Knowledge Discovery by Accuracy Maximization

Stefano Cacciatore, Leonardo Tenori

2025-06-02

Introduction

The `clinical` package is designed to facilitate exploratory data analysis and statistical testing on clinical datasets. This vignette presents the full usage of all core functions included in the package.

2 Installation

2.1 Installation via CRAN

The R package clinical is part of the Comprehensive R Archive Network (CRAN)[1]. The simplest way to install the package is to enter the following command into your R session: `install.packages("clinical")`.

2.3 Compatibility issues

All versions downloadable from CRAN have been built using R version, R.3.2.3. The package should work without major issues on R versions > 3.5.0.

3 Getting Started

To load the package, enter the following instruction in your R session:

If this command terminates without any error messages, you can be sure that the package has been installed successfully. The clinical package is now ready for use.

The package includes both a user manual (this document) and a reference manual (help pages for each function). To view the user manual, enter `vignette("clinical")`. Help pages can be viewed using the help command `help(package="clinical")`.

Prostate Data

The `clinical` package includes a simulated dataset representing clinical information from patients diagnosed with prostate cancer. This dataset is provided as a `data.frame` and is intended for demonstration and instructional purposes.

The dataset includes the following variables:

- **Hospital**: Factor indicating the hospital where the patient was treated.
- **Gender**: Factor indicating the patient's gender.
- **Gleason**: Ordered factor representing the Gleason score assigned to the tumor.
- **BMI**: Numeric value for the patient's Body Mass Index.
- **Age**: Numeric value for the patient's age (in years).
- **Hypertension**: Factor indicating whether the patient has hypertension.

To load the dataset:

[1]https://cran.r-project.org/

```
data(prostate)
head(prostate)
```

```
##     Hospital Gender Gleason score BMI Age hypertension
## 1 Hospital A   Male           3+3  35  71           no
## 2 Hospital A   Male           3+3  36  75           no
## 3 Hospital A   Male           3+4  29  78           no
## 4 Hospital A   Male           4+3  28  76           no
## 5 Hospital A   Male           4+4  35  73           no
## 6 Hospital A   Male           3+4  36  74           no
```

Function: `txtsummary`

The `txtsummary()` function provides a concise textual summary of a numeric variable using either the **mean** or **median**, along with a measure of variability such as the **interquartile range (IQR)**, **95% confidence interval**, **standard deviation**, or **full range**.

This is particularly useful when preparing reports or markdown documents where inline descriptive statistics are needed in a clean format.

Example with the `prostate` dataset

```
# Summarize Age using mean and IQR
txtsummary(prostate$Age, f = "mean", digits = 2, range = "IQR")
```

```
## [1] "67.52 [56 75]"
```

Function: `continuous.test`

Comparing Continuous Variables Across Groups

The `continuous.test()` function allows comparison of a continuous variable across groups. It returns a formatted summary of the data (e.g., mean and standard deviation or median and IQR) for each group, along with a p-value from a statistical test. This is useful for generating clean, publication-ready result tables.

Function Parameters

- `feature`: A string indicating the name of the variable.
- `values`: A numeric vector containing the continuous data.
- `group`: A factor or character vector indicating group membership.
- `center`: The measure of central tendency, either `"mean"` or `"median"`.
- `range`: A measure of variability: `"sd"`, `"IQR"`, `"range"`, or `"95%CI"`.
- `method`: Statistical method to use: `"parametric"` or `"non-parametric"`. The function uses:
  - t-test (2 groups) or ANOVA (>2 groups) for parametric
  - Wilcoxon (2 groups) or Kruskal-Wallis (>2 groups) for non-parametric

---

Example 1: Wilcoxon Rank-Sum Test (2 Groups, Non-Parametric)

```
# Non-parametric comparison using Wilcoxon test
result_wilcox <- continuous.test(
  name = "Age",
  x = prostate$Age,
  y = prostate$Hospital,
  center = "median",
  range = "IQR",
  method = "non-parametric"
)

print(result_wilcox)

##             Feature Hospital A     Hospital B  p-value
## 1 Age, median [IQR] 63 [56 74] 74 [57.5 79.5] 3.07e-01
```

