

HW-1-utkarshapatil01

Utkarsha Patil

R HW-1 - Utkarsha Patil

Getting to know your Data with R

Installing Packages

```
if ( !require('pacman'))  
  install.packages('pacman')  
  
library(pacman)  
  
p_load(dlookr, # Data inspection and exploration  
       DMwR2, # Data Mining with R functions  
       GGally, # Pair-wise plots using ggplot2  
       Hmisc, # Data analysis  
       palmerpenguins, # Alternative to the Iris dataset  
       tidyverse) # Data wrangling, manipulation, visualization
```

Loading Data

Following command will load the **algae** dataset into your R session, making it available for analysis.

```
data(algae, package = "DMwR2") # Loading the algae dataset from the "DMwR2" package  
  
algae <- algae # Saving dataset in the variable
```

`glimpse()` will give you information about the dataset's variables, their data types, and the first few rows of data to help you understand its structure.

```
algae |> glimpse() # Data inspection
```

```
Rows: 200
Columns: 18
$ season <fct> winter, spring, autumn, spring, autumn, winter, summer, autumn, ~
$ size   <fct> small, small, small, small, small, small, small, small, small, ~
$ speed  <fct> medium, medium, medium, medium, medium, high, high, high, mediu~
$ mxPH   <dbl> 8.00, 8.35, 8.10, 8.07, 8.06, 8.25, 8.15, 8.05, 8.70, 7.93, 7.7~
$ mnO2   <dbl> 9.8, 8.0, 11.4, 4.8, 9.0, 13.1, 10.3, 10.6, 3.4, 9.9, 10.2, 11.~
$ Cl     <dbl> 60.800, 57.750, 40.020, 77.364, 55.350, 65.750, 73.250, 59.067, ~
$ NO3    <dbl> 6.238, 1.288, 5.330, 2.302, 10.416, 9.248, 1.535, 4.990, 0.886, ~
$ NH4    <dbl> 578.000, 370.000, 346.667, 98.182, 233.700, 430.000, 110.000, 2~
$ oPO4   <dbl> 105.000, 428.750, 125.667, 61.182, 58.222, 18.250, 61.250, 44.6~
$ PO4    <dbl> 170.000, 558.750, 187.057, 138.700, 97.580, 56.667, 111.750, 77~
$ Chla   <dbl> 50.000, 1.300, 15.600, 1.400, 10.500, 28.400, 3.200, 6.900, 5.5~
$ a1     <dbl> 0.0, 1.4, 3.3, 3.1, 9.2, 15.1, 2.4, 18.2, 25.4, 17.0, 16.6, 32.~
$ a2     <dbl> 0.0, 7.6, 53.6, 41.0, 2.9, 14.6, 1.2, 1.6, 5.4, 0.0, 0.0, 0.0, ~
$ a3     <dbl> 0.0, 4.8, 1.9, 18.9, 7.5, 1.4, 3.2, 0.0, 2.5, 0.0, 0.0, 0.0, 2.~
$ a4     <dbl> 0.0, 1.9, 0.0, 0.0, 0.0, 0.0, 3.9, 0.0, 0.0, 2.9, 0.0, 0.0, 0.0~
$ a5     <dbl> 34.2, 6.7, 0.0, 1.4, 7.5, 22.5, 5.8, 5.5, 0.0, 0.0, 1.2, 0.0, 1~
$ a6     <dbl> 8.3, 0.0, 0.0, 0.0, 4.1, 12.6, 6.8, 8.7, 0.0, 0.0, 0.0, 0.0, 0.~
$ a7     <dbl> 0.0, 2.1, 9.7, 1.4, 1.0, 2.9, 0.0, 0.0, 0.0, 1.7, 6.0, 1.5, 2.1~
```

Central tendency: mean, median, mode

Central tendency measures are used to describe the central or typical value in a dataset. The main measures of central tendency are the mean, median, and mode:

1. Mean

The mean, also known as the average, is calculated by adding up all the values in a dataset and then dividing by the number of values. It is the most common measure of central tendency.

the `|>` symbol represents the pipe operator, which is used for chaining or piping operations in a way that enhances code readability and maintainability.

```
algae$a1 |>
  mean() # Selecting a1 col from dataset and passing value to mean() function using pipe o
```

```
[1] 16.9235
```

