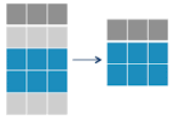
Lesson 04
Data Exploration

Outline

- Subsetting vectors and data frames
  - Extract rows and/or columns we want
  - Omit rows and/or columns we don't want
  - Using `order()` function for sorting
  - Using relational and logical operators
  - Using `with()` function to simplify conditions
  - Adding calculated columns to the data frame
  - Using `subset()` function for efficiency
  - Using `which()` function for more complex subsetting
  - Using `%in%` sub-set operator for selection
- Summarizing vectors and data frames
  - Sum, average, median, min, max, …

Subset Data Frame Rows in R

Data Subsetting – Relational Operators



- A more systematic review of some of the already familiar vector / data frame operations
- Relational operators and negative integers for omitting

```
loans_mortg_df <- loans_df[loans_df$loanType
== "Mortg", -c(2:5,9)]
```

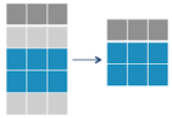
- Relational operators and positive integers for keeping

```
loans_no_Taos_df <- loans_df[loans_df$city !=
"Taos", c(1,4,6:10)]
```

- Using `order()` function for sorting

```
sort_mth_pmt <- order(loans_df$mthPmt,
decreasing = TRUE)
loans_df[sort_mth_pmt, ]
```

<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to
==	exactly equal to
!=	not equal to



Data Subsetting – Logical Operators

!x	Not x
x | y	x OR y
x & y	x AND y

- Logical OR operator | (pipe symbol)

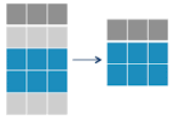
```
# Select records on loans made in
either January or beyond April of 2030
loans_jan_apr_df <- loans_df[loans_df$loanDate <
"2030-02-01" | loans_df$loanDate >= "2030-04-01", ]
```

- Logical AND operator & (amp symbol)
  - Simplifying using `with()` function

```
# Select records on Taos mortgage loans
loans_mortg_Taos_df <- loans_df[with(loans_df, city ==
"Taos"  & loanType=="Mortg"), ]
```

- Combining multiple relational and logical operators

```
# Select records on mortgage loans that are either over
500,000 or under 200,000
loans_mortg_high_low_df <- loans_df[with(loans_df,
loanType == "Mortg" & (amount > 500000 | amount <
200000)), ]
```



Data Subsetting and Summarizing

- Adding columns to data frame
  - Use $ to create a new column with given name followed by calculation involving the existing columns

```
loans_df$totPmt <- loans_df$mthPmt *
loans_df$loanTerm * 12
```

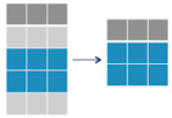
- Simple data summary

```
avg_tot_pmt <- mean(loans_df$totPmt)
```

- Using summaries to subset data

```
above_avg_tot_pmt_df <- loans_df[loans_df$totPmt >
avg_tot_pmt, c(1,9:11)]
```

Subset Data Frame Rows in R



Data Exploration – Product Groups

- Run the code generating the `prod_df` data frame from the Product_Groups view
- Subset the result for inventory products only

```
# Data frame of inventory products only
na_prod_groups <- is.na(prod_df$ProdGroup)
prod_inv_df <- prod_df[na_prod_groups == FALSE, ]
```

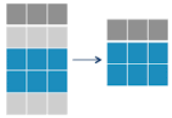
- Average retail price

```
avg_retail_price <- mean(prod_inv_df$RetailPrice)
```

- Products above average retail price

```
prod_above_avg_df <-
prod_inv_df[prod_inv_df$RetailPrice >
avg_retail_price,]
nrow(prod_above_avg_df)
```

Subset Data Frame Rows in R



Data Exploration – Retail Customers

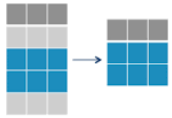
- Retail_Customers view
- Using `subset()` function

```
gender <- "F"
age_low <- 40
age_high <- 45
cust_F_age_df <- subset(cust_df, Gender == gender &
(Age >= age_low & Age <= age_high), select =
c(FirstName, LastName, Age))
```

- Using %in% operator

```
mw_states <- c("MN","WI","ND","SD","IA")
cust_mw_states_df <- cust_df[cust_df$State %in%
mw_states, c(1:2,5:8)]
```

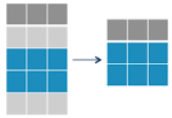
Subset Data Frame Rows in R



Data Exploration – Employee Orders

- Employee_Orders view
- Using which() function to find the row numbers
  - `empl_low_ords <- which(empl_df$NumOrds == 10 | empl_df$NumOrds == 20)`
  - `empl_high_ords <- which(empl_df$NumOrds > 30)`
- Using row numbers to find average employee salary

```
avg_sal_low <-
mean(empl_df$Salary[empl_low_ords])
avg_sal_high <-
mean(empl_df$Salary[empl_high_ords])
```



Data Exploration – Customer Sales

- Customer Sales view
- Demonstrates efficiency of `which()` function
- Using `which()` to retrieve line items for product

```
prod_name <- "Hockey Stick"
prod_sold <- which(sales_df$ProdName == prod_name)
prod_sold
```

- Using only the logical test to retrieve line items for a particular customer

```
cust_id <- 100
cust_sold <- sales_df$CustomerID == cust_id
cust_sold
sum(cust_sold) # Confirms only 5 TRUE values
```

Summary

- Presented a systematic review of vector and data frame subsetting
- Basic subsetting with row and column numbers
- More complex subsetting with relational and logical operators
- The use of a `subset()` and `which()` subsetting functions
- Basic data summaries with `sum()` and `mean()` functions
- Additional examples in class and next assignment