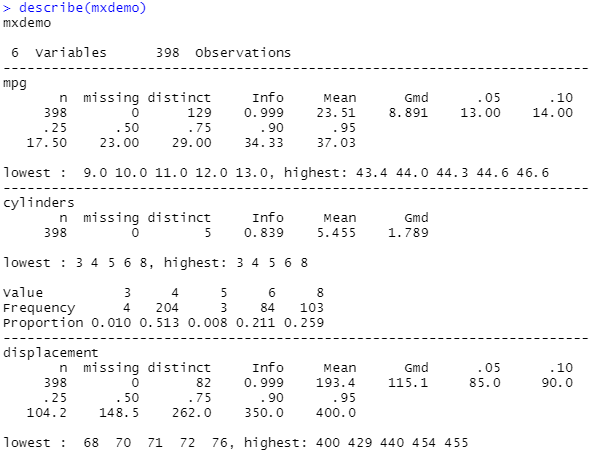
**Data Wrangling with R MAX522/F22 version 1.0** Outlier item (e) highlighted in yellow was missing in prior version

The data preparation tasks done with IBM SPSS Modeler can be done with R. Prior to performing any data preparation tasks as per Modeler, an Understanding of the data set must be completed to identify the profile of the variables, i.e., continuous variables: mean, median, standard deviation, skewness, and completeness. These R functions were addressed in prior MAX courses and the R Workshop provided to everyone.

1. Discover issue with data integrity using the “automregr.csv” file

#After installing/activating the HMISC package from CRAN, use the DESCRIBE function for more extensive information—only displayed some of the variables:

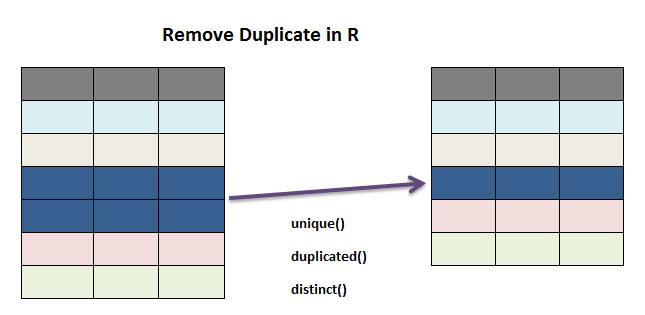
1.   
   #Info—larger values indicate fewer tied data values

**2.Remove Duplicate rows in R using Dplyr – distinct () function**

Distinct function in R is used to remove duplicate rows in R using Dplyr package.  Dplyr package in R is provided with distinct() function  which eliminate duplicates rows with single variable or with multiple variable. There are other methods to drop duplicate rows in R one method is duplicated() which identifies and removes duplicate in R. The other method is unique() which identifies the unique values.

we will looking at example on How to

* Get distinct Rows of the dataframe in R using distinct() function.
* Drop duplicates of the dataframe using duplicated() function in R
* Get unique rows (remove duplicate rows) of the dataframe in R using unique() function.



**Create Dataframe**

We will be using the following dataframe  to depict the above functions. Lets first create the dataframe.

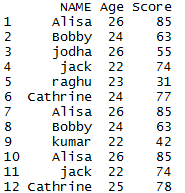
|  |  |  |
| --- | --- | --- |
| 1 | # simple Data frame creation | |
| 2 |  |

|  |  |  |
| --- | --- | --- |
| 3 | mydata = data.frame (NAME =c ('Alisa','Bobby','jodha','jack','raghu','Cathrine', | |
| 4 | 'Alisa','Bobby','kumar','Alisa','jack','Cathrine'), |

|  |  |
| --- | --- |
| 5 | Age = c (26,24,26,22,23,24,26,24,22,26,22,25), |
| 6 | Score =c(85,63,55,74,31,77,85,63,42,85,74,78)) |

|  |  |
| --- | --- |
| 7 |  |
| 8 | mydata | |

so the resultant data frame will be



**distinct() Function in Dplyr  –  Remove duplicate rows of a dataframe in R:**

*[mb, you need tidyverse package installed first]*

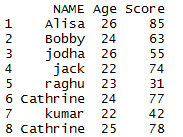
|  |  |  |
| --- | --- | --- |
| 1 | library(dplyr) | |
| 2 |  |

|  |  |  |
| --- | --- | --- |
| 3 | # Remove duplicate rows of the dataframe | |
| 4 | distinct(mydata) |

**OR**

|  |  |
| --- | --- |
| 1 | library(dplyr) |
| 2 | mydata %>% distinct() | |

In this dataset, all the duplicate rows are eliminated so it returns the unique rows in mydata.



