Some ideas for exploring missing data

Nicholas Tierney

Infectious Disease Ecology and Modelling group

Telethon Kids Institute, Perth, WA, Australia

library(tidyverse)  
library(visdat)  
library(naniar)

# Introduction

When you do data analysis, you come across missing data. Because I felt so frustrated by how hard it was to handle and wrangle missing data, I wanted to make it easier. In my endeavours I have written two R packages, visdat and naniar, for exploring missing data, and several papers on the topic.

The goal of this article is to share some condensed ideas on exploring missing data, using naniar, and visdat. To that end, we will focus on four questions.

1. How do we start looking at missing data?
2. How do we explore missingness in variables?
3. How do we explore missingness relationships?
4. How do we explore imputed values?

But first, let’s introduce ourselves to the data.

# The data

The data used for this paper is measurements of rodents in Kansas, from Hope (2023). The use of this data is inspired by Allison Horst’s “Exploring missing values in naniar” [shiny application](https://allisonhorst.shinyapps.io/missingexplorer/). In this paper we use a different, larger set of the data. For information on the metadata of the paper see [here](http://lter.konza.ksu.edu/content/csm08-small-mammal-host-parasite-sampling-data-16-linear-trapping-transects-located-8-lter). The data set provides various biometric length and weight measurements, for four species of rodents: the Eastern woodrat, Prairie vole, Western harvest mouse, and Deer mouse. Table [Table 1](#tbl-rodents-slice) shows a snapshot of 6 selected rows of the data, which shows some of the missingness.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Table 1: The rodents dataset, containing measurements of various rodents, including: total length - from tail to nose, tail length, hind foot length, ear length, and weight. Each row represents a measurement of a given species of roden at a particular date. There are missing values.   | species | total\_length | tail\_length | hind\_foot\_length | ear\_length | weight | sex | age | | --- | --- | --- | --- | --- | --- | --- | --- | | prairie vole | NA | 32 | 20 | NA | 30.0 | F | A | | deer mouse | NA | 62 | 19 | NA | 17.5 | F | A | | western harvest mouse | NA | NA | NA | NA | NA | N | NA | | western harvest mouse | NA | NA | NA | NA | NA | N | NA | | prairie vole | NA | NA | NA | NA | NA | N | NA | | prairie vole | NA | 26 | 18 | NA | 18.0 | M | Juv | |

# How do we start looking at missing data?

To get an overview of the missing data, we can use the visdat package (Tierney 2017), which was inspired by the work in [csv-fingerprint](https://setosa.io/blog/2014/08/03/csv-fingerprints/), and functions like missmap, from Amelia (Honaker, King, and Blackwell 2011). The key function for exploring missingness is vis\_miss(), which visualises the missingness of a whole dataframe. [Figure 1](#fig-vis-miss) gives an example where it displays the data as missing, or not missing, and provides information on the amount of missings in each column.

vis\_miss(rodents)

|  |
| --- |
| Figure 1: An overview of missing data in the rodents dataset. The x axis shows the variables of the data, along with the amount of missingness in that variable, and the y axis shows the rows. Each cell represents the missingness of a datum. The overall missingness is given in a percentage below in the legend. We learn that there is nearly 29% missing data overall, the missing data occurs in total\_length, tail\_length, hind\_foot\_length, ear\_length, weight, and age. |

We learn there is nearly 29% missing data overall, the missing data occurs in total\_length, tail\_length, hind\_foot\_length, ear\_length, weight, and age, and mostly in total\_length and ear\_length.

## Exploring subgroups: Using facetting in visdat

To see this plot split up by species, we can split up the vis\_miss plots into several facetted plots via the facet argument. For example, in [Figure 2](#fig-vis-miss-facet) we facet by the species variable. Visually, it appears that the missingness occurs in each species at roughly the same rate.

vis\_miss(rodents, facet = species)

|  |
| --- |
| Figure 2: An further overview of missing data in the rodents dataset. Similar to the previous graphic, we now present a facetted series of sub plots, one for each species. We learn from this that the missingness seems to be the same across each species |

There are other functions in the visdat package that focus on other types of data, for example, vis\_value(), vis\_binary(), and vis\_compare(). To read more about the functions available in visdat see the vignette [“Using visdat”](https://CRAN.R-project.org/package=visdat/vignettes/using_visdat.html).

# How do we explore missingness in variables?

The visdat package provides overviews of data, whereas naniar provides a more comprehensive set of tools for missing data.

