CSSCR R Workshop

Dadmehr Didgar

2024-1-22

Introduction to R using R Studio

R is a versatile programming language widely used for statistical analysis and data visualization. Its comprehensive libraries enable effective data manipulation, making it essential for researchers and data scientists. This workshop provides a foundational understanding of R's syntax and functions, focusing on data handling and graphical representation.

Simple Calculation

```
3+4
```

```
## [1] 7
```

```
2/3
```

```
## [1] 0.6666667
```

```
5*2
```

```
## [1] 10
```

`x = 1` vs. `x <- 1`

In R, `x = 1` and `x <- 1` both assign the value `1` to `x`. The traditional `<-` operator is preferred for variable assignments due to its clarity and readability, while `=` is commonly used for specifying function arguments. Though functionally similar in most cases, `<-` is the conventional choice in R scripting for assignment operations.

Data Type & Assign Values

```
2.1
```

```
## [1] 2.1
```

```
F
```

```
## [1] FALSE
```

```
"Happy"
```

```
## [1] "Happy"
```

```
"2"
```

```
## [1] "2"
```

```r
a <- 3
## assign the character 2.1 to object called b
b <- "2.1"
## assign the character hello to object called bb
c <- "happy"
## assign the value of object a to object called c
d <- a
```

Built-in Mathematical Functions

R provides a variety of built-in mathematical functions. Here are a few examples:

- **Square Root**: The `sqrt()` function computes the square root of a number. For example, `sqrt(16)` will give 4.
- **Exponential**: The `exp()` function calculates the exponential of a number. For instance, `exp(1)` computes e^1, which is approximately 2.7182818.
- **Logarithm**: The `log()` function computes logarithms. `log(10)` gives the natural logarithm of 10, equal to 2.3025851.
- **Trigonometry**: Functions like `sin()`, `cos()`, and `tan()` are used for trigonometric calculations.

These functions exemplify the simplicity and power of R for mathematical computations.

Types of Objects

```r
# vector
numbers <- c(1,4,2)

colors <- c("lightgreen", "pink", "blue")

# data frame
demo_data <-    data.frame(
  gender = c("Male", "Male","Female"),
  height = c(152, 171.5, 165),
  weight = c(81,93, 78),
  Age    = c(42,38,26)
  )

# list
mylist <- list(2.1, c(1,3,7), c("abc", "def"), demo_data)
```

If Clause in R

```r
x <- 5

if (x > 0) {
  print("x is positive")
} else {
  print("x is not positive")
}
```

```
## [1] "x is positive"
```

For Loop in R

```
for (i in 1:5) {
  print(i)
  #print(paste("Value of i is", i))
}
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
```

Census Data

CSSCR_data is a small random sample (0.005%) of the census data for 2022

```
# sampled_data <- data[sample(nrow(data), nrow(data) * 0.00005), ]

CSSCR_data <- data.frame(AGE = c(36, 66, 48, 84, 76, 61, 69, 33, 95, 61, 83, 69, 29, 73, 28, 80, 49, 48


CSSCR_data$INCTOT <- c(100000, 11600, 105000, 79400, 2300, 73000, 138000, 15000, 24000, 96900, 9300, 350
 14000, 45000, 46000, 65000, 39000, 36600, 75000, 56000, 25000, 170000, 16800, 18300, 400, 40000, 1900,
```

Data Analysis Essentials

The R code provided performs fundamental data analysis operations on the `CSSCR_data` dataset.

```
dim(CSSCR_data)
```

```
## [1] 124   2
```

```
mean(CSSCR_data$INCTOT)
```

```
## [1] 50146.37
```

```
median(CSSCR_data$INCTOT)
```

```
## [1] 30550
```

```
mean(CSSCR_data$AGE)
```

```
## [1] 51.5
```

```
median(CSSCR_data$AGE)
```

```
## [1] 53
```

