Exploratory data analysis with R

Exploratory Data Analysis

- Exploratory Data Analysis (EDA) is a method used to analyze and summarize the main characteristics of a dataset, often employing statistical graphics and data visualization techniques.

- It's a crucial preliminary step before more formal data analysis or modeling, helping to understand the data, identify patterns, and formulate hypotheses.

What EDA covers

- Data management (Data wrangling…)
- Descriptive analytics (Central tendency, dispersion)
- Diagnostic analytics (correlation….)
- Hypothesis testing
- Visualization (bar graphs, box plots, scatter plots…)

Operators in R

- R has many operators to carry out different mathematical and logical operations.

Type of operators in R

Arithmetic operators
Relational operators
Logical operators
Assignment operators

Arithmetic operators

- These operators are used to carry out mathematical operations like addition and multiplication.

- Examples of arithmetic operators available in R are in the table.

Arithmetic Operators in R	
Operator	Description
+	Addition
-	Subtraction
*	Multiplication
/	Division
^	Exponent
%%	Modulus (Remainder from division)
%/%	Integer Division

R Relational Operators

- Relational operators are used to compare between values. Here is a list of relational operators available in R.

Relational Operators in R	
Operator	Description
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
==	Equal to
!=	Not equal to

R Logical Operators

- Logical operators are used to carry out Boolean operations like AND, OR etc.

- & represents AND

- | represents OR

R Assignment Operators

- These operators are used to assign values to variables.

- The operators <- and = can be used, almost interchangeably, to assign to variable in the same environment.

The pipe operator in R

- The pipe operator **(%>%) or (|>)** is a binary operator that forwards the value of its left-hand side to the first argument of the function on its right-hand side.

- This enables a natural and readable way of chaining multiple functions together, reducing the need for intermediate variables and improving code clarity.

- %>% is a function in dplyr package

- |> is a function in base R.

Pipe operator usage

```r
# Load required library
library(dplyr)
# Create a data frame
data <- data.frame(x = 1:10, y = rnorm(10))
# Filter rows, mutate column, and summarize data using pipe operator
summary <- data %>%
  filter(x > 5) %>%
  mutate(z = x + y) %>%
  summarize(mean_z = mean(z))
# Print summary
print(summary)
```

Part 1

Data wrangling with R

Filtering rows in data sets

filter() selects the rows of a data frame that meet a column criteria.

Example: Create a new dataset containing the maize crop only

```
maize <- filter(crop_recommendation, Crop == "maize")
```

Example2: Create a new dataset containing humidity values above 50

```
hum_above50 <- filter(crop_recommendation, humidity > 50)
```

Example3: Create a new dataset containing humidity values above 50 for maize crop

```
hum_maize_above50 <- filter(crop_recommendation, humidity > 50 & Crop == "maize")
```

	Crop	Nitrogen	Phophorous	Patassium	temperature
1	maize	75.0	32.3	130.6	25.7
2	maize	115.0	35.4	87.4	18.5
3	maize	121.1	40.1	134.6	25.0
4	maize	73.5	36.0	112.4	22.1
5	maize	109.4	26.1	80.3	18.6
6	maize	81.5	25.7	135.6	23.4
7	maize	93.6	27.4	125.3	23.2
8	maize	81.0	54.4	107.5	27.4
9	maize	70.9	49.8	131.4	22.5
10	maize	126.8	34.0	114.7	26.4
11	maize	120.2	54.7	113.4	28.1
12	maize	112.1	44.4	119.1	21.8

Select columns

select() is used to select columns that you want to **retain**.

Example: Select Crop and yield_category then store them.

```
data <- select(crop_recommendation, Crop, yield_category )
```

Example Using pipe operator

```
data1 <- crop_recommendation  %>%
          select(Crop, yield_category )
```

#deselect/remove columns from the data set

```
data2 <- select(crop_recommendation, -Crop, -yield_category )
```

Filter

	Crop	yield_category
1	maize	High
2	cotton	High
3	rice	High
4	coffee	Low
5	cotton	High
6	coffee	High
7	chickpea	Low
8	maize	High
9	cotton	High
10	wheat	Low
11	chickpea	High
12	maize	Low

Combining functions to select and filter

Example Using pipe operator

```
data3 <- crop_recommendation  %>%
          select( Crop, yield_category ) %>%
           filter(Crop == "rice ")
```

Filter

	Crop	yield_category
1	rice	High
2	rice	High
3	rice	Low
4	rice	Low
5	rice	Low
6	rice	High
7	rice	High
8	rice	High
9	rice	Low
10	rice	Low
11	rice	Low
12	rice	Low

arrange

arrange() orders the rows of a data frame by the values of selected columns.

arrange() sorts values in ascending order or descending order.

Example: Sort the yield in ascending order

```
data4 <- arrange(crop_recommendation, yield_kg_ha)
```

