that with the least to the highest mean distance
dist_tbl |> arrange(MEAN)
```

```
# A tibble: 16 x 3
  carrier MEAN    SD
  <chr>   <dbl> <dbl>
1 YV      229     0
2 9E      476.  334.
3 EV      522.  294.
4 US      536.  553.
5 MQ      566.  223.
6 FL      691.  142.
7 OO      733    NA
8 WN      942.  496.
9 B6     1062.  681.
10 DL     1220.  644.
11 AA     1350.  626.
12 UA     1462.  778.
13 F9     1620     0
14 AS     2402     0
15 VX     2495.  98.2
16 HA     4983     0
```

QUESTION THREE

```
##### NA #####
```

```
carrier00 <- flight1 |> filter(carrier == '00') |> select(carrier, distance)

carrier00
```

```
# A tibble: 1 x 2
  carrier distance
  <chr>         <dbl>
1 00           733
```

```
print('Explanation of NA: Carrier 00 has NA as the standard deviation of the distance, that is :')
```

```
[1] "Explanation of NA: Carrier 00 has NA as the standard deviation of the distance, that is :"
```

```
##### ZERO #####
```

```
carrierYV <- flight1 |> filter(carrier == 'YV') |> select(carrier, distance)

carrierYV
```

```
# A tibble: 46 x 2
  carrier distance
  <chr>      <dbl>
1 YV         229
2 YV         229
3 YV         229
4 YV         229
5 YV         229
6 YV         229
7 YV         229
8 YV         229
9 YV         229
10 YV        229
# i 36 more rows
```

```
print('Explanaition of ZERO: The standard deviation of the distance for the YV carrier is zero')
```

```
[1] "Explanaition of ZERO: The standard deviation of the distance for the YV carrier is zero"
```

QUESTION FOUR

```
#Getting the Average Departure delay for each carrier in each month in the long format
switched <- flights |> group_by(carrier, month) |> summarise(Av_Dep_Delay = mean(dep_delay, na.rm = TRUE))
```

`summarise()` has grouped output by 'carrier'. You can override using the `.groups` argument.

```
#Converting into wide data
switched2 <- switched |> pivot_wider(names_from = carrier, values_from = Av_Dep_Delay)

switched2
```

```
# A tibble: 12 x 17
  month `9E` AA AS B6 DL EV F9 FL HA MQ OO
  <int> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1     1 16.9  6.93  7.35  9.49  3.85 24.2  10   1.97 54.4  6.49 67
2     2 16.5  8.28  0.722 13.8  5.54 21.5 29.8  5.18 17.4  8.09 NA
3     3 13.4  8.70  8.42 14.2  9.93 26.2 16.8 17.3  1.16 7.19 NA
4     4 13.6 11.7 11.3 15.2  8.17 22.8 24.6 13.1 -2.1 13.7 NA
5     5 22.7  9.66  6.77  9.78  9.74 20.2 35.9 19.2 -1.45 13.9 NA
```

```

6      6 29.0  14.6  13.1   20.4  18.7  25.5  29.4  38.8   1.47 20.8  61
7      7 31.4  12.1   2.42  24.9  20.6  26.5  31.8  41.2  -1.71 20.7  NA
8      8 17.3   7.17  2.87  15.7   9.85 16.3  22.2  23.4   1.68 10.1  64
9      9  7.75  5.69 -4.52   6.63  5.53  8.24  8.26 16.9  -5.44  5.35 -4.94
10     10 9.33  3.00  0.677  2.96  3.42 13.4   9.70 13.7  -5.10  4.48  NA
11     11 7.56  3.10  3.08   3.52  2.85  9.83 13.5  16.9  -5.44  3.28  0.8
12     12 19.8 11.7  18.0   17.0  10.8 27.9  13.1  26.1  -3.14 12.7  NA
# i 5 more variables: UA <dbl>, US <dbl>, VX <dbl>, WN <dbl>, YV <dbl>

```

QUESTION FIVE

```

#Getting number of all flights with a delayed departure time and arrived with NO delay
meet_Condition <- flights |> filter(dep_delay > 0) |> filter(arr_delay <= 0) |> count()

#Getting number of total flights
total_Flights <- flights |> count()

#Calculating the proportion
prop_Delayed <- meet_Condition/total_Flights

#Renaming the column
prop_Delayed <- prop_Delayed |> rename(Prop_Delayed = n)

prop_Delayed

```

```

  Prop_Delayed
1      0.1052391

```

QUESTION SIX

```

routes <- flights |> group_by(origin, dest) |> summarize(n_airlines = n_distinct(carrier)) |>

```

`summarise()` has grouped output by 'origin'. You can override using the
`.groups` argument.