Count

```
#install.packages("dplyr")
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
```

```
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

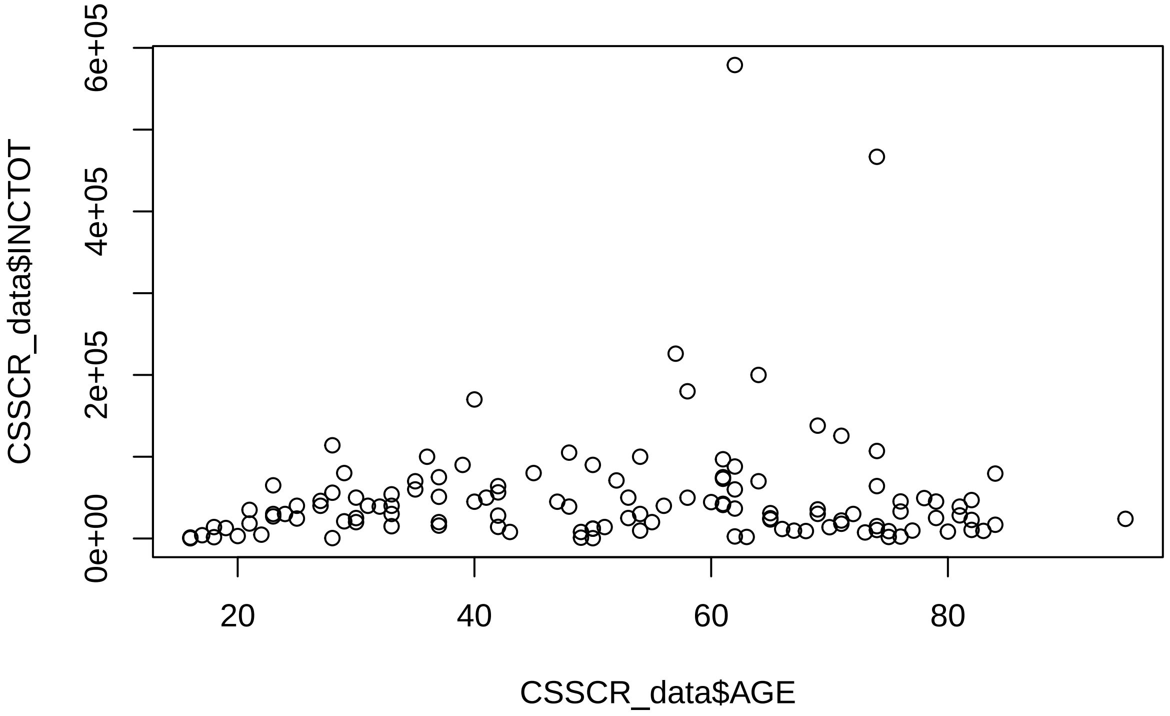
```r
count(CSSCR_data,AGE > 30)
```

```
##   AGE > 30  n
## 1    FALSE 27
## 2     TRUE 97
```

```r
help("count")
?count
```

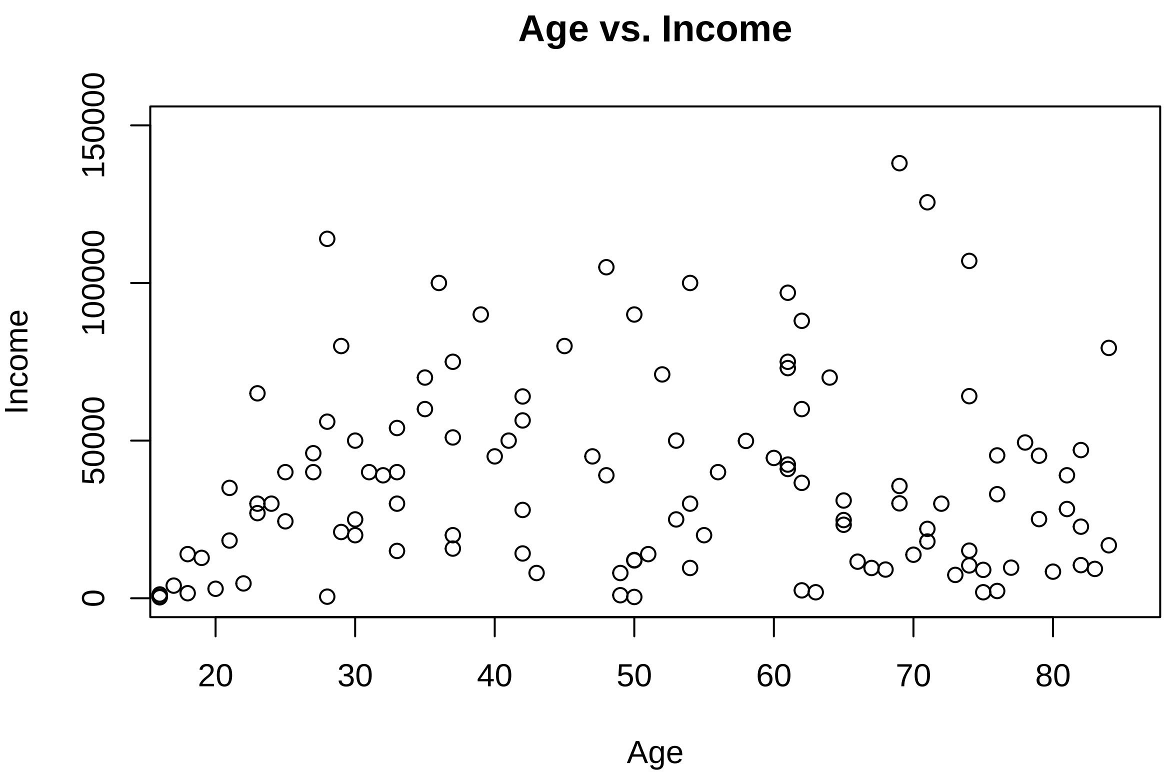
Simple Plot