2. Median

The median is the middle value in a dataset when it is arranged in ascending or descending order. If there is an even number of values, the median is the average of the two middle values. The median is less affected by extreme outliers compared to the mean.

```
algae$a1 |>
  median() # Selecting a1 col from dataset and passing value to median() function using pipe
```

```
[1] 6.95
```

3. Mode

The mode is the value that appears most frequently in a dataset. A dataset can have no mode if all values occur with the same frequency (i.e., it's multimodal), one mode if a single value occurs most frequently (unimodal), or multiple modes if multiple values have the same highest frequency.

```
Mode <- function(x, na.rm=FALSE){
  if(na.rm) x<-x[!is.na(x)]
  ux <- unique (x)
  return (ux[which.max(tabulate(match(x, ux)))])
}

algae$a2 |> Mode()
```

```
[1] 0
```

Using the table() Function:

One common way to find the mode is by using the **table()** function to create a frequency table of the values in your dataset. Then, you can find the value(s) with the highest frequency.

```
# Create a frequency table
freq_table <- table(algae$a2)

# Find the mode(s)
modes <- names(freq_table)[freq_table == max(freq_table)]
cat("Mode(s):", modes, "\n")
```

```
Mode(s): 0
```

DMwRcentralValue() function:

```
# Numerical variable  
algae$a1 |> centralValue()
```

```
[1] 6.95
```

```
# Nominal variable  
algae$speed |> centralValue()
```

```
[1] "high"
```

Statistics of spread (variation)

describe how data points in a dataset are spread out or dispersed. These measures provide insights into the extent to which data points deviate from the central tendency and give a sense of the data's variability.

Variance

Variance measures the average squared deviation of each data point from the mean.

```
algae$a1 |> var() # Selecting a1 col from dataset and passing value to var() function using
```

```
[1] 455.7532
```

Standard deviation

The standard deviation is the square root of the variance. It provides a measure of dispersion in the same units as the data, making it easier to interpret.

```
algae$a1 |> sd() # Selecting a1 col from dataset and passing value to sd() function using
```

```
[1] 21.34838
```

Range

The range is the simplest measure of spread and is calculated as the difference between the maximum and minimum values in the dataset. It provides an idea of how spread out the data is but can be heavily influenced by outliers.

```
algae$a1 |> range() # Selecting a1 col from dataset and passing value to range() function
```

```
[1] 0.0 89.8
```

Maximum value

```
algae$a1 |> max() # Selecting a1 col from dataset and passing value to max() function using
```

```
[1] 89.8
```

Minimum value

```
algae$a1 |> min() # Selecting a1 col from dataset and passing value to min() function using
```

```
[1] 0
```

Quantiles

Quantiles are the set of values/points that divides the dataset into groups of equal size.

```
algae$a1 |> quantile() # Selecting a1 col from dataset and passing value to quantile() function
```

```
 0%   25%   50%   75%  100%  
0.00  1.50  6.95 24.80 89.80
```

Specifying specific quantiles:

```
algae$a1 |> quantile(probs = c(0.2, 0.8)) # Calculating specific i.e. 20th and 80th quantiles
```

```
20%   80%  
1.20 32.18
```

Interquartile range

The IQR is the range between the first quartile (25th percentile) and the third quartile (75th percentile) of the data. It measures the spread of the middle 50% of the data and is robust to outliers.

```
algae$a1 |> IQR() # Selecting a1 col from dataset and passing value to IQR() function using
```

```
[1] 23.3
```

Missing values

Missing values are typically represented as **NA** (which stands for “Not Available”). There are various ways to identify, handle, and analyze missing values in a dataset.

```
if ( !require('purrr'))  
  install.packages('purrr')  
library('purrr')
```

`purrr::map_dbl(~sum(is.na(.)))`: This code applies the function `~sum(is.na(.))` to each column of a dataset. The `~` is used to create a formula or lambda function. Inside the function, `sum(is.na(.))` calculates the sum of missing values (**NA**) in each column.

```
# Compute the total number of NA values in the dataset  
nas <- algae %>%  
  purrr::map_dbl(~sum(is.na(.))) %>%  
  sum()  
  
cat("The dataset contains ", nas, "NA values. \n")
```

The dataset contains 33 NA values.

```
# Compute the number of incomplete rows in the dataset  
incomplete_rows <- algae %>%  
  summarise_all(~!complete.cases(.)) %>%  
  nrow()  
  
cat("The dataset contains ", incomplete_rows, "(out of ", nrow(algae),") incomplete rows.
```

The dataset contains 200 (out of 200) incomplete rows.

