**2. Subsetting in R**

* **Individual Elements Access**:
  + Access vectors, matrices, arrays, or data frames using [ ] by index or name.
  + Example:

r

Copy code

a <- data.frame(row.names = "mpg" %+% chr(1:nrow(mpg)),

manufacturer = mpg$manufacturer,

model = mpg$model,

displ = mpg$displ,

cyl = mpg$cyl)

head(a)

* + Output:

Copy code

manufacturer model displ cyl

mpg1 audi a4 1.8 4

mpg2 audi a4 1.8 4

mpg3 audi a4 2.0 4

mpg4 audi a4 2.0 4

mpg5 audi a4 2.8 6

mpg6 audi a4 2.8 6

* **Subsetting by Index, Row Names, and Column Names**:
  + Access specific elements:

r

Copy code

a[3,3]

a["mpg3", "displ"]

a["mpg3",]

* **Subsetting Rows**:
  + By a vector of indices:

r

Copy code

a[c(1:2),]

a[-c(2:nrow(mpg)), ]

* + By a logical vector:

r

Copy code

a[c(T,F,T),]

* **Subsetting Columns**:
  + Access specific columns:

r

Copy code

a$manufacturer

**3. Functions in R**

* **Defining Functions**:
  + Functions take data as input and process it into output.
  + Example:

add <- function(a,b) {

result <- a + b

return(result)

}

**4. Operators in R**

* **Assignment**:
  + <- for assignment.
  + = does not serve as an assignment operator; instead, it
  + is an operator that specifies a named parameter formula for the lm
  + function.
* **Arithmetic**:
  + +, -, \*, /, ^, %%, %/%, %\*%
* **Logical**:
  + &, |, !
* **Comparison**:
  + <, >, <=, >=, ==, !=
* **Set Operations**:
  + %in% for set membership.
  + Example:

A <- letters[1:3]

B <- letters[1:5]

C <- letters[2:6]

print(A %in% B)

print(A %in% C)

print(all(A %in% B))

print(all(A %in% C))

**5. Frequently Used Functions**

* **Basic Stats**:
  + max, min, summary
* **Rounding**:
  + round, floor
* **Concatenate Vectors**:
  + c, cbind, rbind
* **Size**:
  + length, dim, nrow, ncol
* **Vector Sorting**:
  + sort, rank, order
* **Display or Concatenate into a String**:
  + print, cat, paste, format
* **Others**:
  + apply, table, which

**6. Examples**

* **Basic Example**:

a <- letters[1:5]

b <- table(a, sample(a))

b

apply(b, 1, mean)

**7. Branching in R**

* **If Statements**:

if (logical\_expression) {

statements

} else {

alternative\_statements

}

* **Branching Example**:

x <- -4

if (x >= 0) {

print(sqrt(x))

} else {

print(NA)

}

**8. More Branching**

* **Ifelse Function**:

r

Copy code

x <- c(4:-4)

sqrt(ifelse(x >= 0, x, NA))

**9. Looping in R**

* **For Loop**:

for (i in 1:5) {

print(i\*i)

}

* **While Loop**:

i <- 1

while (i <= 5) {

print(i\*i)

i <- i + sqrt(i)

}

* **Repeat Loop**:

i <- 1

repeat {

if (i > 5) break

print(i\*i)

i <- i + sqrt(i)

}

* **Next and Break in Loops**:

for (i in 1:10) {

if (i %% 2 == 0)

next

print(i\*i)

}

**. Histograms**

* **Basic Histogram**
  + hist() function in R: creates a histogram by binning a variable into fixed-width buckets.
  + Example:

r

Copy code

custdata <- read.table('custdata.tsv',header=T, sep='\t')

hist(custdata$age)

* + Histogram visualization of age distribution in customer data.
* **Customizing Histograms**
  + breaks: specifies where the breaks are.
  + xlim: defines the start and end points of the x-axis.
  + freq: TRUE for raw counts, FALSE for density (normalized by the total count).
  + Example:

x <- custdata$age

hist(x, breaks=seq(0,150,1), xlim=c(0,100), freq = FALSE)

* + Adjusts the histogram to display density instead of raw counts.
* **Adding Titles**
  + Use attributes within the hist() function:

hist(custdata$age, main="Distribution of age", xlab="age")

* + Use title() function:

hist(custdata$age)

title('Distribution of age', xlab='age')

**2. A Layered Grammar of Graphics with ggplot**

* **Introduction to ggplot2**
  + ggplot2: an elegant and versatile library for creating graphs in R.
  + Implements the grammar of graphics: a coherent system for describing and building graphs.
  + Basic structure:

ggplot(data = <DATA>) + <GEOM\_FUNCTION>(mapping = aes(<MAPPINGS>))

* + Replace <DATA>, <GEOM\_FUNCTION>, and <MAPPINGS> with appropriate dataset, geometry function (chart type), and data selection.
* **Histograms with ggplot2**
  + Example:

library(ggplot2)

ggplot(data = custdata) +

geom\_histogram(mapping = aes(x=age), binwidth=5, fill="gray")

* + Creates a histogram with age data from the customer dataset.
* **Density Plots with ggplot2**
  + Continuous histogram where the area under the density plot equals 1.
  + Example:

ggplot(custdata) + geom\_density(aes(x=age)) +

theme(text = element\_text(size = 24))

**3. Exploratory Data Analysis (EDA)**

* **Definition and Purpose**
  + EDA: uses visualization and transformation to explore data systematically.
  + Iterative cycle:
    - Generate questions about data.
    - Search for answers through visualization, transformation, and modeling.
    - Refine questions and generate new ones.
  + Goal: develop an understanding of the data.
* **Key Concepts in EDA**
  + **Variation within Variables**: Understanding how values vary within a single variable.
  + **Covariation between Variables**: Exploring relationships between two or more variables.
* **Questions to Guide EDA**
  + Identify common values, rare values, and unexpected patterns.
  + Determine subgroups within data by clustering similar values.
  + Compare two or more variables to understand their relationships and patterns.
* **Terms Used in EDA**
  + **Variable**: A measurable quantity or quality.
  + **Value**: State of a variable at the time of measurement.
  + **Observation**: Set of measurements under similar conditions.
  + **Tabular Data**: Tidy data with values in cells, variables in columns, and observations in rows.
* **Visual Tools in EDA**
  + **Histograms**: Show distribution and common/rare values.
  + **Bar Charts**: Like histograms but for categorical data.
  + **Scatter Plots**: Visualize relationships between two continuous variables.