Sort yield in descending order

```
data5 <- arrange(crop_recommendation, -yield_kg_ha)
```

idity	pH	rainfall	yield_kg_ha	yi
71.7	5.61	193.1	1347	Lo
89.3	6.31	242.6	1433	Lo
60.7	6.04	180.3	1433	Lo
88.5	5.93	131.8	1433	Lo
87.4	6.01	201.4	1451	Lo
60.1	5.35	208.5	1467	Lo
70.3	5.34	243.2	1470	Lo
40.0	6.86	102.4	1525	Lo
50.3	6.06	72.1	1542	Lo
72.5	5.42	223.3	1544	Lo
33.0	7.01	110.5	1549	Lo
87.0	6.55	211.6	1575	Lo

idity	pH	rainfall	yield_kg_ha	y
72.7	6.83	200.1	84071	H
85.2	6.21	243.9	83453	H
78.3	6.38	220.1	82620	H
68.5	7.01	265.7	81222	H
66.2	6.63	148.2	80924	H
73.7	6.36	223.2	80729	H
87.7	7.26	276.5	80657	H
69.6	7.12	248.2	80031	H
86.8	6.97	147.5	79779	H
82.6	6.76	240.0	79705	H
69.7	6.82	198.4	79257	H
60.7	6.60	260.0	79225	H

mutate

mutate() creates new variables in the data set

It adds the new variable to existing data

#Example: creates a new variable "new_yield" by getting half the yield_kg_ha variable

```
half_yield <- mutate(crop_recommendation , new_yield = yield_kg_ha/2)
```

#Alternative

```
half_yield <- crop_recommendation %>%
              mutate(new_yield = yield_kg_ha/2)
```

rainfall	yield_kg_ha	yield_category	new_yield
171.5	6062	High	3031.0
63.6	1789	High	894.5
156.6	5697	High	2848.5
126.4	1579	Low	789.5
92.4	1930	High	965.0
144.4	1675	High	837.5
67.4	1726	Low	863.0
198.3	6558	High	3279.0
71.6	1750	High	875.0
128.8	4465	Low	2232.5
86.0	1780	High	890.0
139.9	6004	Low	3002.0

rename

rename() creates new names for variables in the data set

#Example: rename Nitrogen to N

nit <- rename(crop_recommendation, N = Nitrogen)

	Crop	Nitrogen	Phophorous
1	maize	75.0	32.
2	cotton	78.9	48.
3	rice	101.3	58.
4	coffee	108.8	26.
5	cotton	70.5	32.
6	coffee	98.7	34.
7	chickpea	43.6	26.
8	maize	115.0	35.
9	cotton	50.7	43.
0	wheat	97.0	25.
1	chickpea	48.7	39.
2	maize	121.1	40.

	Crop	N	Phophorou
1	maize	75.0	32.3
2	cotton	78.9	48.8
3	rice	101.3	58.8
4	coffee	108.8	26.6
5	cotton	70.5	32.2
6	coffee	98.7	34.9
7	chickpea	43.6	26.9
8	maize	115.0	35.4
9	cotton	50.7	43.9
0	wheat	97.0	25.1
1	chickpea	48.7	39.2
2	maize	121.1	40.1

relocate

relocate() gives a new order to column names

Example: relocate the rainfall and yield_category from their original position

```
New_order <- relocate(crop_recommendation, rainfall, yield_category)
```

	Crop	Nitrogen	Phosphorous	Potassium
1	maize	75.0	32.3	130.6
2	cotton	78.9	48.8	105.7
3	rice	101.3	58.8	142.5
4	coffee	108.8	26.6	100.9
5	cotton	70.5	32.2	90.8
6	coffee	98.7	34.9	102.2
7	chickpea	43.6	26.9	43.7
8	maize	115.0	35.4	87.4
9	cotton	50.7	43.9	96.4
10	wheat	97.0	25.1	93.9
11	chickpea	48.7	39.2	79.9
12	maize	121.1	40.1	134.6

	rainfall	yield_category	Crop	Nitrogen
1	171.5	High	maize	75.0
2	63.6	High	cotton	78.9
3	156.6	High	rice	101.3
4	126.4	Low	coffee	108.8
5	92.4	High	cotton	70.5
6	144.4	High	coffee	98.7
7	67.4	Low	chickpea	43.6
8	198.3	High	maize	115.0
9	71.6	High	cotton	50.7
10	128.8	Low	wheat	97.0
11	86.0	High	chickpea	48.7
12	139.9	Low	maize	121.1

Part 2

Descriptive analytics

Data summary and missingness

- Data summary gives a comprehensive understanding of large datasets and variables.
- Categorical data can be summarized using frequency distribution tables.
- Frequency distribution tables are obtained using the table function: table()
- Missingness can be detected using the table function: table()
- Descriptive summaries are obtained using the summary function: summary()

❖Demonstration in RStudio

Outlier detection

- Outliers are values that are considered out of range in a given numerical variable.