Summaries of a dataset

the `summary()` function is used to generate a statistical summary of numeric or complex data. It provides various statistics for each variable or column in a dataset.

```
algae |> summary() # Selecting a1 col from dataset and passing value to summary() function
```

season	size	speed	mxPH	mn02
autumn:40	large :45	high :84	Min. :5.600	Min. : 1.500
spring:53	medium:84	low :33	1st Qu.:7.700	1st Qu.: 7.725
summer:45	small :71	medium:83	Median :8.060	Median : 9.800
winter:62			Mean :8.012	Mean : 9.118
			3rd Qu.:8.400	3rd Qu.:10.800
			Max. :9.700	Max. :13.400
			NA's :1	NA's :2

C1	N03	NH4	oP04
Min. : 0.222	Min. : 0.050	Min. : 5.00	Min. : 1.00
1st Qu.: 10.981	1st Qu.: 1.296	1st Qu.: 38.33	1st Qu.: 15.70
Median : 32.730	Median : 2.675	Median : 103.17	Median : 40.15
Mean : 43.636	Mean : 3.282	Mean : 501.30	Mean : 73.59
3rd Qu.: 57.824	3rd Qu.: 4.446	3rd Qu.: 226.95	3rd Qu.: 99.33
Max. :391.500	Max. :45.650	Max. :24064.00	Max. :564.60
NA's :10	NA's :2	NA's :2	NA's :2

P04	Chla	a1	a2
Min. : 1.00	Min. : 0.200	Min. : 0.00	Min. : 0.000
1st Qu.: 41.38	1st Qu.: 2.000	1st Qu.: 1.50	1st Qu.: 0.000
Median :103.29	Median : 5.475	Median : 6.95	Median : 3.000
Mean :137.88	Mean : 13.971	Mean :16.92	Mean : 7.458
3rd Qu.:213.75	3rd Qu.: 18.308	3rd Qu.:24.80	3rd Qu.:11.375
Max. :771.60	Max. :110.456	Max. :89.80	Max. :72.600
NA's :2	NA's :12		

a3	a4	a5	a6
Min. : 0.000	Min. : 0.000	Min. : 0.000	Min. : 0.000
1st Qu.: 0.000	1st Qu.: 0.000	1st Qu.: 0.000	1st Qu.: 0.000
Median : 1.550	Median : 0.000	Median : 1.900	Median : 0.000
Mean : 4.309	Mean : 1.992	Mean : 5.064	Mean : 5.964
3rd Qu.: 4.925	3rd Qu.: 2.400	3rd Qu.: 7.500	3rd Qu.: 6.925
Max. :42.800	Max. :44.600	Max. :44.400	Max. :77.600

a7
Min. : 0.000
1st Qu.: 0.000

```

Median : 1.000
Mean    : 2.495
3rd Qu.: 2.400
Max.    :31.600

```

The `describe()` function provides a comprehensive summary of various aspects of your data, including measures of central tendency, dispersion, distributions, and more.

```

data("penguins")
penguins |> Hmisc::describe()

```

penguins

```

8 Variables      344 Observations
-----
species
      n missing distinct
  344      0         3

Value      Adelie Chinstrap   Gentoo
Frequency      152       68     124
Proportion    0.442    0.198    0.360
-----
island
      n missing distinct
  344      0         3

Value      Biscoe   Dream Torgersen
Frequency      168     124       52
Proportion    0.488    0.360    0.151
-----
bill_length_mm
      n missing distinct      Info      Mean      Gmd      .05      .10
  342      2      164        1    43.92    6.274    35.70    36.60
  .25    .50    .75    .90    .95
 39.23  44.45  48.50  50.80  51.99

lowest : 32.1 33.1 33.5 34   34.1, highest: 55.1 55.8 55.9 58   59.6
-----
bill_depth_mm