```r
plot(CSSCR_data$AGE, CSSCR_data$INCTOT)
```



Less Simple Plot

```r
plot(CSSCR_data$AGE, CSSCR_data$INCTOT,
     main = "Age vs. Income",
     xlab = "Age",
     ylab = "Income",
     ylim = c(0,150000),
     xlim = c(18,85))
```



```
?plot
```

```
## Help on topic 'plot' was found in the following packages:
##
##   Package               Library
##   graphics              /Library/Frameworks/R.framework/Versions/4.3-arm64/Resources/library
##   base                  /Library/Frameworks/R.framework/Resources/library
##
##
## Using the first match ...
```

Fancy Plot

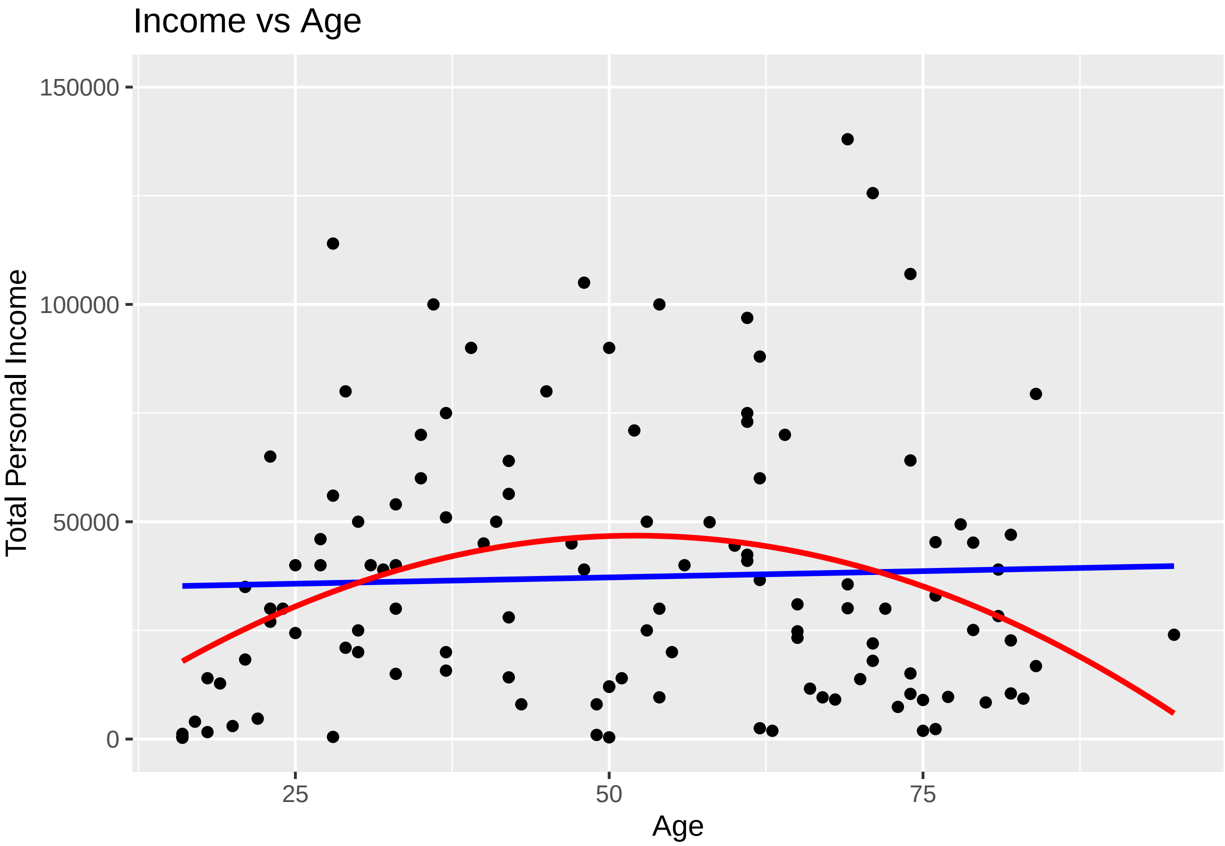
```
# Install and load the ggplot2 package
#install.packages("ggplot2")

library(ggplot2)

# Plotting
ggplot(CSSCR_data, aes(x = AGE, y = INCTOT)) +
  geom_point() +  # Scatter plot
  geom_smooth(method = "lm", se = FALSE, color = "blue") +  # Linear fit
  geom_smooth(method = "lm", formula = y ~ poly(x, 2), se = FALSE, color = "red") +  # Quadratic fit
  ylim(0, 150000) +
  labs(title = "Income vs Age", x = "Age", y = "Total Personal Income")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Removed 6 rows containing non-finite values (`stat_smooth()`).
## Removed 6 rows containing non-finite values (`stat_smooth()`).
## Warning: Removed 6 rows containing missing values (`geom_point()`).
```


Income vs Age
Total Personal Income
Age