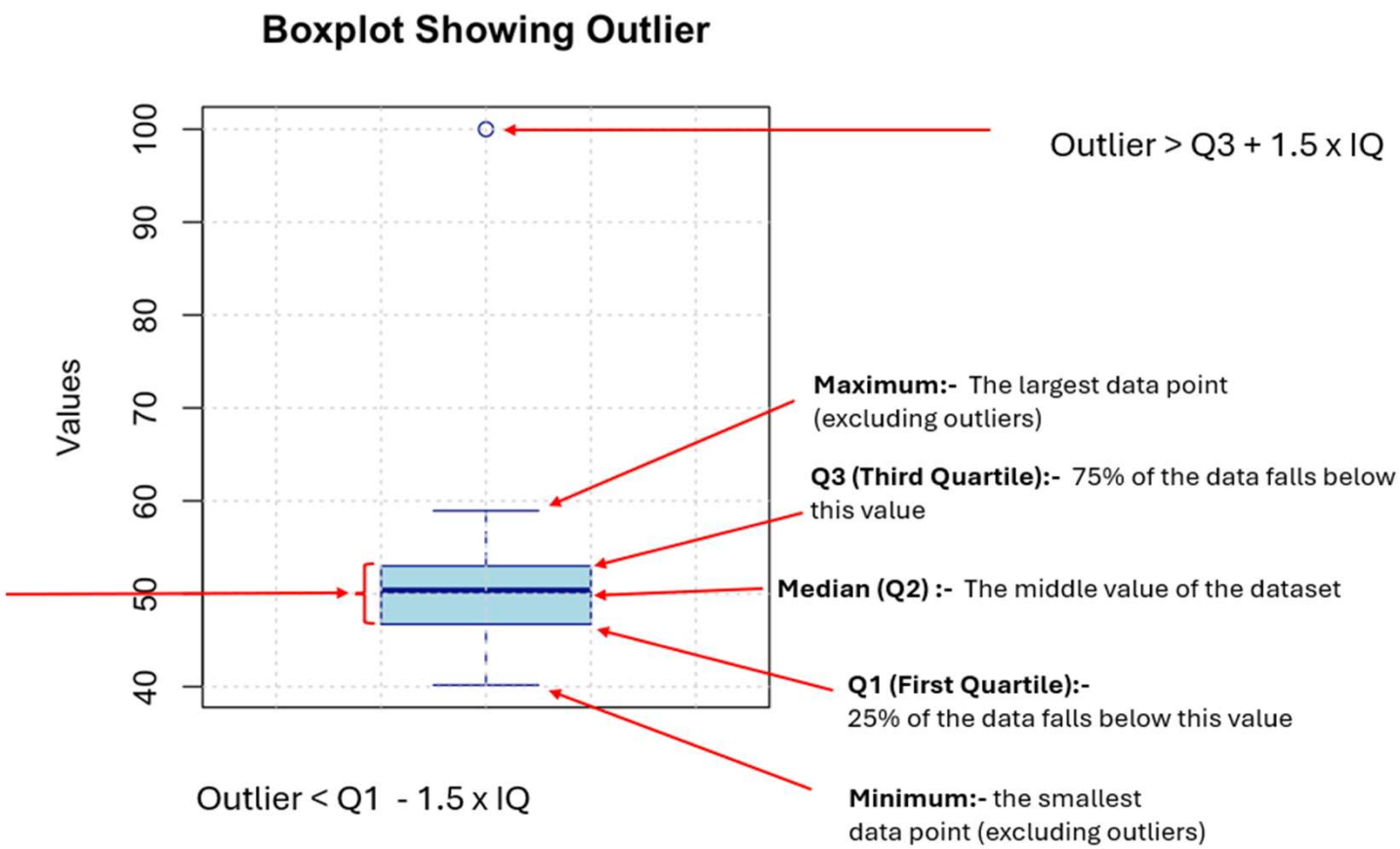
- Outliers affect data modelling and can cause drastic changes in results.

- Outliers can be quickly detected visually using box plots.

Box plots

- Box plots are figures that contain important information about the summary of numeric variable.
- The box represents the interquartile range (IQR), from Q1 (25th percentile) to Q3 (75th percentile).
- The line inside the box is the median.
- Whiskers extend up to 1.5 × IQR from Q1 and Q3.
- Points beyond the whiskers are considered outliers.
- Minimum:- the smallest data point (excluding outliers)
- Maximum:- The largest data point (excluding outliers)

Box plots showing outlier



The **box** represents the interquartile range (IQR), from Q1 (25th percentile) to Q3 (75th percentile).

Distribution of data

- Understanding of distribution of numeric data is key.
- It involves testing for normality of key variables
- Distributions can be assessed in two ways:

➢ Visually using graphs such as the histogram

➢ Performing a statistical test of normality such as Shapiro's wilk test.

Data visualization with ggplot2

What is ggplot2?

- ggplot2 is part of the collection of packages within tidyverse.

  - It is used for visualization of data in R

  - "gg" stands for grammar of graphics

```
#load the library of ggplot2 to access its functions
library(ggplot2)
```

How ggplot2 works….

- ggplot2 works by adding different layers of information to a graph.
- Layers are added to the graph using the plus sign (+).
- Different layers perform different functions within ggplot2 package.
- Layers can be optionally added onto the graph.

Syntax of ggplot2

```
ggplot(dataframe, aes(x=variable, y=variable) +
geom_object() +
labs(title= "title of graph", subtitle = "subtitle of graph", x= "xlabel",y= "ylabel")+
coord_cartesian(xlim=c(a,b),ylim=c(a,b))
```

- **Importance of different functions above:**
  - ggplot() function specifies the data frame.
  - aes() specifies the variables to be plotted,color etc. It is the aestetics function.
  - geom() function specifies the type of graph to be plotted.
  - labs() function specifies the title of the graph and axis labels.
  - coord_cartesian() specifies the limits on the axes of a graph.
- **There are many other layers that can be added to graphs.**

Types of graphs

- Graphs are specified using the geom() function.
- You can have more than one type of graph on the same visualization.

Examples of graphs created by the geometric objects:

geom_point() : to draw points on a graph e.g scatter plots.

geom_smooth(): to draw smooth lines on a graph.