```

routes

```

```

# A tibble: 128 x 3
# Groups:   origin [3]
  origin dest n_airlines

```

```

      <chr>  <chr>      <int>
1 EWR      ATL         4
2 EWR      AUS         2
3 EWR      BDL         2
4 EWR      BNA         2
5 EWR      BOS         3
6 EWR      BWI         2
7 EWR      CHS         2
8 EWR      CLE         2
9 EWR      CLT         3
10 EWR     CVG         2
# i 118 more rows

```

```

#Joining the routes with at least two airlines with rest of the flights data
joint <- routes |> left_join(flights, by = join_by(origin, dest))

```

```

#Grouping to get each route+carrier combo then calc it's average arrival delay
joint2 <- joint |> group_by(origin, dest, carrier) |> summarise(Av_Arr_Delay = mean(arr_delay))

```

`summarise()` has grouped output by 'origin', 'dest'. You can override using the `.groups` argument.

```
joint2
```

```

# A tibble: 343 x 4
# Groups:   origin, dest [128]
   origin dest  carrier Av_Arr_Delay
   <chr>  <chr> <chr>      <dbl>
1 EWR    ATL   9E         -6.25
2 EWR    ATL   DL          10.00
3 EWR    ATL   EV          19.5
4 EWR    ATL   UA          10.5
5 EWR    AUS   UA           4.28
6 EWR    AUS   WN         -11.2
7 EWR    BDL   EV           6.78
8 EWR    BDL   UA          22.6
9 EWR    BNA   EV          17.7
10 EWR    BNA   WN          -2.13
# i 333 more rows

```

```
# Airline with the best average arrival delay
```

```
best <- joint2 |> group_by(origin, dest) |> summarise(best_av_arr_delay = min(Av_Arr_Delay))
```

`summarise()` has grouped output by 'origin'. You can override using the
`.groups` argument.

```
best2 <- best |> left_join(joint2, join_by(best_av_arr_delay == Av_Arr_Delay)) |> select(!or
```

Warning in left_join(best, joint2, join_by(best_av_arr_delay == Av_Arr_Delay)): Detected an
to-many relationship between `x` and `y`.
i Row 17 of `x` matches multiple rows in `y`.
i Row 41 of `y` matches multiple rows in `x`.
i If a many-to-many relationship is expected, set `relationship =
"many-to-many"` to silence this warning.

```
best2
```

```
# A tibble: 134 x 4
  origin.x dest.x best_av_arr_delay carrier
  <chr>    <chr>          <dbl> <chr>
1 EWR     ATL             -6.25  9E
2 EWR     AUS            -11.2   WN
3 EWR     BDL             6.78   EV
4 EWR     BNA            -2.13   WN
5 EWR     BOS            -4.01   EV
6 EWR     BWI             5.95   WN
7 EWR     CHS            -14     UA
8 EWR     CLE            -3.71   EV
9 EWR     CLT             0.920  US
10 EWR     CVG             1.40   9E
# i 124 more rows
```

```
#Airline with the worst arrival delay
```

```
worst <- joint2 |> group_by(origin, dest) |> summarise(worst_av_arr_delay = max(Av_Arr_Delay))
```

`summarise()` has grouped output by 'origin'. You can override using the
`.groups` argument.

```
worst2 <- worst |> left_join(joint2, join_by(worst_av_arr_delay == Av_Arr_Delay)) |> select(
```

```
Warning in left_join(worst, joint2, join_by(worst_av_arr_delay == Av_Arr_Delay)): Detected an  
to-many relationship between `x` and `y`.  
i Row 75 of `x` matches multiple rows in `y`.  
i Row 194 of `y` matches multiple rows in `x`.  
i If a many-to-many relationship is expected, set `relationship =  
  "many-to-many"` to silence this warning.
```

```
worst2
```

```
# A tibble: 130 x 4  
  origin.x dest.x worst_av_arr_delay carrier  
  <chr>    <chr>          <dbl> <chr>  
1 EWR     ATL             19.5 EV  
2 EWR     AUS              4.28 UA  
3 EWR     BDL             22.6 UA  
4 EWR     BNA             17.7 EV  
5 EWR     BOS              6.87 B6  
6 EWR     BWI             20.1 EV  
7 EWR     CHS             16.2 EV  
8 EWR     CLE              5.97 UA  
9 EWR     CLT             20.5 EV  
10 EWR    CVG             21.2 EV  
# i 120 more rows
```