<https://www.youtube.com/watch?reload=9&v=4DtxkvQegMM>   
one brief tutorial removing duplicated rows

1. **Addressing Missing character & numeric data values coded as NA in R**
2. To discover missing values for a variable, use is.na(variable name) function to display a vector of TRUE (missing) and FALSE (not missing entries.
3. To recode a numeric missing value code, e.g. 99 in Modeler use:  
   filename$variablename[filename$variablename ==99] <- NA
4. Some functions include an optional parameter to exclude missing values prior to execution:  
   na.rm=TRUE
5. Complete rows with 1 or more missing values can be deleted using:   
   newdatafilename<-na.omit(existingdatafilename)

IMPUTE a value to replace NA.   
Replace NA with mean where col=variable name per example below

**df$col[is.na(df$col)] <- mean(df$col, na.rm=TRUE)**

**df$var1[is.na(df$var1)] <- mean(df$var1, na.rm=TRUE)**

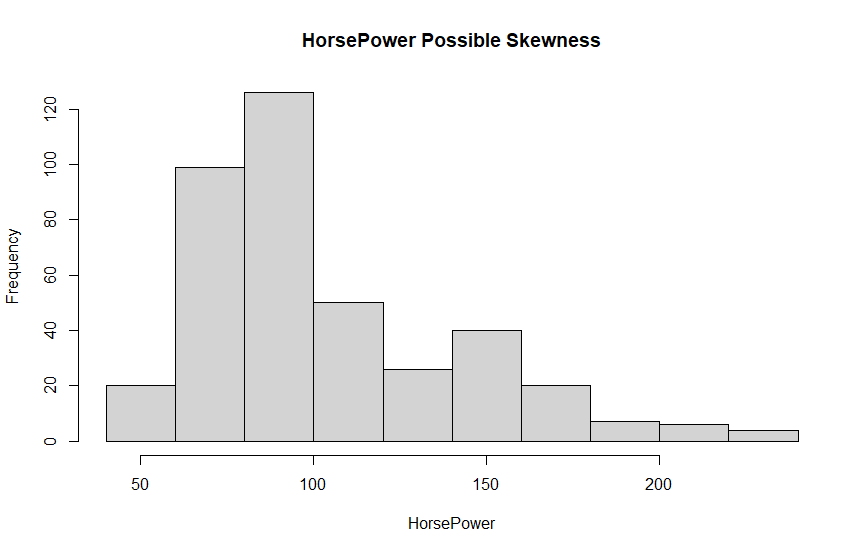
Replace NA with median where col=variable name per example below

**replace missing values in first column with median of first column**

**df$var1[is.na(df$var1)] <- median(df$var1, na.rm=TRUE)**

IF you wanted to use the MODE, replace mean or median terms in the examples above with the calculated mode as illustrated below:

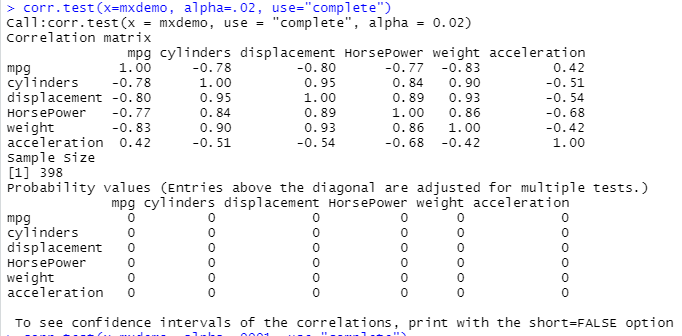
To find the MODE of a variable, first install the modeest package from CRAN. Activate it. The Mode function is mfv(variable name) where mfv = most frequent variable, e.g., for HorsePower:  
  
Of course assign a variable name for the mode calculation to be inserted in either the mean or median functions above.