geom_histogram(): to draw a histogram on a graph

geom_line(): to draw a line graph

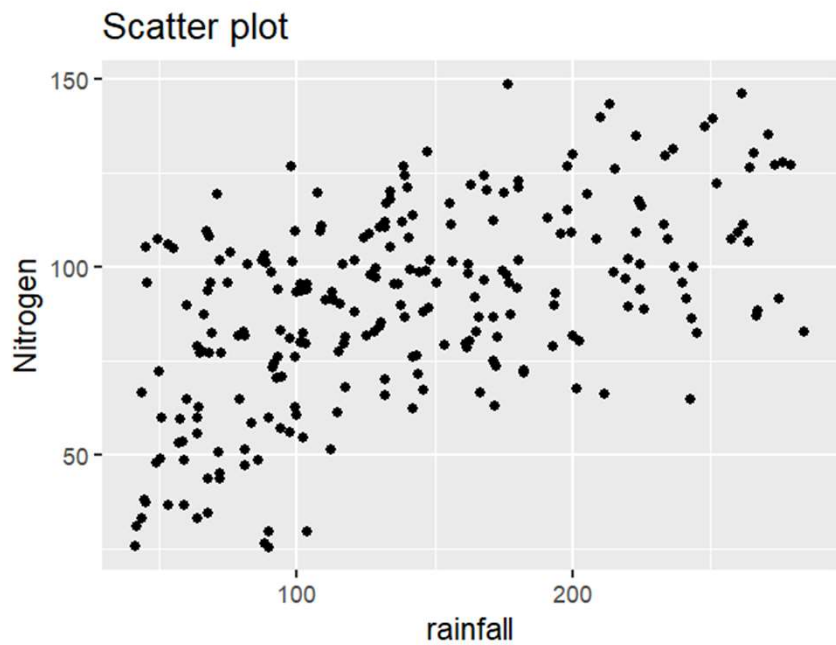
geom_bar(): to draw a bar graphs

geom_boxplot() :to draw a boxplot on a graph

Scatter plot

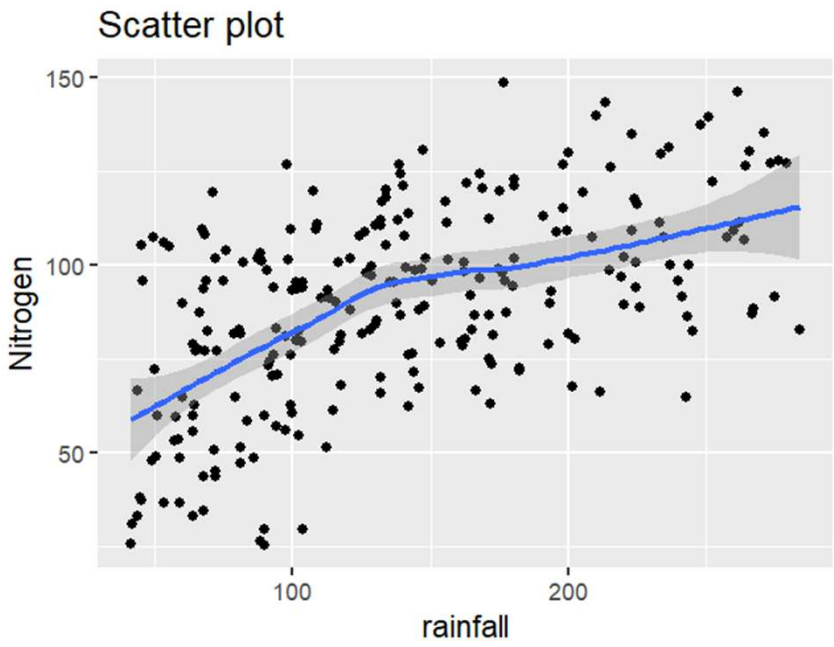
A scatterplot is a visualization of two continuous/quantitative variables.

```
#draw a scatter plot
ggplot(crop_recommendation, aes(x=rainfall, y=Nitrogen)) +
geom_point() +
labs(title= "Scatter plot", x= "rainfall", y= "Nitrogen")
```



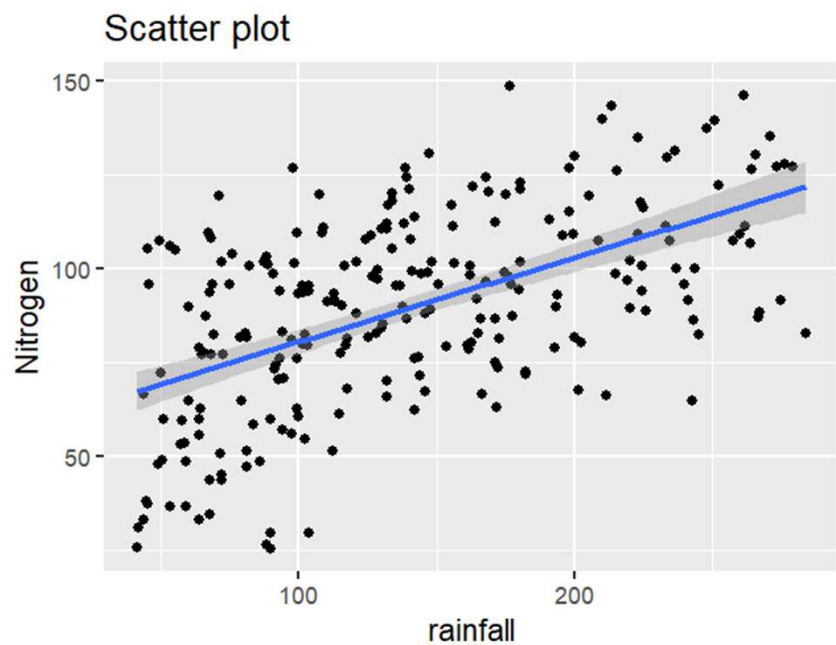
Add a smooth line to the scatter plot.

```
ggplot(crop_recommendation, aes(x=rainfall, y=Nitrogen)) +
geom_point() +
geom_smooth() +
labs(title= "Scatter plot", x= "rainfall", y= "Nitrogen")
```