Example 1: Wilcoxon Rank-Sum Test (2 Groups, Non-Parametric)

```
# Non-parametric comparison using Wilcoxon test
result_wilcox <- continuous.test(
  name = "Age",
  x = prostate$Age,
  y = prostate$Hospital,
  center = "median",
  range = "IQR",
  method = "non-parametric"
)

print(result_wilcox)

##             Feature Hospital A     Hospital B  p-value
## 1 Age, median [IQR] 63 [56 74] 74 [57.5 79.5] 3.07e-01
```

Function: `categorical.test`

Comparing Categorical Variables Across Groups

The `categorical.test()` function compares categorical variables across groups and returns a formatted summary table with the test result. It automatically selects the appropriate statistical test depending on whether the categorical variable is **ordered** or not:

- **Unordered factor**: Uses **Fisher's exact test** (2 groups) or **Chi-squared test** (>2 groups).
- **Ordered factor**: Uses the **Jonckheere–Terpstra test** to detect monotonic trends across ordered categories.

The output is suitable for inclusion in summary tables or reports.

Function Parameters

- `feature`: A string indicating the name of the categorical variable.
- `values`: A factor or ordered factor representing the categorical data.
- `group`: A factor or character vector indicating group membership.

Example 1: Unordered Categorical Variable (Fisher's Exact Test)

```
# Compare Gender (unordered factor) across hospitals
categorical_test_result <- categorical.test(
  name = "Gender",
  x = prostate$Gender,
  y = prostate$Hospital
)
```

```
print(categorical_test_result)
```

```
##   Feature          Hospital A   Hospital B   p-value
## v "Gender"         ""           ""           ""
##   "   Male, n (%)" "81 (100.0)" "24 (100.0)" ""
## attr(,"p-value")
## [1] NA
## attr(,"shapiro test")
## [1] NA
```

Example 2: Ordered Categorical Variable (Jonckheere–Terpstra Test)

```
# Compare Gleason score (ordered factor) across hospitals
categorical_test_result <- categorical.test(
  name = "Gleason",
  x = prostate$Gleason,
  y = prostate$Hospital
)

print(categorical_test_result)
```

```
##   Feature         Hospital A  Hospital B p-value
## v "Gleason"       ""          ""         "1.33e-01"
##   "   3+3, n (%)" "30 (37.0)" "7 (29.2)" ""
##   "   3+4, n (%)" "30 (37.0)" "5 (20.8)" ""
##   "   4+3, n (%)" "8 (9.9)"   "6 (25.0)" ""
##   "   4+4, n (%)" "13 (16.0)" "6 (25.0)" ""
## attr(,"p-value")
## [1] 0.1326
## attr(,"shapiro test")
## [1] NA
```

Function: `correlation.test`

Computes Pearson, Spearman, or MINE correlation between two numeric vectors.

```
correlation_result <- correlation.test(prostate$Age, prostate$BMI, method = "spearman", name = "Age vs I
```

```
## Warning in cor.test.default(x, y, method = "spearman"): Cannot compute exact
## p-value with ties
```

```
print(correlation_result)
```

```
##      Feature  rho  p-value
## 1 Age vs BMI 0.05 6.15e-01
```

Function: `multi_analysis`

Applies a test (continuous or correlation) across multiple features of a dataset.

```
multi_cont <- multi_analysis(prostate[, c("Age", "BMI")], prostate$Hospital, FUN = "continuous.test")
print(multi_cont)
```

```
##             Feature Hospital A     Hospital B  p-value      FDR
## 1 Age, median [IQR] 63 [56 74] 74 [57.5 79.5] 3.07e-01 3.07e-01
## 2 BMI, median [IQR] 32 [25 35]  26 [25 30.25] 9.66e-03 1.93e-02
```

```
multi_corr <- multi_analysis(prostate[, c("Age", "BMI")], prostate$BMI, FUN = "correlation.test")
print(multi_corr)
```

```
##   Feature    r p-value     FDR
## 1     Age 0.03 7.5e-01 7.5e-01
## 2     BMI 1.00   0e+00 0.0e+00
```

