Python ↔ R Stats Cheat Sheet: Common Tests & Summaries

R: base stats (+ optional car). Python: numpy/scipy/statsmodels. Snippets are minimal and copy■paste friendly.

Setup (example data)

R	Python
`set.seed(1) x <- rnorm(30, mean=0) y <- rnorm(30, mean=0.5) g <- factor(rep(c('A','B'), each=15)) df <- data.frame(x=x, y=y, g=g) # For logistic examples: df$z <- rbinom(30, 1, plogis(0.4*x - 0.2))`	`import numpy as np import pandas as pd from scipy import stats import statsmodels.api as sm import statsmodels.formula.api as smf rng = np.random.default_rng(1) x = rng.normal(0, 1, 30) y = rng.normal(0.5, 1, 30) g = np.array(['A']*15 + ['B']*15) df = pd.DataFrame({'x': x, 'y': y, 'g': g}) df['z'] = rng.binomial(1, 1/(1+np.exp(-(0.4*x - 0.2))))`

Summary statistics

R	Python
`mean(x); sd(x); var(x) median(x); quantile(x, c(.25,.5,.75)) summary(df)`	`x.mean(); x.std(ddof=1); x.var(ddof=1) np.median(x); np.quantile(x, [0.25, 0.5, 0.75]) df.describe(include='all')`

Missing values (NA / NaN)

R	Python
`x2 <- c(1, NA, 3) is.na(x2) mean(x2, na.rm=TRUE)`	`x2 = np.array([1, np.nan, 3.0]) np.isnan(x2) np.nanmean(x2)`

95% CI for a mean

R	Python
`m <- mean(x); s <- sd(x); n <- length(x) se <- s/sqrt(n) m + qt(c(.025,.975), df=n-1)*se`	`m = x.mean(); s = x.std(ddof=1); n = x.size se = s/np.sqrt(n) m + stats.t.ppf([0.025, 0.975], df=n-1)*se`

t-test (one-sample)

R	Python

`t.test(x, mu=0)`	`stats.ttest_1samp(x, popmean=0)`

t-test (two-sample, Welch)

R	Python
`t.test(x ~ g, data=df) # Welch by default`	`a = df.loc[df.g=='A', 'x'] b = df.loc[df.g=='B', 'x'] stats.ttest_ind(a, b, equal_var=False)`

t-test (paired)

R	Python
`t.test(x, y, paired=TRUE)`	`stats.ttest_rel(x, y)`

Nonparametric: Wilcoxon / Mann–Whitney

R	Python
`wilcox.test(x, mu=0) # one-sample signed-rank wilcox.test(x ~ g, data=df) # Mann–Whitney U`	`stats.wilcoxon(x) # one-sample signed-rank vs 0 stats.mannwhitneyu(a, b, alternative='two-sided')`

Correlation (Pearson / Spearman)

R	Python
`cor(x, y, method='pearson') cor.test(x, y, method='spearman')`	`stats.pearsonr(x, y) stats.spearmanr(x, y)`

Chi-square test (contingency table)

R	Python
`tbl <- table(g, df$z) chisq.test(tbl)`	`tbl = pd.crosstab(df['g'], df['z']) stats.chi2_contingency(tbl)`

Normality test (Shapiro–Wilk)

R	Python

`shapiro.test(x)`	`stats.shapiro(x)`

Equal variances (Levene)

R	Python
`library(car) leveneTest(x ~ g, data=df)`	`stats.levene(a, b, center='median')`

Linear regression (OLS)

R	Python
`fit <- lm(y ~ x + g, data=df) summary(fit) coef(fit) confint(fit)`	`fit = smf.ols('y ~ x + C(g)', data=df).fit() fit.summary() fit.params fit.conf_int()`

ANOVA

R	Python
`fit <- aov(y ~ g, data=df) summary(fit)`	`fit = smf.ols('y ~ C(g)', data=df).fit() sm.stats.anova_lm(fit, typ=2)`

Logistic regression

R	Python
`fit <- glm(z ~ x + g, data=df, family=binomial()) summary(fit) predict(fit, type='response')[1:5]`	`fit = smf.logit('z ~ x + C(g)', data=df).fit(disp=False) fit.summary() fit.predict(df).head()`