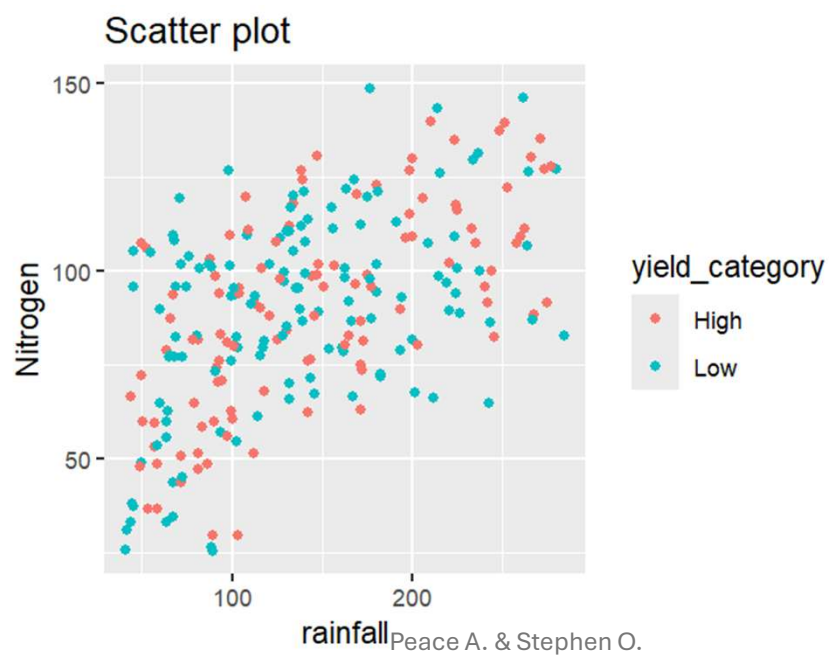
Add a straight line to the scatter plot.

```
ggplot(crop_recommendation, aes(x=rainfall, y=Nitrogen)) +
geom_point() +
geom_smooth(method="lm")  +
labs(title= "Scatter plot", x= "rainfall", y= "Nitrogen")
```



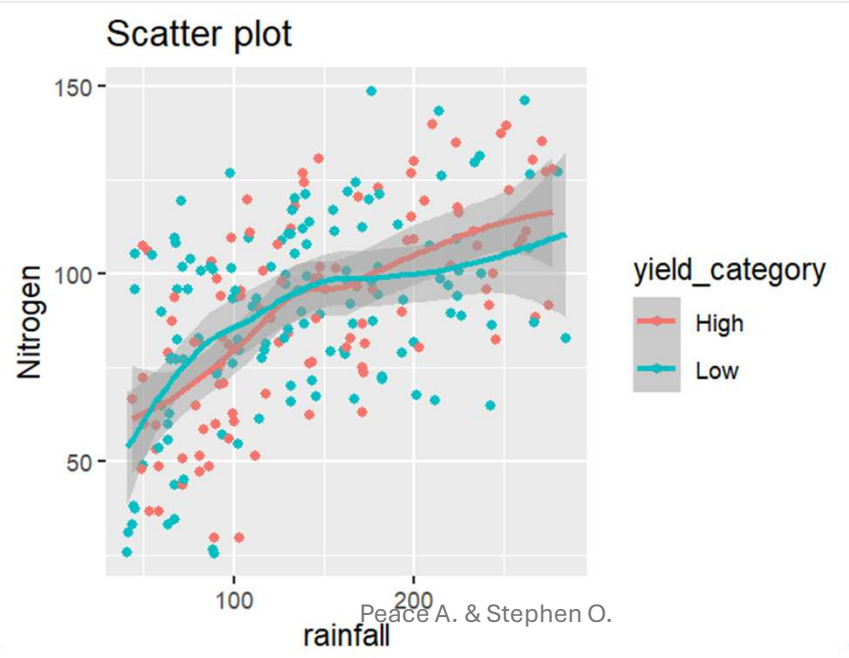
Add color to the scatter plot

```
ggplot(crop_recommendation, aes(x=rainfall, y=Nitrogen, color = yield_category)) +
geom_point() +
labs(title= "Scatter plot", x= "rainfall", y= "Nitrogen")
```



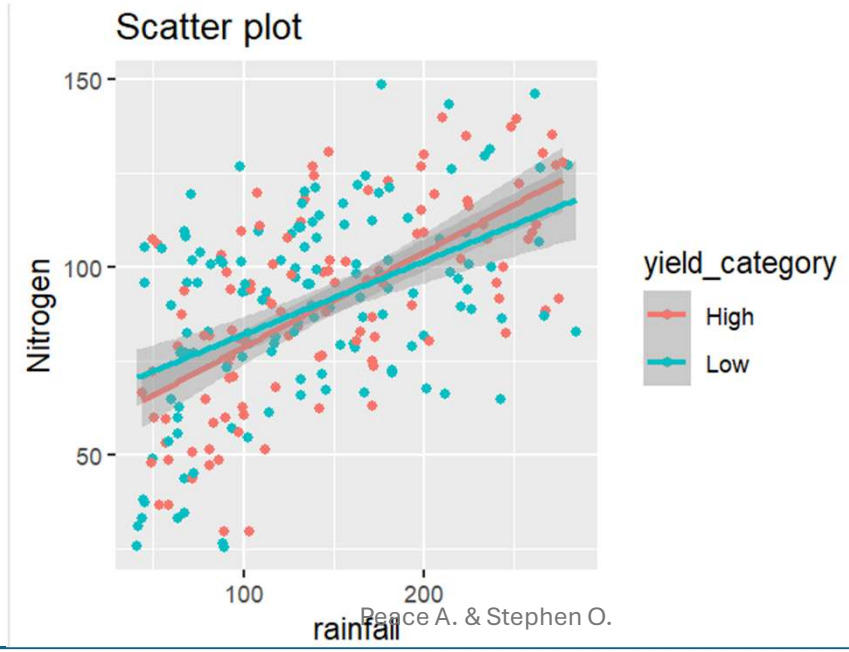
Add smooth line per yield category

```
ggplot(crop_recommendation, aes(x=rainfall, y=Nitrogen, color = yield_category)) +
geom_point() +
geom_smooth() +
labs(title= "Scatter plot", x= "rainfall", y= "Nitrogen")
```