## Numerical summaries of missing values

Two convenient counters of complete values and missings are n\_miss() and n\_complete(). These work on both data frames and vectors, similar to dplyr::n\_distinct()

dplyr::n\_distinct(rodents)

[1] 617

dplyr::n\_distinct(rodents$tail\_length)

[1] 96

n\_miss(rodents)

[1] 2013

n\_miss(rodents$tail\_length)

[1] 245

n\_complete(rodents)

[1] 4944

n\_complete(rodents$tail\_length)

[1] 528

The functions prop\_miss\_case() and pct\_miss\_case() return numeric value describing the proportion or percent of missing values in the dataframe.

prop\_miss\_case(rodents)

[1] 0.7477361

pct\_miss\_case(rodents)

[1] 74.77361

Similar to pct\_miss\_case(), prop\_miss\_case(), pct\_miss\_var() and prop\_miss\_var() returns the percent and proportion of variables that contain a missing value.

prop\_miss\_var(rodents)

[1] 0.6666667

pct\_miss\_var(rodents)

[1] 66.66667

The syntax for the other numerical summaries in naniar are miss\_, and then case, or var to refer to cases or variables. There are then summary, table suffixes. For example, the miss\_case\_summary() function returns a numeric value that describes the number of missings in a given case (aka row), the percent of missings in that row.

miss\_case\_summary(rodents) |> head()

# A tibble: 6 × 3  
 case n\_miss pct\_miss  
 <int> <int> <dbl>  
1 63 6 66.7  
2 74 6 66.7  
3 77 6 66.7  
4 78 6 66.7  
5 79 6 66.7  
6 91 6 66.7

The miss\_case\_table() function tabulates the number of missing values in a case / row. Below, this shows the number of missings in a case:

miss\_case\_table(rodents)

# A tibble: 7 × 3  
 n\_miss\_in\_case n\_cases pct\_cases  
 <int> <int> <dbl>  
1 0 195 25.2   
2 1 48 6.21  
3 2 277 35.8   
4 3 10 1.29  
5 4 14 1.81  
6 5 49 6.34  
7 6 180 23.3

We can interpret this output as follows:

* 195 cases with 0 missings, which comprises about 25% of the data,
* 48 cases with 1 missing, these make up 6% of the data,
* 277 cases with 2 missing, these make up 35% of the data,
* and so on.

miss\_var\_summary() returns the number of missing values in a variable, and the percent missing in that variable.

miss\_var\_summary(rodents)

# A tibble: 9 × 3  
 variable n\_miss pct\_miss  
 <chr> <int> <dbl>  
1 total\_length 569 73.6  
2 ear\_length 530 68.6  
3 tail\_length 245 31.7  
4 hind\_foot\_length 243 31.4  
5 weight 237 30.7  
6 age 189 24.5  
7 date 0 0   
8 species 0 0   
9 sex 0 0

Finally, the miss\_var\_table() function describes the number of missings in a variable:

miss\_var\_table(rodents)

# A tibble: 7 × 3  
 n\_miss\_in\_var n\_vars pct\_vars  
 <int> <int> <dbl>  
1 0 3 33.3  
2 189 1 11.1  
3 237 1 11.1  
4 243 1 11.1  
5 245 1 11.1  
6 530 1 11.1  
7 569 1 11.1

We can interpret this as there being:

* 3 variables with 0 missings, comprising 33% of variables in the dataset, and
* in the remaining variables similar patterns of missings, but not the exact same number of missing values.

## Visualise missingness in variables

To specifically focus on the number or proportion of missings in each variable, we can use gg\_miss\_var(), as seen in [Figure 3](#fig-gg-miss-var).

gg\_miss\_var(rodents)

|  |
| --- |
| Figure 3: Number of missing values for each variable. The x axis shows the number of missings, and the y axis shows each variable. We learn total length, and ear length have the most missing values, followed by tail length, hind foot length, weight, and age. |

This displays the number of missing values in each variable. We learn similar information to [Figure 1](#fig-vis-miss): total length, and ear length have the most missing values, followed by tail length, hind foot length, weight, and age. Just like with vis\_miss(), we can add in facets in these plots, via the facet argument, see [Figure 4](#fig-gg-miss-var-facet). We learn again, that the species have similar amounts of missing data in their variables. Sometimes it is useful to confirms the same piece of information!