```


n	missing	distinct	Info	Mean	Gmd	.05	.10
342	2	80	1	17.15	2.267	13.9	14.3
.25	.50	.75	.90	.95			
15.6	17.3	18.7	19.5	20.0			

lowest : 13.1 13.2 13.3 13.4 13.5, highest: 20.7 20.8 21.1 21.2 21.5

flipper_length_mm

n	missing	distinct	Info	Mean	Gmd	.05	.10
342	2	55	0.999	200.9	16.03	181.0	185.0
.25	.50	.75	.90	.95			
190.0	197.0	213.0	220.9	225.0			

lowest : 172 174 176 178 179, highest: 226 228 229 230 231

body_mass_g

n	missing	distinct	Info	Mean	Gmd	.05	.10
342	2	94	1	4202	911.8	3150	3300
.25	.50	.75	.90	.95			
3550	4050	4750	5400	5650			

lowest : 2700 2850 2900 2925 2975, highest: 5850 5950 6000 6050 6300

sex

n	missing	distinct
333	11	2

Value	female	male
-------	--------	------

Frequency	165	168
-----------	-----	-----

Proportion	0.495	0.505
------------	-------	-------

year

n	missing	distinct	Info	Mean	Gmd
344	0	3	0.888	2008	0.8919

Value	2007	2008	2009
-------	------	------	------

Frequency	110	114	120
-----------	-----	-----	-----

Proportion	0.320	0.331	0.349
------------	-------	-------	-------

For the frequency table, variable is rounded to the nearest 0.02

dlookr's describe()

```
penguins |> dlookr::describe()

# A tibble: 5 x 26
  described_variables      n    na  mean      sd se_mean      IQR skewness
  <chr>          <int> <int> <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
1 bill_length_mm      342     2  43.9    5.46    0.295    9.27    0.0531
2 bill_depth_mm       342     2  17.2    1.97    0.107     3.1   -0.143
3 flipper_length_mm   342     2  201.   14.1    0.760    23     0.346
4 body_mass_g         342     2 4202.   802.    43.4   1200     0.470
5 year               344     0 2008.    0.818   0.0441     2   -0.0537
# i 18 more variables: kurtosis <dbl>, p00 <dbl>, p01 <dbl>, p05 <dbl>,
#   p10 <dbl>, p20 <dbl>, p25 <dbl>, p30 <dbl>, p40 <dbl>, p50 <dbl>,
#   p60 <dbl>, p70 <dbl>, p75 <dbl>, p80 <dbl>, p90 <dbl>, p95 <dbl>,
#   p99 <dbl>, p100 <dbl>
```

Summaries on a subset of data

The **summarize()** function is used for aggregating and summarizing data in a data frame or data table.

```
algae |>
  summarise(avgN03 = mean(N03, na.rm=TRUE),
            medA1 = median(a1))
```

```
# A tibble: 1 x 2
  avgN03 medA1
  <dbl> <dbl>
1   3.28  6.95
```

The **summarise_all()** function is used to apply a summary function to all columns in a data frame, resulting in a summary statistic or value for each column. This is particularly useful when you want to calculate summary statistics for multiple columns simultaneously.

```
algae |>
  select(mxPH:C1) |> # Select columns from mxPH to C1 and calculate mean and median
  summarise_all(list(mean, median), na.rm = TRUE)
```

```
# A tibble: 1 x 6
  mxPH_fn1 mn02_fn1 Cl_fn1 mxPH_fn2 mn02_fn2 Cl_fn2
    <dbl>    <dbl>  <dbl>    <dbl>    <dbl>  <dbl>
1      8.01      9.12  43.6      8.06      9.8   32.7
```

```
algae |>
  select(a1:a7) |>
  summarise_all(funs(var))
```

Warning: `funs()` was deprecated in dplyr 0.8.0.
i Please use a list of either functions or lambdas:

```
# Simple named list: list(mean = mean, median = median)

# Auto named with `tibble::lst()`: tibble::lst(mean, median)

# Using lambdas list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
```

```
# A tibble: 1 x 7
  a1    a2    a3    a4    a5    a6    a7
  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1  456.  122.  48.3  19.5  56.1  136.  26.6
```

```
algae |>
  select(a1:a7) |>
  summarise_all(c("min", "max"))
```

```
# A tibble: 1 x 14
  a1_min a2_min a3_min a4_min a5_min a6_min a7_min a1_max a2_max a3_max a4_max
  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
1      0      0      0      0      0      0      0  89.8  72.6  42.8  44.6
# i 3 more variables: a5_max <dbl>, a6_max <dbl>, a7_max <dbl>
```