```
#Combining
```

```
results <- best2 |> left_join(worst2, join_by(origin.x, dest.x))
```

```
Warning in left_join(best2, worst2, join_by(origin.x, dest.x)): Detected an unexpected many-  
to-many relationship between `x` and `y`.  
i Row 77 of `x` matches multiple rows in `y`.  
i Row 17 of `y` matches multiple rows in `x`.  
i If a many-to-many relationship is expected, set `relationship =  
  "many-to-many"` to silence this warning.
```

```
results
```

```
# A tibble: 136 x 6
  origin.x dest.x best_av_arr_delay carrier.x worst_av_arr_delay carrier.y
  <chr>    <chr>          <dbl> <chr>          <dbl> <chr>
1 EWR     ATL           -6.25 9E           19.5 EV
2 EWR     AUS          -11.2 WN           4.28 UA
3 EWR     BDL           6.78 EV          22.6 UA
4 EWR     BNA          -2.13 WN          17.7 EV
5 EWR     BOS          -4.01 EV           6.87 B6
6 EWR     BWI           5.95 WN          20.1 EV
7 EWR     CHS          -14    UA          16.2 EV
8 EWR     CLE          -3.71 EV           5.97 UA
9 EWR     CLT           0.920 US          20.5 EV
10 EWR     CVG           1.40 9E          21.2 EV
# i 126 more rows
```

```
results2 <- results |> mutate(diff = sqrt((best_av_arr_delay - worst_av_arr_delay)**2))

results3 <- results2 |> summarise(greatest_diff = max(diff, na.rm = TRUE)) |> left_join(results, results3)

results3
```

```
# A tibble: 1 x 7
  greatest_diff origin.x dest.x best_av_arr_delay carrier.x worst_av_arr_delay
      <dbl> <chr>    <chr>          <dbl> <chr>          <dbl>
1      127. JFK     ATL           1.40 9E          128
# i 1 more variable: carrier.y <chr>
```

```
#Trying to find the reason

reason <- results |> mutate(diff = sqrt((best_av_arr_delay - worst_av_arr_delay)**2))

reason2 <- reason |> summarise(terrible = max(worst_av_arr_delay, na.rm = TRUE))

reason2
```

```
# A tibble: 1 x 1
  terrible
    <dbl>
1      128
```

```
print('The above code shows that the worst average arrival delay was 128, it so happens that
```

```
[1] "The above code shows that the worst average arrival delay was 128, it so happens that t
```

QUESTION SEVEN