gg\_miss\_var(rodents, facet = species, show\_pct = TRUE)

|  |
| --- |
| Figure 4: Similar to the above plot but one subplot for each species. We learn that the missingness pattern is pretty similar for each species. We normalise using show\_pct = TRUE as there are different numbers of observations in each species. |

It feels like there are several patterns with the missingness - some variables tend to go missing at the same time. To explore these patterns we can use gg\_miss\_upset(), which produces an “upset” plot of the intersecting sets of missingness (Conway, Lex, and Gehlenborg 2017). This can be thought of as a generalised way to visualise intersecting Venn diagrams.

gg\_miss\_upset(rodents)

|  |
| --- |
| Figure 5: An upset plot of 7 sets of missingness in the rodents data displaying a more nuanced depiction of the patterns of missingness in the data. The number of missings in each variable is seen on the bottom left bar plot. The bottom panel shows the different sets of co-occuring missingness. For example, the bottom left two dots show ear length and total length going missing together - corresponding to the bar plot above it, showing 278 missings. We learn that there are two dominant sets of missingness, where ear length and total length go missing, and then weight hind foot length, tail length, ear length and total length being missing. |

There are more visualisations available in naniar (each starting with gg\_miss\_) - you can see these in the [“Gallery of Missing Data Visualisations” vignette](https://cran.r-project.org/package=naniar/vignettes/naniar-visualisation.html). Most plots created with the gg\_miss family all have a basic theme (except for gg\_miss\_upset()), but you can customise them by adding components like a standard ggplot object:

gg\_miss\_var(rodents) +   
 theme\_bw() +   
 labs(y = "Number of missing observations")

It is also worth noting that for every visualisation of missing data in naniar, there is an accompanying function to extract the data used in the plot. This is important as the plot should not return a dataframe - but we want to make the data available for use by the user so it isn’t locked into a plot.

For example, miss\_var\_summary() provides the dataframe that gg\_miss\_var() is based on.

miss\_var\_summary(rodents)

# A tibble: 9 × 3  
 variable n\_miss pct\_miss  
 <chr> <int> <dbl>  
1 total\_length 569 73.6  
2 ear\_length 530 68.6  
3 tail\_length 245 31.7  
4 hind\_foot\_length 243 31.4  
5 weight 237 30.7  
6 age 189 24.5  
7 date 0 0   
8 species 0 0   
9 sex 0 0

Which also works with group\_by():

rodents %>%   
 group\_by(species) %>%   
 miss\_var\_summary()

Similarly, there is a data\_vis\_miss() function in the visdat package, which returns the data in the format that this visualisation requires.

data\_vis\_miss(rodents)

The aim of these is to provide the data required to make these visualisations, so if people want to create their own more customised versions of vis\_miss() or gg\_miss\_var() then they can do that.

# How to explore missingness relationships?

We can identify key missing variables using vis\_miss(), gg\_miss\_var(), and gg\_miss\_upset(), but for further exploration, we need to explore the relationship amongst the variables in this data:

* date
* species
* total\_length
* tail\_length
* hind\_foot\_length
* ear\_length
* weight
* sex
* age

## Exploring using bivariate plots

Let’s say that we want to explore the relationship between tail length and ear length. [Figure 6](#fig-example-geom-point) shows a scatter plot of tail length and ear length.

library(ggplot2)  
ggplot(rodents,   
 aes(x = ear\_length,   
 y = tail\_length)) +   
 geom\_point()

|  |
| --- |
| Figure 6: Plot of ear length against tail length. Ear length is on the X axis and tail length is on the Y axis. We learn that there is a reasonable positive correlation of tail length and ear length. |

The problem with this is ggplot removes the missing values. This makes them hard to explore. We can impute missings with values 10% lower than the minimum value in that variable, which puts these values in a margin area on the graphic. This method comes from [ggobi](https://en.wikipedia.org/wiki/GGobi) (Cook and Swayne 2007), and [manet](http://www.rosuda.org/MANET/) (Unwin et al. 1996).

