#df stands for dataframe

Package installation

install.packages("tidyverse")

library(“tidyverse”)

Data creation

df <- read_csv("quarter_earnings.csv") to read the csv file

merge(df1,df2) to merge two data frames by common columns or row names

merge(df1,df2, by =”specific_column”) to merge df1 on df2 using a shared coloumn name

x <- (1,2,3) assignment operator for creating vectors, variables etc

Data examinations

str(df) display the internal structure of df

head(df) return the first elements in df

glimps(df)

Data wrangling

name_df$variable_x <- as.factor(name_df$variable_x) create new variable name and new labels

name_df$variable_x <- as.numeric(name_df$variable_x)

df$variable_old <- factor(df $New, levels = c("Level_1","Level_2"), labels = c("Level_X"," Level_Y") assign new names to values of a variable

Select_Group <- subset(df, Group == "children") filtering a specific value

df %>% group_by(variable_to_group_by) %>% mean(variable) use pipeline and groupby to get a mean of the certain elebemt in variable of interest

Mathematical calculations

max(x), min(x) maximum and minimum of elements of x

range(x) range of the elements of x

sum(x) sum of the elements of x

prod(x) product of the elements of x

cumsum(x) returns a vector of same length as x with the cumulative sum of the elements of x

mean(x) mean of element of x

median(x) median of elements of x weighted

mean(x, w) mean of x with weights w

sd(x) standard deviation of x

cor(x) correlation matrix of x (matrix or data frame)

round(x) rounds the elements of x

round(x, n) to round elements of x to n decimals

Statistical analysis

cor.test(x, y) correlation test; default method is pearson, use method = "spearman" to specify spearman rank correlation, can also use "kendall

t.test(y, mu=0) one-sample t-test with null hypothesis that mean is 0

t.test(y ~ x, data=df) independent-samples t-test where y is the response and x is the grouping variable

t.test(y1, y2) independent-samples t-test to compare the means of y1 and y2; use paired = TRUE for a paired samples t-test

aov(y ~ A, data = df) one-way ANOVA

aov(y ~ A + x, data = df) ANCOVA for factor A and covariate x

aov(y ~ A + B + A:B, data = df) full two-way ANOVA; can also use A*B in formula to specify both main effects and interaction

aov(y ~ A*B + Error(Subject/(A*B)), data = df) two-way within-subject ANOVA

aov(y ~ W*B + Error(Subject/W), data = df) mixed ANOVA for within-subject factor W and between-subject factor B

lm(y ~ x1 + x2 + x3, data=df) basic multiple linear regression

glm(y ~ x, data=mydata, family="binomial") basic logistic regression

glm(cbind(Y,N) ~ x, data=mydata, family="binomial") logistic regression for binary variable

lmer(y ~ x+(1|Item)+(1|Subject),data=df) multi-level regression with random effects of Item and Subject

lmer(y ~ x + (x | Subject), data=df) for multilevel regression with random effect of Subject on slope

model_name <- lmer(y ~ x + (x | Subject), data=df) to assign a name to model you make

summary(model_name) to print model summary, coefficients, parameter estimates.

Data visualization

ggplot(data, aes(x,y)) set up a plot of data with x on the horizontal and y on the vertical

ggplot(data, aes(x,y,color=z)) to specify mappings for color, shape, linetype, size, etc. use color= , shape= , linetype= , size=

ggplot(data, aes(x,y)) + geom_boxplot() to create a box plot or use geom_line(), geom_point(), geom_bar(), geom_errorbar(), geom_pointrange(), geom_tile()

ggplot(data, aes(x,y)) + stat_summary(fun.y="mean", geom="line") stat_summary summarize y values (mean, median, etc.) at every unique x; geom specifies the resulting plot type (line, point, pointrange, etc.)

ggplot(data, aes(x,y)) + stat_summary(fun.y="mean", geom="line") + facet_wrap(~z) create panels using the subsetting factor, z

ggplot(data, aes(x,y)) + stat_summary(fun.y="mean", geom="line") + facet_grid(d~z) create a grid of panels with subsets of the dataset in different panels

ggplot(data, aes(x,y)) + stat_summary(fun.y="mean", geom="line") + facet_grid(d~z) + labs(x= "x_axis label", y= "y-axis label) set labels for x and y axes; can also be used to set labels for other aesthetics

ggplot(data, aes(x,y)) + stat_summary(fun.y="mean", geom="line") + facet_grid(d~z) + labs(x= "x_axis label", y= "y-axis label) + scale_color_manual(values=c("black", "red")) useful for overriding default scales such as color of lines

ggplot(data, aes(x,y)) + stat_summary(fun.y="mean", geom="line") + facet_grid(d~z) + labs(x= "x_axis label", y= "y-axis label) + scale_color_manual(values=c("black", "red")) + annotate(geom = "text", x = 600, y = 0.8, label = "Your text", color = "black", angle = 90, size = 2.5) useful for creating annotations on the plot

plotname <- ggplot(…) to assign a name to the plot

ggsave("filename.png", plotname, width=6, height=6, dpi=300) to download/save the plot with any extension and resolution