Welcome to

instats

The Session Will Begin Shortly

START

Statistics in R with Tidyverse

Session 1: Introduction to R: Basics and Advanced Techniques

Welcome and Introduction

Dr. Chester Ismay

- PhD in Statistics
- Worked in academia, online education, corporate training, tech bootcamps, and independent consulting
- Currently,
  - Faculty Member in Data Analytics, Portland State University
  - Vice President of Data and Automation, MATE Seminars
  - Freelance data scientist and educator
- Fun Fact: Slept a night or eaten a meal in all 50 US states

instats

Course Learning Objectives

By the end of this course, you will be able to

- Perform data wrangling techniques in R via the `tidyverse`
- Develop skills in data visualization with `ggplot2`
- Apply fundamental concepts of statistical inference with `infer`
- Build and interpret regression models with `moderndive`
- Integrate Theory-Based and Simulation-Based Approaches

Agenda

Day 1: Working with Data in R - Explore, Visualize, Wrangle, Import

- Session 1: Introduction to R – Basics and Advanced Techniques
- Session 2: Data Visualization using `ggplot2`
- Session 3: Data Wrangling and Tidy Data

Introduction to R and RStudio

- R: programming language mainly for statistical computing and data analysis
- RStudio: IDE
- R vs RStudio

R: Engine	RStudio: Dashboard

Installing R and RStudio

- R: https://cloud.r-project.org/
- RStudio: https://posit.co/download/rstudio-desktop/
- Download and install for your operating system

Coding in R

- Commands entered as code in the Console or via scripts.
- Key concepts include objects, vectors, and data types
- Conditional statements and functions help perform tasks
- Learning to code takes frequent practice, but it is one of the most rewarding things you can do!

Using R packages

- Extend R’s capabilities with additional functions and/or datasets
- First install the package with `install.packages()`
- Load the package using `library()`

R: A new phone	R Packages: Apps you can download

orms app
a the App Store or the Google Play depe
Available on the App Store
ANDROID APP ON Google play
Android, the two operating

Exploring Data in R with RStudio

- Data frames are like tables with rows and columns
- Use `View()`, `glimpse()`, or `kable()` to inspect
- The $ operator extracts columns from data frames
- Identification versus measurement variables/columns

Demo & Exercises

Q & A

STOP

Welcome to

instats

The Session Will Begin Shortly

START

Statistics in R with Tidyverse

Session 2: Data Visualization using ggplot2

instats

Introduction to Data Visualization

- Insights that raw data alone cannot provide
- `ggplot2` package based on Grammar of Graphics by Leland Wilkinson
- Visualizations help to identify outliers, distributions, and relationships

Grammar of Graphics

- A statistical graphic maps data variables to aesthetic attributes
- Key components:

1. `data`: The dataset
2. `geom`: The geometric objects (points, lines, bars)
3. `aes`: Aesthetic attributes like position, color, shape, size

- Create visualizations by layering these components in `ggplot()`

The Five Named Graphs

- Essential tools for data visualization
- Scatterplots, linegraphs, histograms, boxplots, and barplots
    - Each type works best for different data relationships and distributions
    - Goal is to uncover trends, patterns, and outliers in data

Scatterplots

- Display relationships between two numerical variables
- Using `geom_point()`
- Customizing points (`color`, `shape`, `size`)
- **Tip**: Handling overplotting
    - `alpha` transparency
    - jittering with `geom_jitter()`

Linegraphs

- Display trends over time or relationships between two sequential variables
- Use `geom_line()`
- Commonly used for time-based data (hours, days, weeks, etc.)
- **Tip**: Avoid using linegraphs when the x-axis variable has no inherent order

Histograms

- Display the distribution of a single numerical variable
- Use `geom_histogram()`
- Visualize data spread, center, and frequency of values
- **Tip**: Adjust bin width or number of bins for better data representation

Boxplots

- Summarize numerical data using quartiles and medians
- Use `geom_boxplot()`
- Effective for identifying data spread and detecting outliers
- **Tip**: Use boxplots for comparing distributions across groups

Barplots

- Display the distribution of a categorical variable's frequencies
- Use `geom_bar()`or `geom_col()`
- Barplots are ideal for comparing frequencies of categories or groups
- Tip: Use `geom_bar()` for raw (uncounted) data and `geom_col()` for pre-counted data

Demo & Exercises

Q & A

STOP

Welcome to

instats

The Session Will Begin Shortly

START

Statistics in R with Tidyverse

Session 3: Data Wrangling and Tidy Data

instats

Data Wrangling

- Overview of the `tidyverse`
- Importance of Data Wrangling in Research
- Key Packages: `tidyr`, `dplyr`

Filter Rows

- Use `filter()`to select rows based on conditions
- Focuses on rows
    - Similar to `slice()` which selects rows by position, not condition
- Combine conditions with `&` (AND) and `|` (OR)
- **Tip**: Use `!=` to filter out specific values

Mutate Columns

- Use `mutate()`to create new columns based on existing ones
- Adds new columns; unlike `transmute()`, which drops all other columns
- Useful for transforming or calculating new values from existing data
- **Tip**: Can also be used to modify an existing column

Summarize Data

- Use `summarize()` to calculate summary statistics
- Reduces data to a single row or value; unlike `mutate()` which keeps

original data format

- **Tip**: Can handle missing data with `na.rm = TRUE`

Group By and Summarize

- Use `group_by()` to split data into groups, then apply `summarize()`
- Organizes data into groups; unlike `arrange()`, which only sorts data
- Combine `group_by()` with `summarize()` to create grouped statistics
- **Tip**: `ungroup()` data after grouping if further processing is needed

Arrange Data

- Use `arrange()` to sort rows based on specific columns
- Sorts data; unlike `filter()` which selects rows without changing order
- **Tip**: Sort in ascending order by default; use `desc()` for descending

Select Columns

- Use `select()` to choose specific columns
- Different from `mutate()`, which adds new columns
- Can deselect columns using – (e.g., `select(-year)`)
- **Tip**: Use helpers like `starts_with()` to select columns by pattern

Tidy Data

- "Tidy" data means
    - each variable has its own column
    - each observation has its own row
    - each kind of thing you’re observing is its own table
- Different from "wide" data in that it is often longer to be tidy
- **Tip**: Use `pivot_longer()` to convert wide data for easier analysis

Pipe Operator (|>)

- Use the pipe operator to chain multiple operations together
- Chains operations unlike using nested functions, which is harder to read
- Often improves workflows
- **Tip**: Think of |> as “then” to improve readability

Demo & Exercises

Q & A

STOP