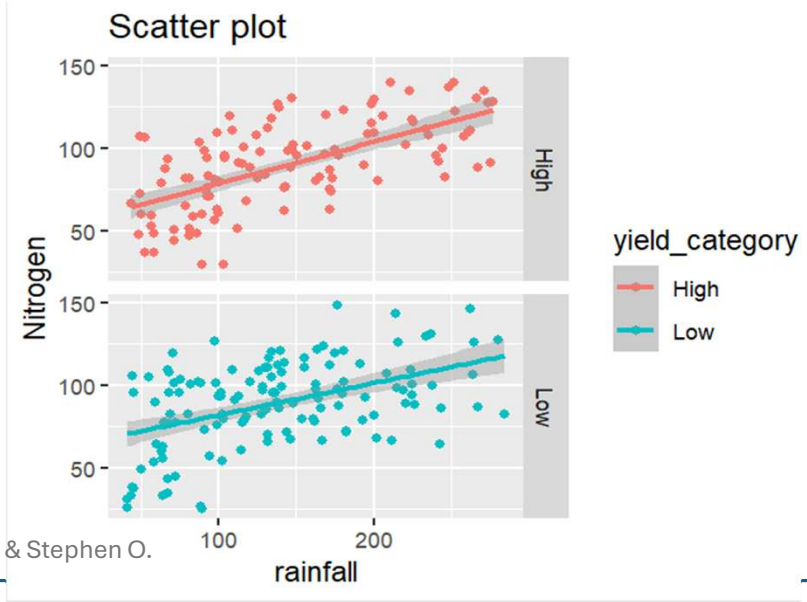
Add straight line per yield category

```
ggplot(crop_recommendation, aes(x=rainfall, y=Nitrogen, color = yield_category)) +
geom_point() +
geom_smooth(method="lm") +
labs(title= "Scatter plot", x= "rainfall", y= "Nitrogen")
```



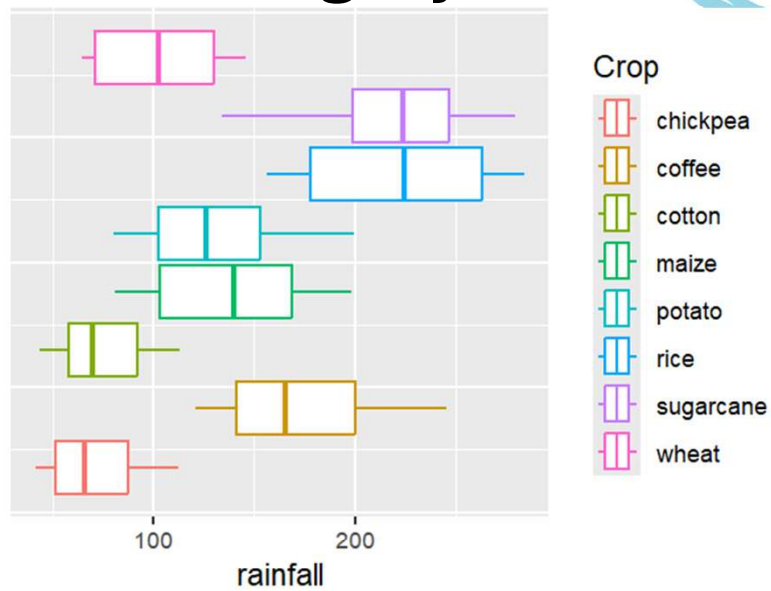
Add facets per yield category

```
ggplot(crop_recommendation, aes(x=rainfall, y=Nitrogen, color = yield_category)) +
geom_point() +
geom_smooth(method="lm") +
labs(title= "Scatter plot", x= "rainfall", y= "Nitrogen") +
facet_grid(rows = vars(yield_category))
```

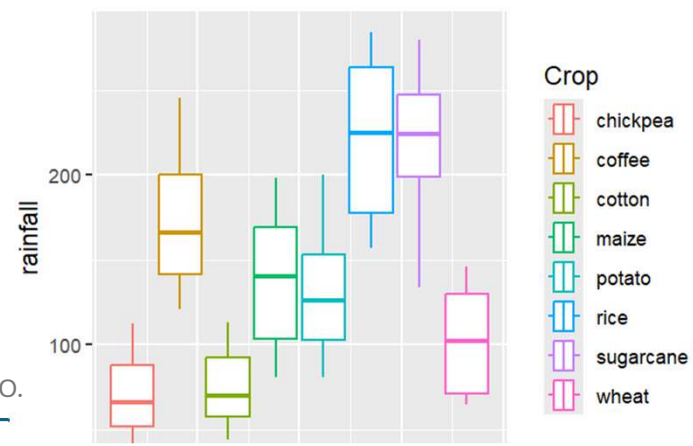


Box plot of Nitrogen per yield category

```
ggplot(crop_recommendation, aes(x=rainfall, color = Crop))+
geom_boxplot() +
 labs(title= "Amount of rainfall")
```