Function: `intersect`

Finds the intersection of multiple vectors.

```
v1 <- c("A", "B", "C")
v2 <- c("B", "C", "D")
v3 <- c("C", "B", "E")

intersect(v1, v2, v3)
```

```
## [1] "B" "C"
```

Function: `frequency_matching`

Matches samples across classes (e.g., control vs case) by discretizing numeric features into bins and stratifying selection.

```
hosp=prostate[,"Hospital"]
gender=prostate[,"Gender"]
GS=prostate[,"Gleason score"]
BMI=prostate[,"BMI"]
age=prostate[,"Age"]

A=categorical.test("Gender",gender,hosp)
B=categorical.test("Gleason score",GS,hosp)

C=continuous.test("BMI",BMI,hosp,digits=2)
D=continuous.test("Age",age,hosp,digits=1)

# Analysis without matching
rbind(A,B,C,D)
```

```
##               Feature Hospital A     Hospital B  p-value
## v              Gender
##           Male, n (%) 81 (100.0)     24 (100.0)
## v1      Gleason score                            1.32e-01
## X         3+3, n (%)  30 (37.0)       7 (29.2)
## X.1       3+4, n (%)  30 (37.0)       5 (20.8)
## X.2       4+3, n (%)    8 (9.9)       6 (25.0)
## X.3       4+4, n (%)  13 (16.0)       6 (25.0)
## 1   BMI, median [IQR] 32 [25 35]  26 [25 30.25] 9.66e-03
## 11  Age, median [IQR] 63 [56 74] 74 [57.5 79.5] 3.07e-01
```

```
# The order is important. Right is more important than left in the vector
# So, Ethnicity will be more important than Age
var=c("Age","BMI","Gleason score")
data.categorized=prostate[,var]

# Extract the Age vector
```

```
x <- data.categorized[["Age"]]

# Compute quantiles (0%, 25%, 50%, 75%, 100%) with NA handling
breaks <- quantile(x, probs = c(0, 0.25, 0.5, 0.75, 1), na.rm = TRUE)

# Apply the cut and update the Age column with labeled bins
data.categorized[["Age"]] <- cut(x, breaks = breaks, include.lowest = TRUE)

# Extract the Age vector
x <- data.categorized[["BMI"]]

# Compute quantiles (0%, 25%, 50%, 75%, 100%) with NA handling
breaks <- quantile(x, probs = c(0, 0.25, 0.5, 0.75, 1), na.rm = TRUE)

# Apply the cut and update the Age column with labeled bins
data.categorized[["BMI"]] <- cut(x, breaks = breaks, include.lowest = TRUE)

times=c(1,1)
names(times)=c("Hospital A","Hospital B")
t=frequency_matching(data.categorized,prostate[,"Hospital"],times=times)



newdata=prostate[t$selection,]

hosp.new=newdata[,"Hospital"]
gender.new=newdata[,"Gender"]
GS.new=newdata[,"Gleason score"]
BMI.new=newdata[,"BMI"]
age.new=newdata[,"Age"]

A=categorical.test("Gender",gender.new,hosp.new)
B=categorical.test("Gleason score",GS.new,hosp.new)

C=continuous.test("BMI",BMI.new,hosp.new,digits=2)
D=continuous.test("Age",age.new,hosp.new,digits=1)

# Analysis with matching
rbind(A,B,C,D)
```

```
##                Feature    Hospital A     Hospital B  p-value
## v               Gender
##           Male, n (%)    24 (100.0)     24 (100.0)
## v1      Gleason score                                  1e+00
## X          3+3, n (%)      7 (29.2)       7 (29.2)
## X.1        3+4, n (%)      5 (20.8)       5 (20.8)
## X.2        4+3, n (%)      6 (25.0)       6 (25.0)
## X.3        4+4, n (%)      6 (25.0)       6 (25.0)
## 1   BMI, median [IQR] 29 [25 32.75]  26 [25 30.25]    3e-01
## 11  Age, median [IQR]  61 [56 73.2] 74 [57.5 79.5] 2.51e-01
```

Conclusion

The `clinical` package provides an extensive toolkit for evaluating clinical datasets, from statistical comparisons to frequency matching and summarization. This vignette serves as a comprehensive guide for using each function effectively.