This imputation is wrapped up in the geom\_miss\_point() ggplot2 geom. [Figure 7](#fig-geom-miss-point) illustrates this by exploring the relationship between tail length and ear length from the rodents dataset.

ggplot(rodents,   
 aes(x = ear\_length,   
 y = tail\_length)) +   
 geom\_miss\_point() +   
 scale\_colour\_brewer(palette = "Dark2")

|  |
| --- |
| Figure 7: Improved plot of tail length against ear length, we can now see the missing values are imputed 10% below the minimum value. The green dots on the Y axis represent tail\_length values that have missing ear\_length. There aren’t any missing values on the X axis, because there aren’t times where tail length is missing when ear length is missing. The row of dots in the bottom left corner are missing for both tail length and ear length |

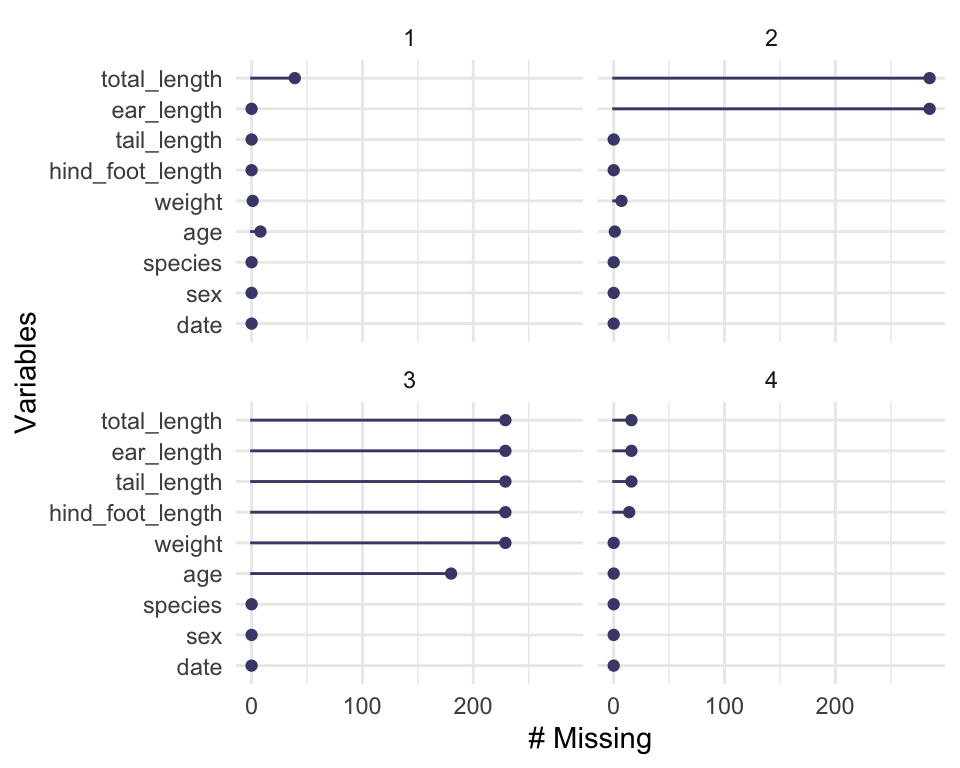
Being a proper ggplot geom, it supports all of the standard features of ggplot2, such as **facets** and **themes** as shown in the code below (output suppressed for brevity).

ggplot(rodents,   
 aes(x = ear\_length,   
 y = tail\_length)) +   
 geom\_miss\_point() +   
 facet\_wrap(~species) +   
 theme\_dark()

## Exploring using modelling

As evidenced by [Figure 5](#fig-gg-miss-upset), there is a structure in the missingness in the rodents data. We can perform some basic clustering on the missingness and then and learn which variables and their values predict these missingness groups using decision trees (Tierney et al. 2015; Barnett et al. 2017). We start by adding missingness clusters, choosing four based on [Figure 5](#fig-gg-miss-upset). We encourage exploring different numbers of clusters. We can then confirm this pattern using visualisations.

rodents\_miss\_clust <- rodents %>% add\_miss\_cluster(n\_clusters = 4)  
gg\_miss\_var(rodents\_miss\_clust, facet = miss\_cluster)



Number of missings for each variable for each cluster. We see clear patterns emerge where there are two variables missing in cluster two, six variables missing in cluster three, and not as many missings in clusters one and four.