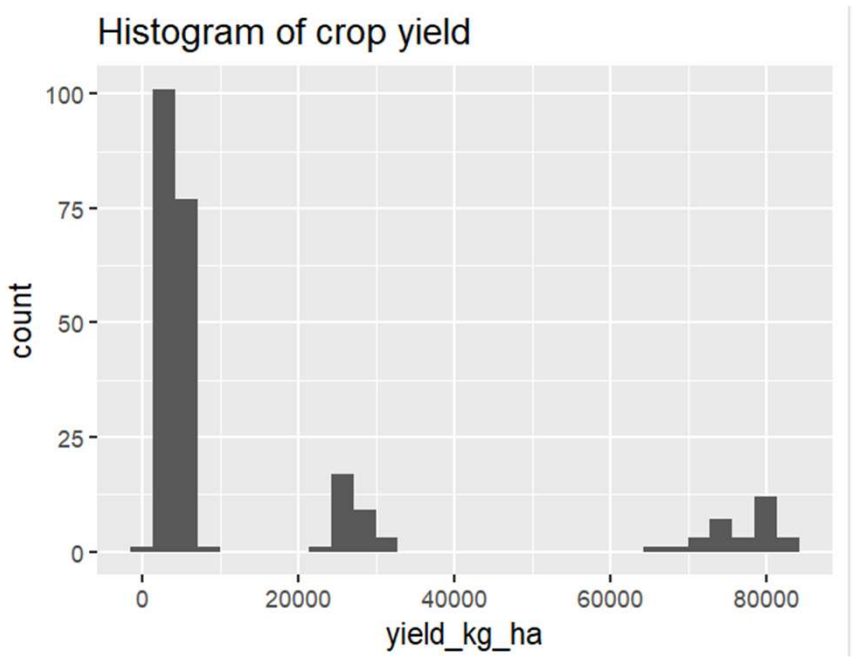



```
ggplot(crop_recommendation, aes(x=rainfall,  color = Crop))+
geom_boxplot() +
labs(title= "Amount of rainfall")  +
coord_flip()
```



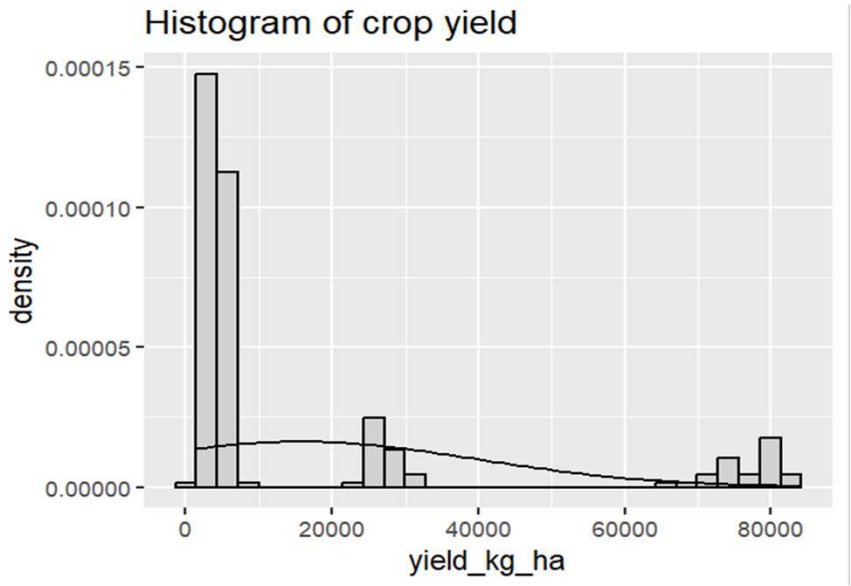
Histogram

```
ggplot(crop_recommendation, aes(x=yield_kg_ha))+
geom_histogram() +
 labs(title= "Histogram of crop yield")
```



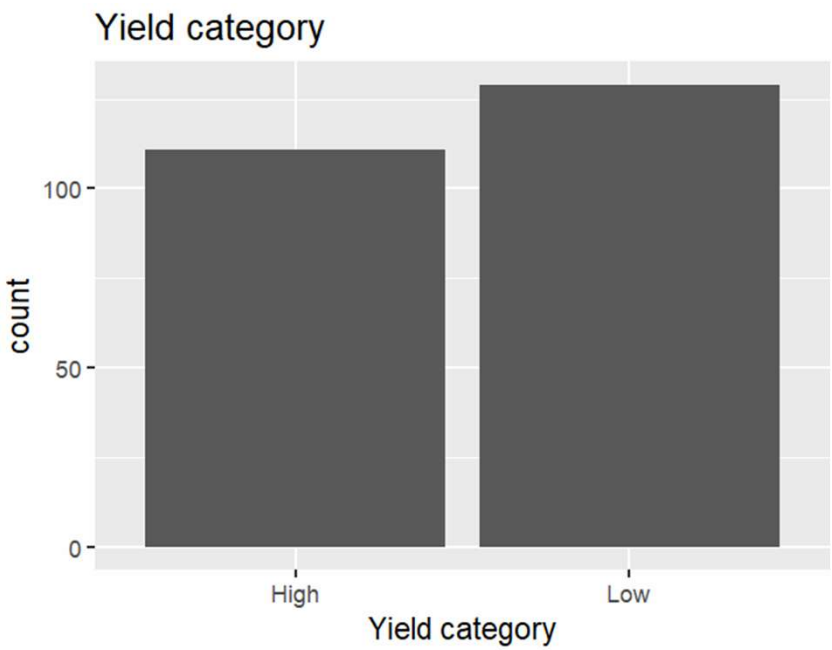
Histogram with normal curve

```
ggplot(crop_recommendation, aes(x=yield_kg_ha))+
geom_histogram(aes(y = ..density..), fill='lightgray', col='black') +
stat_function(fun = dnorm,args = list(mean = mean(crop_recommendation $ yield_kg_ha),sd =
sd(crop_recommendation $ yield_kg_ha)))+
 labs(title= "Histogram of crop yield")
```



