MVA Lab 1: Introduction to Data Analysis in R

Dani

1 Introduction

This document provides an introduction to data analysis in R, covering essential topics like data importing, basic data manipulation, and visualization. Each section builds upon the fundamentals of R programming, offering practical code examples, explanations, and visualization techniques to aid in understanding multivariate data analysis.

2 Data Importing

- In R, data can be imported using various functions depending on the file format. For CSV files, the function `read.csv()` is commonly used.
- Example: Importing the `cars.csv` dataset:

```
# Importing a .csv file
cars <- read.csv("/path/to/cars.csv", stringsAsFactors = TRUE)

# View the imported data
View(cars)
```

- **Concept**: 'stringsAsFactors = TRUE' converts character columns into factors automatically. This is useful when dealing with categorical data.

3 Understanding the Dataset

- Use `str()` to understand the structure of a data frame, and `summary()` to generate basic descriptive statistics:

```
# Check the structure of the data
str(cars)

# Summary of the data
summary(cars)
```

- **Explanation**: `str()` provides an overview of the data types and the first few entries for each column, while `summary()` calculates the mean, median, quartiles, and missing values for numeric columns.

4 Variable Transformation

- Convert variables to factors for categorical analysis:

```
# Convert year and number of cylinders to factors
cars$year <- as.factor(cars$year)
cars$cylinders <- as.factor(cars$cylinders)

# Updated summary
summary(cars)
```

- **Concept**: Factors are used to handle categorical data in R, allowing for easier data manipulation and visualization.

5 Exploring the Data: Visualizations

5.1 Bar Plots for Categorical Variables

- Visualize the distribution of categorical variables using `barplot()`:

```
# Bar plot for year
barplot(summary(cars$year), ylab = "Frequency", main = "Distribution of Cars by
    Year")

# Bar plot for brand
barplot(summary(cars$brand), main = "Distribution of Cars by Brand")
```

- **Concept**: Bar plots display the frequency of each category, making them ideal for understanding distributions of categorical variables.

5.2 Contingency Tables and Proportions

- Create contingency tables to understand joint distributions:

```
# Reclassify 'year' into categories
cars$year <- as.numeric(as.character(cars$year))
cars$year.cat <- cut(cars$year, breaks = c(1970, 1975, 1979, 1983))

# Contingency table of year and brand
t <- table(cars$year.cat, cars$brand)
```

- **Proportions**:

```
# Overall proportions
prop.table(t)

# Row-wise proportions
prop.table(t, 1)

# Column-wise proportions
prop.table(t, 2)
```

- **Explanation**: Proportions help in understanding how categories relate to one another, providing insight into patterns within the data.

6 Visualizing Quantitative Variables

6.1 Histograms for Continuous Variables

- Use `hist()` to visualize distributions of continuous variables:

```
# Visualize distributions
hist(cars$mpg, main = "Distribution of MPG", xlab = "Miles per Gallon")
hist(cars$hp, main = "Distribution of Horsepower", xlab = "Horsepower")
hist(cars$weightlbs, main = "Distribution of Weight", xlab = "Weight (lbs)")
```

- **Best Practice**: Customize histograms with labels and titles to improve clarity.

6.2 Boxplots for Comparing Distributions

- Boxplots are ideal for comparing distributions across different categories:

```
# Compare weight distributions across brands
boxplot(cars$weightlbs ~ cars$brand, main = "Weight Distribution by Brand",
        ylab = "Weight (lbs)", xlab = "Brand")
```

- **Interpretation**: Boxplots show the median, quartiles, and potential outliers, providing a concise summary of distributional characteristics.

7 Descriptive Statistics

- Calculate descriptive statistics (mean, median, standard deviation) for selected variables:

```
# Calculate mean, median, and standard deviation
mean_values <- apply(cars[, c(1, 4, 5)], 2, mean, na.rm = TRUE)
median_values <- apply(cars[, c(1, 4, 5)], 2, median, na.rm = TRUE)
std_dev <- apply(cars[, c(1, 4, 5)], 2, sd, na.rm = TRUE)

# Create a descriptive statistics table
tab <- rbind(mean_values, median_values, std_dev)
rownames(tab) <- c("Mean", "Median", "Standard Deviation")
tab
```

- **Concept**: Use the `apply()` function to compute summary statistics across columns efficiently.

8 Advanced Table Operations

- Use `tapply()` to compute group-wise summaries:

```
# Grouped summary of weight by brand
tapply(cars$weightlbs, cars$brand, summary)
```

- **Key Insight**: `tapply()` simplifies the process of calculating statistics for each level of a categorical variable.

9 Conclusion

This introductory guide has covered essential techniques for importing, exploring, visualizing, and summarizing data in R. Understanding these foundational tools is critical for more complex analyses, as they allow you to efficiently preprocess and analyze datasets, whether for multivariate analysis, data mining, or machine learning.