1. **Skewness Detection**:   
   #install Moments package from CRAN—find skewness of a single variable  
     
     
   #To copy the plot below: select in lower right window: EXPORT, pull down options and choose Copy to Clipboard. When image displays, select Copy Plot. Then PASTE into WORD doc.  
   BTW graph options include: col=”color”, breaks= integer (is#bins)  
     
   What type of skewness, if any, is displayed—recall value was 1.08862? Should a data transform be applied to this variable?

Skewness function options:

SQRT(x) if +skew; SQRT(max(x+1)-x) of – skew  
 log10(x) for + skew; log10(max(x+1)-x) if – skew  
 ln(x) or ln(+1) for natural log

5. If you wanted to investigate multicollinearity, install the PSYCH package/activate it and run this test that displays the correlation matrix above and the statistical probability values below



How would you fix multicollinearity?

1. Outliers can be  
   a. deleted by recoding values above e.g., 3\*calculated standard deviation as NA  
   install data.table package first  
   outlierReplace(my\_data, c("num\_students\_total\_gender.num\_students\_female", "num\_students\_total\_gender.num\_students\_total"), which(my\_data$num\_students\_total\_gender.num\_students\_female > 1000), NA)

b. scissored or capped with X\*SD calculated value using:  
 instead of of changing outliers to NA, we could make them equal to a maximal number. To do this, and show you a clear results, we'll take all observations with more than 500 female students, and cap them at 500.

outlierReplace(my\_data, "num\_students\_total\_gender.num\_students\_female", which(my\_data$num\_students\_total\_gender.num\_students\_female >

500), 500)

More on Capping outlier values:

**Z-score method:**

The following code shows how to calculate the z-score of each value in each column in the data frame, then remove rows that have at least one z-score with an absolute value greater than 3:

**#find absolute value of z-score for each value in each column**

**z\_scores <- as.data.frame(sapply(df, function(df) (abs(df-mean(df))/sd(df))))**

**#view first six rows of z\_scores data frame**

**head(z\_scores)**

**A B C**

**1 1.2813403 0.25350805 0.39419878**

**2 0.3110243 1.80496734 0.05890232**

**3 1.3483190 0.12766847 0.08112630**

**4 1.2908343 1.32044506 0.38824414**

**5 0.4313316 1.40102642 0.44450451**

**6 1.5271674 0.04327186 0.70295309**

**#only keep rows in dataframe with all z-scores less than absolute value of 3**

**no\_outliers <- z\_scores[!rowSums(z\_scores>3), ]**

**#view row and column count of new data frame**

**dim(no\_outliers)**

**[1] 994 3**

Removed 6 rows in above example  
PERHAPS **an easier approach** to cap or scissor data values for a variable:

1. Compute the mean for the variable m1<-mean(variable)
2. Compute the standard deviation for a variable m2<-sd(variable)
3. Create the upper limit value for scissoring: UL<-m1+3\*m2
4. Determine the number of data values for the variable to use as the number of iterations for a Loop, e.g., quant1<-length(bop.df$LOCAL)
5. Create a loop that checks IF the variable’s data value >UL, replace with UL, e.g.,  
   for (i in 1:quant1) if (bop.df$LOCAL[i]>4) bop.df$LOCAL[i]<-4
6. **Balancing subgroups within a variable**

To REDUCE the number of data items for a variable subgroup identified as:  
SG1, SG2, SG3, the number of items for each can be determined using  
len1<-length(filename$variablename)  
Pause RStudio and manually decide the number of data values that you want for each subgroup.  
Assume you only want to reduce one subgroup, use the SAMPLE function:   
**sample(x, size, replace = FALSE, prob = NULL)**

where:

* **x:** A vector of elements from which to choose.
* **size:**Sample size.
* **replace:**Whether to sample with replacement or not. Default is FALSE.
* **prob:**Vector of probability weights for obtaining elements from vector. Default is NULL.