Use summarise() with group_by()

The **summarise()** function in combination with **group_by()** is a powerful tool for performing group-wise summarization of data in R.

```
# Group the dataset by the "season" and "size" columns
# Then, calculate the number of observations (nObs) and the median of the "a7" column (mA7)
algae |>
  group_by(season, size) |>
  summarise(nObs = n(), mA7 = median(a7))
```

```
# A tibble: 12 x 4
# Groups:   season [4]
  season size    nObs  mA7
  <fct>  <fct>  <int> <dbl>
1 autumn large    11  0
2 autumn medium   16 1.05
3 autumn small    13  0
4 spring large    12 1.95
5 spring medium   21  1
6 spring small    20  0
7 summer large    10  0
8 summer medium   21  1
9 summer small    14 1.45
10 winter large    12  0
11 winter medium   26 1.4
12 winter small    24  0
```

```
# Group the dataset by the "species" column
# Then, calculate the variance of the "bill_length_mm" column within each group, ignoring
penguins |>
  group_by(species) |>
  summarise(var = var(bill_length_mm, na.rm = TRUE))
```

```
# A tibble: 3 x 2
  species    var
  <fct>    <dbl>
1 Adelie    7.09
2 Chinstrap 11.2
3 Gentoo    9.50
```

Aggregating data

Aggregating data in R typically involves summarizing, grouping, or transforming data to obtain meaningful insights or to prepare it for further analysis.

```
penguins |>
  group_by(species) |> # group the "penguins" dataset by the "species" column
  reframe(var = quantile(bill_length_mm, na.rm = TRUE))
```

```
# A tibble: 15 x 2
  species      var
  <fct>      <dbl>
1 Adelie    32.1
2 Adelie    36.8
3 Adelie    38.8
4 Adelie    40.8
5 Adelie     46
6 Chinstrap 40.9
7 Chinstrap 46.3
8 Chinstrap 49.6
9 Chinstrap 51.1
10 Chinstrap 58
11 Gentoo   40.9
12 Gentoo   45.3
13 Gentoo   47.3
14 Gentoo   49.6
15 Gentoo   59.6
```

```
penguins |>
  group_by(species) |>
  dlookr::describe(bill_length_mm)
```

```
# A tibble: 3 x 27
  described_variables species      n    na mean    sd se_mean  IQR skewness
  <chr>              <fct>  <int> <int> <dbl> <dbl>  <dbl> <dbl>  <dbl>
1 bill_length_mm    Adelie   151     1  38.8  2.66  0.217  4      0.162
2 bill_length_mm    Chinstrap  68     0  48.8  3.34  0.405  4.73 -0.0906
3 bill_length_mm    Gentoo   123     1  47.5  3.08  0.278  4.25  0.651
# i 18 more variables: kurtosis <dbl>, p00 <dbl>, p01 <dbl>, p05 <dbl>,
#   p10 <dbl>, p20 <dbl>, p25 <dbl>, p30 <dbl>, p40 <dbl>, p50 <dbl>,
```

```
# p60 <dbl>, p70 <dbl>, p75 <dbl>, p80 <dbl>, p90 <dbl>, p95 <dbl>,
# p99 <dbl>, p100 <dbl>
```

[Advanced]

Getting to know your dataset:

1. List data types of the attributes in your tidy dataset

-> **str()** function to inspect the structure of the dataset and see the data types of each attribute:

```
str(penguins)
```

```
tibble [344 x 8] (S3: tbl_df/tbl/data.frame)
 $ species      : Factor w/ 3 levels "Adelie","Chinstrap",...: 1 1 1 1 1 1 1 1 1 1 ...
 $ island       : Factor w/ 3 levels "Biscoe","Dream",...: 3 3 3 3 3 3 3 3 3 3 ...
 $ bill_length_mm : num [1:344] 39.1 39.5 40.3 NA 36.7 39.3 38.9 39.2 34.1 42 ...
 $ bill_depth_mm  : num [1:344] 18.7 17.4 18 NA 19.3 20.6 17.8 19.6 18.1 20.2 ...
 $ flipper_length_mm: int [1:344] 181 186 195 NA 193 190 181 195 193 190 ...
 $ body_mass_g    : int [1:344] 3750 3800 3250 NA 3450 3650 3625 4675 3475 4250 ...
 $ sex           : Factor w/ 2 levels "female","male": 2 1 1 NA 1 2 1 2 NA NA ...
 $ year          : int [1:344] 2007 2007 2007 2007 2007 2007 2007 2007 2007 2007 ...
```