We use the R package rpart (Therneau and Atkinson 2023) to fit a classification and regression tree (CART) to the data using all variables.

library(rpart)  
rodent\_miss\_cart <- rpart(  
 factor(miss\_cluster) ~ .,   
 data = rodents\_miss\_clust  
 )

Variable importance scores ([Table 2](#tbl-var-imp)) reveal the most important variables for predicting missingness cluster are date and sex.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Table 2: Variable importance scores for predicting missingness cluster. The most important variables are date and sex.   | variable | importance | | --- | --- | | date | 221.39 | | sex | 167.96 | | hind\_foot\_length | 20.28 | | age | 15.08 | | species | 10.67 | | tail\_length | 7.37 | | weight | 5.18 | |

To better understand these relationships, we would recommend exploring using partial dependence plots using packages such as vip (Greenwell and Boehmke 2020) and other decision tree plots using rpart.plot (Milborrow 2022).

# How do we explore imputed values?

The [simputation](https://cran.r-project.org/package=simputation) package provides a nice interface to imputation. We will impute values for tail\_length using the impute\_lm() function, then visualise the data, as seen in [Figure 8](#fig-simpute-invisible).

library(simputation)  
library(dplyr)  
  
rodents %>%  
 impute\_lm(tail\_length ~ species + age) %>%  
 ggplot(aes(x = weight,  
 y = tail\_length)) +   
 geom\_point()

Warning: Removed 237 rows containing missing values (`geom\_point()`).

|  |
| --- |
| Figure 8: Imputed values are not visible. A plot of tail length by weight. The Imputed tail length values are not visible because we have no way to identify them in the data. |

We don’t get any warnings regarding missing observations - because they are all imputed! However this comes at a cost: we don’t know where the imputations are - they are now sort of invisible.

We can track a copy of the missing data locations by using the function nabular(), which binds another dataset to the current one which notes the locations of the missing data. “Nabular” data is a really important idea in naniar, but to keep it brief, for each column with missing values, a new column is created to help identify misingness. For example, a new column called ear\_length\_NA is created:

nabular(rodents) |>   
 select(starts\_with("ear\_length")) |>   
 head()

# A tibble: 6 × 2  
 ear\_length ear\_length\_NA  
 <dbl> <fct>   
1 39 !NA   
2 18 !NA   
3 17 !NA   
4 21 !NA   
5 19 !NA   
6 19 !NA

The key takeaway here is there is now a copy of the data bound to it, with each column ending in \_NA, and the values either being “NA” for missing, or “!NA” for not missing. For more details on the ideas underlying this, and the benefits, we recommend reading our paper, “Expanding Tidy Data Principles to Facilitate Missing Data Exploration, Visualization and Assessment of Imputations” (Tierney and Cook 2023).

Using the shadow matrix to keep track of where the missings are, you can actually keep track of the imputations, colouring by what was previously missing in tail\_length. For example, let’s create the nabular data, then impute the data using a random forst, and plot it in [Figure 9](#fig-simpute-visible-lm).

rodents\_nabular <- rodents %>%  
 nabular() %>%   
 as.data.frame()  
  
rodents\_lm\_tail\_imputed <- rodents\_nabular %>%   
 impute\_lm(tail\_length ~ species + date) %>%   
 impute\_lm(weight ~ tail\_length + species + date)  
  
ggplot(rodents\_lm\_tail\_imputed,  
 aes(x = weight,  
 y = tail\_length,  
 colour = tail\_length\_NA)) +   
 geom\_point(alpha = 0.5) +   
 scale\_colour\_brewer(palette = "Dark2")

|  |
| --- |
| Figure 9: Linear model imputed values of tail length in a scatterplot of tail length vs weight. Weight is on the X axis and tail\_length is on the Y axis, and the points are coloured by whether they are imputed - ‘NA’ indicates a previously missing value that has been imputed. We can see where imputations are added, but they are very concentrated |

The simputation package has a nice option to add residual noise to the imputations - in this case we can add some normal noise to the observations, where the residuals are draws with replacement from the model residuals. This gives us much greater variation in the imputations. For comparison to other naive approaches, we will also add mean imputation for comparison

Importantly, we can actually compare the two methods as below. This first imputes the data using the residual method, then rowbinds the two datasets together, creating a column called “imputation\_type”, which records which type of imputation was used, either “add\_residual” or “no\_residual”:

rodents\_lm\_tail\_imputed\_res <- rodents\_nabular %>%   
 impute\_lm(tail\_length ~ species + date, add\_residual = "observed") %>%   
 impute\_lm(weight ~ tail\_length + species + date, add\_residual = "observed")  
  
rodents\_mean\_imputed <- rodents %>%  
 nabular() %>%   
 as.data.frame() %>%   
 impute\_mean\_all()  
  
rodents\_imputed\_comparison <- bind\_rows(  
 add\_residual = rodents\_lm\_tail\_imputed\_res,  
 no\_residual = rodents\_lm\_tail\_imputed,  
 mean\_imp = rodents\_mean\_imputed,  
 .id = "imputation\_type"  
) %>% as\_tibble()