For example:

Filename$variablename<-sample(x=filename$variablename, size=value)  
where value is the reduced number of data items for this variable

Another Approach:

Balancing subgroups

There are (at least) three ways to do subgroup analyses in R.

1. First (and I think easiest), we can use a **'select**' statement to restrict an analysis to a subgroup of subjects.
2. Second, the tapply() function can be used to perform analyses across a set of subgroups in a dataframe.
3. Third, we can create a new data frame for a particular subgroup using the subset() function, and then perform analyses on this new data frame.

An analysis can be restricted to a subset of subjects using the 'varname[subset]' format. For example,

> mean(agewalk[group==1])

[1] 10.72727

finds the mean of the variable 'agewalk' for those subjects with group equal to 1. When specifying the condition for inclusion in the subset analysis ('Group==1' in this example), two equal signs '==' are needed to indicate a value for inclusion. Less than (<) and greater than (>) arguments can also be used. For example, the following command would find the mean systolic blood pressure for subjects with age over 50:

> mean(sysbp[age>50])

Another approach is to use the tapply() function to perform an analysis on subsets of the data set. The input for the tapply( ) function is 1) the outcome variable (data vector) to be analyzed, 2) the categorical variable (data vector) that defines the subsets of subjects, and 3) the function to be applied to the outcome variable. To find the means, standard deviations, and n's for the two study groups in the 'kidswalk' data set:

> tapply(agewalk, group, mean)

|  |  |
| --- | --- |
| 1 | 2 |
| 10.72727 | 11.91176 |

> tapply(agewalk, group, sd)

|  |  |
| --- | --- |
| 1 | 2 |
| 1.231684 | 1.277636 |

> tapply(agewalk, group, length)

1 2

33 17

The subset() function creates a new data frame, restricting observations to those that meet some criteria. For example, the following creates a new data frame for kids in Group 2 of the kidswalk data frame (named 'group2kids'), and finds the n and mean Age\_walk for this subgroup:

> group2kids <- subset(kidswalk,Group==2)

> length(group2kids)

[1] 5

> mean(group2kids$Age\_walk)

[1] 11.91176

In this example, there are two data sets open in R (kidswalk for the overall sample and group2kids for the subsample) that use the same set of variables names. In this situation, it is helpful to use the 'dataframe$variablename' format to specify a variable name for the appropriate sample.

When specifying the condition for inclusion in the subsample ('Group==2' in this example), two equal signs '==' are needed to indicate a value for inclusion. Less than (<), less than or equal to (<=), greater than (>), greater than or equal to (>=), or not equal to (!=) arguments can also be used. For example,

> age65plus <- subset(allsubjects,age>64)

would create a dataframe of subjects aged 65 and older.

PARTITION Data set using Target variable—example below “train” is dataframe name and age is variable used to split dataset. To use “careateDataPartition” function, FIRST install caret package from CRAN.

Graphical user interface

Description automatically generated with low confidence

More efficient approach:

The following code splits 70% of the data selected randomly into training set and the remaining 30% sample into test data set.

data <-read.csv("c:/datafile.csv")

dt = sort(sample(nrow(data), nrow(data)\*.7))

train<-data[dt,]

test<-data[-dt,]

The minus sign is the last command line indicates select the remaining .3 of the dataset for testing.

**REFERENCES:**

1. R in Action: Data analysis and graphics with R, R Kabacoff, Manning, 2020.
2. Discovering Statistics using R, A Fields,et.al., Sage, 2012 (encyclopedia)
3. Machine Learning Using R: …, 2nd.ed., K Ramasubramanian & A Singh, Apress, 2019
4. Statistical Analysis with R(Dummies), J. Schmuller, Wiley, 2017
5. R Fundamental 2: Data Manipulation in R, 2nd ed., S.Locke, Independent publisher, 2017