2. Check for skewness in data distribution in the attributes

```
if ( !require('e1071'))
  install.packages('e1071')

# Load necessary libraries (choose one of the packages)
library(e1071) # For e1071 package

# Select numeric attributes from the penguins dataset
numeric_attributes <- penguins %>%
  select_if(is.numeric)

# Calculate skewness for each numeric attribute
skewness_values <- sapply(numeric_attributes, skewness, na.rm = TRUE)

# Create a data frame to display the results
skewness_df <- data.frame(
```

```

    Attribute = names(skewness_values),
    Skewness = skewness_values
  )

```

```

# Print the skewness values
print(skewness_df)

```

	Attribute	Skewness
bill_length_mm	bill_length_mm	0.05265303
bill_depth_mm	bill_depth_mm	-0.14220862
flipper_length_mm	flipper_length_mm	0.34265545
body_mass_g	body_mass_g	0.46621168
year	year	-0.05326012

3. Check for correlations among attributes

```

# Select numeric attributes from the penguins dataset
numeric_attributes <- penguins %>%
  select_if(is.numeric)

# Calculate the correlation matrix
correlation_matrix <- cor(numeric_attributes, use = "complete.obs")

print(correlation_matrix)

```

	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g
bill_length_mm	1.00000000	-0.23505287	0.6561813	0.59510982
bill_depth_mm	-0.23505287	1.00000000	-0.5838512	-0.47191562
flipper_length_mm	0.65618134	-0.58385122	1.00000000	0.87120177
body_mass_g	0.59510982	-0.47191562	0.8712018	1.00000000
year	0.05454458	-0.06035364	0.1696751	0.04220939

	year
bill_length_mm	0.05454458
bill_depth_mm	-0.06035364
flipper_length_mm	0.16967511
body_mass_g	0.04220939
year	1.00000000

4. Examine the extent of missing data. What would be the best way to deal with the missing data in this case?

```
# Check for missing data in the penguins dataset
missing_data <- penguins %>%
  summarise_all(~ sum(is.na(.)))

# Print the number of missing values in each column
print(missing_data)
```

```
# A tibble: 1 x 8
  species island bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
  <int>   <int>         <int>         <int>           <int>         <int>
1      0     0           2             2             2             2
# i 2 more variables: sex <int>, year <int>
```

Dropping: If you have a few missing values in specific columns or rows and the missings are random, you can choose to remove rows or columns with missing data using functions like `na.omit()` or `drop_na()`.

```
# Remove rows with missing values from the penguins dataset
removena_penguins <- drop_na(penguins)
penguins
```

```
# A tibble: 344 x 8
  species island bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
  <fct>   <fct>         <dbl>         <dbl>           <int>         <int>
1 Adelie Torgersen      39.1          18.7            181          3750
2 Adelie Torgersen      39.5          17.4            186          3800
3 Adelie Torgersen      40.3           18             195          3250
4 Adelie Torgersen      NA             NA              NA             NA
5 Adelie Torgersen      36.7          19.3            193          3450
6 Adelie Torgersen      39.3          20.6            190          3650
7 Adelie Torgersen      38.9          17.8            181          3625
8 Adelie Torgersen      39.2          19.6            195          4675
9 Adelie Torgersen      34.1          18.1            193          3475
10 Adelie Torgersen      42            20.2            190          4250
# i 334 more rows
# i 2 more variables: sex <fct>, year <int>
```

```
# Check for missing data in the penguins dataset
missing_data <- penguins %>%
```



```

    summarise_all(~ sum(is.na(.)))

# Print the number of missing values in each column
print(missing_data)

# A tibble: 1 x 8
  species island bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
  <int>   <int>         <int>         <int>             <int>         <int>
1     0     0           2             2                 2           2
# i 2 more variables: sex <int>, year <int>

```