We can see see the two different imputation methods side by side in [Figure 10](#fig-imputed-comparison).

ggplot(rodents\_imputed\_comparison,  
 aes(x = weight,  
 y = tail\_length,  
 colour = tail\_length\_NA)) +   
 geom\_point(alpha = 0.5) +   
 scale\_colour\_brewer(palette = "Dark2") +   
 facet\_wrap(~imputation\_type)

|  |
| --- |
| Figure 10: Comparing imputation methods of tail length in a scatterplot of tail length vs weight. Weight is on the X axis and tail\_length is on the Y axis, and the points are coloured by whether they are imputed - ‘NA’ indicates a previously missing value that has been imputed. We learn that mean imputation provides imputations that are not representative, linear models with no residuals are representative of the data but very concentrated, and adding residuals adds much more variation to your data. |

# Conclusion

In this software corner we have demonstrated the use of the visdat and naniar R packages for exploring and understanding missing data.

# References

Barnett, Adrian G., Paul McElwee, Andrea Nathan, Nicola W. Burton, and Gavin Turrell. 2017. “Identifying Patterns of Item Missing Survey Data Using Latent Groups: An Observational Study.” *BMJ Open* 7 (10): e017284. <https://doi.org/10.1136/bmjopen-2017-017284>.

Conway, Jake R., Alexander Lex, and Nils Gehlenborg. 2017. “UpSetR: An R Package for the Visualization of Intersecting Sets and Their Properties.” *Bioinformatics (Oxford, England)* 33 (18): 2938–40. <https://doi.org/10.1093/bioinformatics/btx364>.

Cook, Dianne, and Deborah F. Swayne. 2007. *Interactive and Dynamic Graphics for Data Analysis: With and* . Use R! New York: Springer. <https://doi.org/10.1007/978-0-387-71762-3>.

Greenwell, Brandon M., and Bradley C. Boehmke. 2020. “Variable Importance Plots—an Introduction to the Vip Package.” *The R Journal* 12 (1): 343–66. <https://doi.org/10.32614/RJ-2020-013>.

Honaker, James, Gary King, and Matthew Blackwell. 2011. “Amelia II: A Program for Missing Data.” *Journal of Statistical Software* 45 (1): 1–47. <https://doi.org/10.18637/jss.v045.i07>.

Hope, Andrew. 2023. “CSM08 Small Mammal Host-Parasite Sampling Data for 16 Linear Trapping Transects Located in 8 LTER Burn Treatment Watersheds at Konza Prairie.” Environmental Data Initiative. <https://doi.org/10.6073/PASTA/F7BFB7226B093C817DC391C34A88B514>.

Milborrow, Stephen. 2022. *Rpart.plot: Plot ’Rpart’ Models: An Enhanced Version of ’Plot.rpart’*. <https://CRAN.R-project.org/package=rpart.plot>.

Therneau, Terry, and Beth Atkinson. 2023. *Rpart: Recursive Partitioning and Regression Trees*. <https://CRAN.R-project.org/package=rpart>.

Tierney, Nicholas. 2017. “visdat: Visualising Whole Data Frames.” *The Journal of Open Source Software* 2 (16): 355. <https://doi.org/10.21105/joss.00355>.

Tierney, Nicholas, and Dianne Cook. 2023. “Expanding Tidy Data Principles to Facilitate Missing Data Exploration, Visualization and Assessment of Imputations.” *Journal of Statistical Software* 105 (7): 1–31. <https://doi.org/10.18637/jss.v105.i07>.

Tierney, Nicholas, Fiona A. Harden, Maurice J. Harden, and Kerrie L. Mengersen. 2015. “Using Decision Trees to Understand Structure in Missing Data.” *BMJ Open* 5 (6): e007450. <https://doi.org/10.1136/bmjopen-2014-007450>.

Unwin, Antony, George Hawkins, Heike Hofmann, and Bernd Siegl. 1996. “Interactive Graphics for Data Sets with Missing Values—MANET.” *Journal of Computational and Graphical Statistics* 5 (2): 113–22. <https://doi.org/10.1080/10618600.1996.10474700>.