```
given_data <- structure(list(id = c("id_1", "id_2", "id_3", "id_4", "id_5",  
"id_6", "id_7", "id_8", "id_9", "id_10", "id_11", "id_12", "id_13",  
"id_14", "id_15", "id_16", "id_17", "id_18", "id_19", "id_20",  
"id_21", "id_22", "id_23", "id_24", "id_25", "id_26", "id_27",  
"id_28", "id_29", "id_30", "id_31", "id_32", "id_33", "id_34",  
"id_35", "id_36", "id_37", "id_38", "id_39", "id_40", "id_41",  
"id_42", "id_43", "id_44", "id_45", "id_46", "id_47", "id_48",  
"id_49", "id_50"), age = c(50L, 34L, 70L, 33L, 22L, 61L, 69L,  
73L, 62L, 56L, 71L, 33L, 73L, 44L, 45L, 46L, 24L, 70L, 46L, 76L,  
47L, 76L, 28L, 48L, 54L, 27L, 45L, 26L, 61L, 28L, 38L, 55L, 33L,  
36L, 62L, 58L, 72L, 31L, 34L, 51L, 61L, 64L, 26L, 28L, 60L, 29L,  
42L, 46L, 79L, 72L), gender = c("male", "male", "male", "female",  
"female", "male", "female", "male", "male", "female", "female",  
"male", "male", "female", "male", "male", "male", "male", "female",  
"male", "male", "male", "male", "female", "femal", "male", "female",  
"female", "female", "female", "male", "female", "female", "female",  
"male", "male", "female", "male", "female", "female", "male",  
"female", "female", "male", "male", "female", "male", "male",  
"male", "female"), height = c(174.4, 197.7, 174.1, 194.5, NA,  
180.4, 170.5, 157.4, 196.8, 165.1, 153, 197.4, 186, 157.1, 177.5,  
197.7, 179.3, 170.2, 182.4, NA, 165.4, 161, 168.5, 199.2, 157.7,  
154.6, 157.1, 184.5, 181, 194.6, 183.6, 186.9, 176.1, 183, 191.1,  
189.3, 199, 172, 165.6, 170.5, 150.5, 159.2, 192.1, 161.6, 162,  
153.8, 162.3, 186.6, 192.4, 174.9), weight = c(69.4, 62.3, 55.6,  
69.5, 78.6, 60.8, 72.2, 60.9, 75.1, 67.7, 82.5, 68.7, 67.8, 76.7,  
87, 61.1, 70.6, 63.3, 81.5, 59.2, 93.2, 87.3, 83.4, 80.9, 68.6,  
76.5, 93.7, 79.1, 92, 65.6, 85.4, 63.3, 79.7, 74.1, 63.3, 78.2,  
95.7, 95.1, 63.7, 66.1, 99.3, 81, 96.9, 73.3, 70.3, 83, 57.6,  
78.6, 61.9, 98.1), blood_type = c("O", "A", "O", "O", "B", "AB",  
"O", "O", "O", "AB", "A", "O", "O", "O", "B", "A", "B", "AB",  
"O", "AB", "A", "AB", "O", "B", "A", "A", "B", "AB", "A", "B",  
"B", "A", "O", "O", "O", "B", "O", "A", "A", "B", "A", "O", "AB",  
"A", "A", "O", "O", "B", "A", "O"), disease_status = c("diseased",  
"healthy", "healthy", "healthy", "healthy", "healthy", "diseased",  
"healthy", "diseased", "Healthy", "diseased", "healthy", "diseased",
```

```

"healthy", "diseased", "healthy", "healthy", "healthy", "healthy",
"healthy", "healthy", "diseased", "healthy", "diseased", "healthy",
"healthy", "healthy", "healthy", "diseased", "diseased", "healthy",
"healthy", "healthy", "diseased", "diseased", "diseased", "healthy",
"diseased", "healthy", "healthy", "healthy", "healthy", "healthy",
"diseased", "diseased", "diseased", "healthy", "healthy", "diseased",
"diseased"), cholesterol = c(228, 223, 213, 198, 166, 151, 195,
199, 189, 196, 221, 156, 185, 230, 234, 174, 185, 236, 235, 180,
165, 220, 160, 153, 250, 153, 184, 242, 212, 179, 224, 233, 181,
199, 220, 214, 214, 248, 191, 162, 203, 173, 199, 187, 248, 189,
173, 212, 164, 247), glucose = c(96, 78, 101, 119, 103, 91, 86,
NA, 77, 80, 115, 85, 88, 109, NA, 71, 90, 94, 91, 87, 113, 93,
97, 118, 109, 80, 85, 119, 99, 108, 89, 108, 97, 116, 79, 84,
75, 81, 119, NA, 106, 109, 75, 82, 84, 75, 76, 120, 119, 77),
smoker = c("yes", "yes", "yes", "yes", "no", "yes", "no",
"yes", "no", "no", "no", "no", "no", "yes", "no", "yes",
"yes", "yes", "yes", "yes", "yes", "yes", "yes", "yes", "no",
"no", "yes", "yes", "yes", "no", "no", "yes", "no", "yes",
"no", "yes", "no", "yes", "yes", "yes", "no", "no", "yes",
"no", "no", "no", "no", "no", "no", "yes"), exercise = c("occasional",
"regular", "occasional", "regular", "none", "occasional",
"regular", "none", "occasional", "none", "occasional", "none",
"none", "regular", "occasional", "none", "regular", "regular",
"none", "occasional", "none", "occasional", "occasional",
"occasional", "regular", "occasional", "regular", "regular",
"regular", "occasional", "occasional", "none", "none", "regular",
"occasional", "occasional", "none", "none", "none", "none",
"occasional", "regular", "regular", "none", "regular", "occasional",
"occasional", "none", "occasional", "regular"), income = c(84820L,
81547L, 22588L, 72490L, 74533L, 25338L, 41469L, 57315L, 63629L,
88662L, 62615L, 56261L, 58499L, 82232L, 77584L, 77275L, 38468L,
54510L, 91326L, 78611L, 31402L, 29586L, 21441L, 58269L, 84173L,
88295L, 37940L, 43750L, 69750L, 92356L, 82518L, 91455L, 68866L,
51178L, 68275L, 27689L, 35418L, 81318L, 62405L, 86851L, 25654L,
47553L, 74474L, 51409L, 22607L, 55360L, 96351L, 21516L, 41927L,
55810L), education = c("master", "bachelor", "PhD", "master",
"bachelor", "highschool", "PhD", "highschool", "PhD", "PhD",
"bachelor", "highschool", "master", "bachelor", "PhD", "PhD",
"PhD", "bachelor", "master", "highschool", "PhD", "highschool",
"bachelor", "master", "highschool", "highschool", "master",
"master", "bachelor", "PhD", "highschool", "PhD", "master",
"master", "master", "PhD", "highschool", "master", "master",

```

