What is missing data?

Some definitions are based on representation:

Missing data is the lack of a recorded answer for a particular field.

Other definitions are based on context:

Missing data is lack of a recorded answer where we “expected” to find one

WHY DO WE CARE ABOUT MISSINGNESS?

Missing data can result in:

- Reduced statistical power
- Biased estimators
- Reduced representativeness of the sample
- Generally incorrect inference and conclusions

THERE IS NO ONE CLEAR ANSWER FOR HANDLING MISSINGNESS

- Throw out the missing data and make a note of it.
- Throw out missing data or fill it in and make an informative note of it.

Disguised Missingness

Don’t assume it will be in native form

- Blanks
- Empty stings
- NA
- NULL
- Anything else that well intentioned humans may come up with
- -999999
- “Did not answer”
- “Ugh, sensor was broken”

How is missingness represented in your dataset?

- Mixing of missing indicators, e.g. both NA and NULL in the same variable, may indicate different interpretations.
- Don’t hesitate to reach out to the client or other subject matter experts .
- Null data can be visiualized in R
- naniar and Amelia packages can produce “missingness maps”

Mechanisms of Missingness

- Missing Completely at Random (MCAR)

The data are equally likely to be missing.

- 2. Missing at Random (MAR)

The likelihood of being missing depends only on non-missing data.

- 3. Missing Not at Random (MNAR)

Missingness depends on unobserved data or the value of the missing data itself.

Imputing missing values

- R has robust packages for missing value imputations.
- These packages arrive with some inbuilt functions and a simple syntax to impute missing data at once.
- Some packages are known best working with continuous variables and others for categorical..

Injecting missing value

- set.seed(86)
- iris[sample(1:nrow(iris), 5), "Sepal.Width"] <- NA
- iris[sample(1:nrow(iris), 10), "Petal.Length"] <- NA
- iris[sample(1:nrow(iris), 8), "Sepal.Length"] <- NA

How to identify missing values

#Using is.na() fucntion

any(is.na(iris))

#complete.cases() function to get percentage of missing value

nrow(iris[!complete.cases(iris),])/nrow(iris)*100

Next is to identify which variables and what percentage of observations from each variable are missing.

use md.pattern function from mice package in R.

library(mice)

md.pattern(iris)

Deleting missing observations

- When total number of missing observatins is significant then we can think of removing those observations from the data.
- Imputing too many missing observations can lead to bias in the dataset.
- Also it can result into poor statistical models.
- We can delete the missing values at the data preparation stage or at the time of building the model.
- However, not all algorithms provide this option of deleting missing values while we train the model.

- iris <- iris[complete.cases(iris),]
- # or we can use na.omit() function
- iris <- na.omit(iris)

- For linear regression mtcars ignoring missing values while building lm model
- lm(mpg ~ cyl + disp, data=mtcars, na.action=na.omit)

How to delete variables with missing values

- Sometimes one or two variables contribute to the most number of missing values.
- In such cases, deleting these variables with high percentage of missing values will help save lots of observations.
- According to we delete all variable which have more than 30% of missone thumb rule ing values.

- ## Removing columns with more than 30% NA
- iris[, -which(colMeans(is.na(iris)) > 0.3)]

Imputing Missing values

#Imputing missing values

- Replacing missing values with a rough approximate value is acceptable and could possibly result into satisfactory result. Let us look at some of the ways in which we can replace the missing values.

#Using mean/median/mode

- To replace missing values with mean, median or mode we can use impute function from Hmisc package. This can also be achieved by using square brackets[] or ifelse statement.

Other ways to replace missing values

- Using impute function from Hmisc package
- library(Hmisc)
- impute(iris$Sepal.Length, mean) # replace with mean
- impute(iris$Sepal.Length, median) # median

#Filling missing values with Mean

- iris$Sepal.Length[is.na(iris$Sepal.Length)] = mean(iris$Sepal.Length, na.rm=TRUE)
- # alternative way is to use ifelse
- iris = transform(iris, y = ifelse(is.na(iris), mean(iris, na.rm=TRUE), Sepal.Length))

Multivariate Imputation By Chained Equations

- #The mice function from the package automatically detects the variables which have missing values.
- Once detected, the missing values are then replaced by Predictive Mean Matching (PMM), this is a default method.
- library(mice)
- # Imputing the values using mice
- imputed_iris <- mice(iris, m=5, method = 'pmm', seed = 101)
- # checking the summary
- summary(imputed_iris)

#Checking the imputed values

- imputed_iris$imp

Using Machine Learning Algorithms

- Using KNN to fill the missing values
- library(bnstruct)
- knn.impute(iris, k = 5, cat.var = 1:ncol(iris), to.impute = 1:nrow(iris),
  using = 1:nrow(iris))
- Using RandomForest to fill the missing values
- Set.seed(86)
- iris <- rfImpute(Species ~ ., iris.na)

Which is the best imputation method?

- Although there are many ways in which we can impute missing values, one cannot say with certainty, that a particular method provides a best result.
- Therefore, it is advised to test out some of these methods and see which one is providing the best result.
- As we know statistics is all about trial and error.