```

"highschool", "bachelor", "highschool", "bachelor", "PhD",
"bachelor", "highschool", "master", "highschool", "bachelor",
"bachelor"), region = c("North", "South", "North", "West",
"North", "West", "South", "South", "West", "South", "West",
"South", "West", "East", "North", "West", "North", "North",
"West", "North", "East", "West", "South", "North", "North",
"East", "East", "North", "North", "West", "South", "West",
"West", "East", "West", "North", "West", "North", "East",
"North", "West", "South", "South", "East", "North", "West",
"West", "East", "North", "East"), marital_status = c("divorced",
"single", "divorced", "divorced", "divorced", "divorced",
"divorced", "married", "divorced", "married", "divorced",
"widowed", "married", "single", "widowed", "widowed", "single",
"divorced", "widowed", "widowed", "single", "married", "single",
"married", "widowed", "married", "single", "single", "widowed",
"married", "widowed", "divorced", "single", "married", "single",
"widowed", "widowed", "married", "widowed", "divorced", "married",
"married", "divorced", "single", "married", "widowed", "divorced",
"divorced", "single", "divorced")), row.names = c(NA, -50L
), class = c("tbl_df", "tbl", "data.frame"))

library(UtillsDataRSV)
see <- given_data |> view_cols()

```

```

[1] "id"
[1] "id_16" "id_9"  "id_1"  "id_20" "id_50" "id_48" "id_14" "id_41" "id_47"
[10] "id_27" "id_21" "id_40" "id_29" "id_25" "id_45" "id_15" "id_18" "id_11"
[19] "id_6"  "id_2"
[1] "30 unique entries not displayed"
[1] "-----"
[1] "age"
[1] 71 60 33 70 47
[1] "-----"
[1] "gender"
[1] "female" "male"   "femal"
[1] "-----"
[1] "height"
[1] 161.0 177.5 180.4 182.4    NA
[1] "-----"
[1] "weight"
[1] 99.3 79.7 87.3 93.2 79.1
[1] "-----"

```

```

[1] "blood_type"
[1] "B" "A" "AB" "O"
[1] "-----"
[1] "disease_status"
[1] "Healthy" "diseased" "healthy"
[1] "-----"
[1] "cholesterol"
[1] 162 198 223 189 185
[1] "-----"
[1] "glucose"
[1] 109 115 80 116 NA
[1] "-----"
[1] "smoker"
[1] "yes" "no"
[1] "-----"
[1] "exercise"
[1] "occasional" "regular" "none"
[1] "-----"
[1] "income"
[1] 51409 77275 69750 56261 27689
[1] "-----"
[1] "education"
[1] "bachelor" "highschool" "PhD" "master"
[1] "-----"
[1] "region"
[1] "West" "East" "North" "South"
[1] "-----"
[1] "marital_status"
[1] "divorced" "widowed" "single" "married"
[1] "-----"

```

Warning: Not all unique entries displayed for these non-numeric cols: id

```
print('LIST OF ERRORS: ')
```

```
[1] "LIST OF ERRORS: "
```

```
print('From the above we can see that the gender column has a typo: \'femal\' instead of \'f'
```

```
[1] "From the above we can see that the gender column has a typo: 'femal' instead of 'female"
```

```
print(' The height column has some missing entries.')
```

```
[1] " The height column has some missing entries."
```

```
print(' The disease_status column has a typo, \'Healthy\' and \'healthy\'.')
```

```
[1] " The disease_status column has a typo, 'Healthy' and 'healthy'."
```

```
print(' The glucose column has some missing entries.')
```

```
[1] " The glucose column has some missing entries."
```