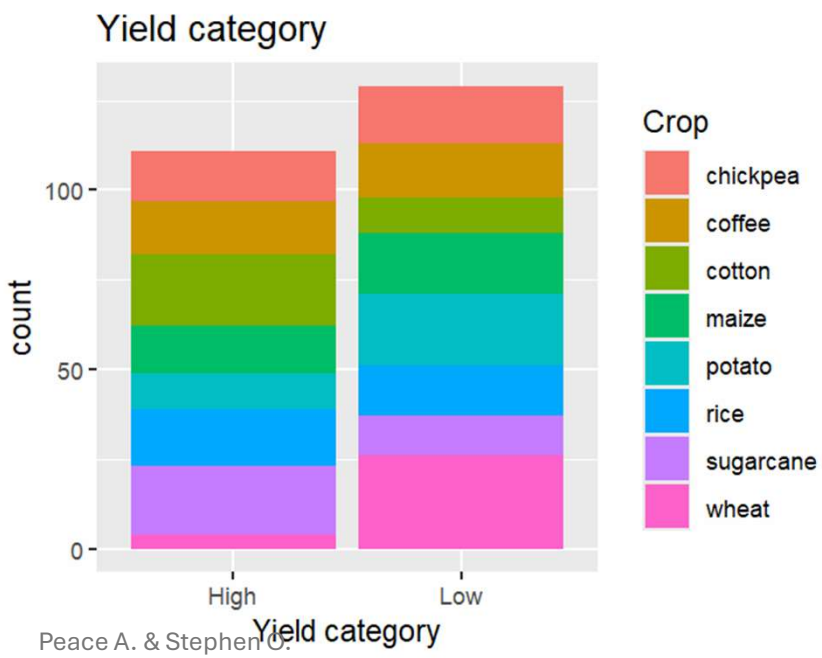
Bar graphs

```
ggplot(crop_recommendation, aes(x=yield_category)) +
geom_bar()+
labs(title =“Yield category", x="Yield category")
```



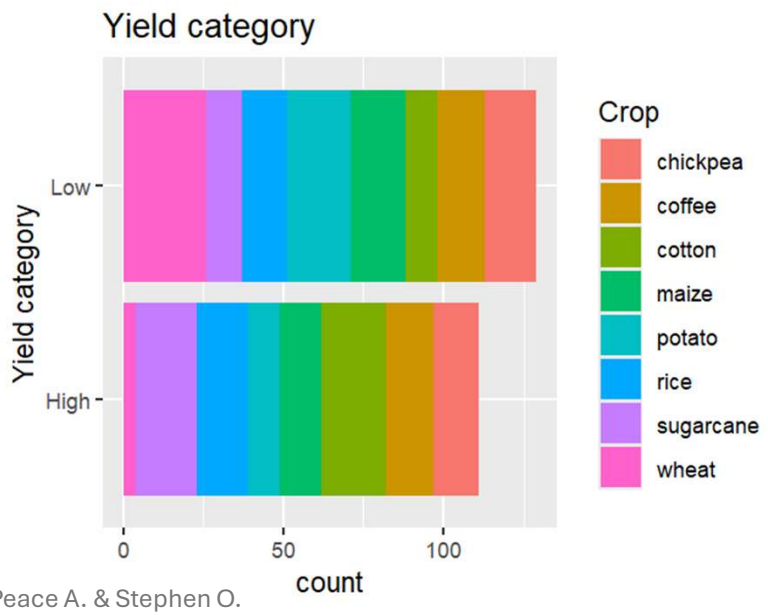
Bar graphs

```
ggplot(crop_recommendation, aes(x=yield_category, fill =Crop )) +
geom_bar()+
labs(title ="Yield category", x="Yield category")
```



Horizontal bar graphs

```
ggplot(crop_recommendation, aes(x=yield_category, fill =Crop )) +
geom_bar()+
labs(title ="Yield category", x="Yield category") +
coord_flip()
```



Diagnostic Analytics

Hypothesis testing

T-test

ANOVA

Hypothesis

- Hypothesis is a claim or statement about the value of single population characteristics or values of several population characteristics.

There are two types of hypothesis:

1) Null hypothesis (H_o) :is a claim about a population characteristic that is initially assumed to be true.

2) Alternative hypothesis (H_a) : is the competing claim.

- A test of hypothesis is a method that uses sample data to decide two competing claims (hypothesis) about population characteristics.

Hypothesis testing

- The form of null hypothesis is:

  H_o: population characteristic = hypothesized value.
  average yield of maize = 6000 kg_ha

- The alternative hypothesis can be any of the following three forms:

  - H_a: population characteristic > hypothesized value (is greater than 6000)

  - H_a: population characteristic < hypothesized value (is less than 6000)

  - H_a: population characteristic ≠ hypothesized value (is not equal to 6000)

Hypothesis testing procedure

The critical concepts are these

1. The procedure begins with the assumption that the H_0 is true.

2. The goal is to determine whether there is enough evidence to infer that H_a is true.

3. There are two possible decisions:

   a) Reject the null. i.e Conclude that there is **enough** evidence to support the **alternative** hypothesis.

   b) Fail to reject the null. i.e Conclude that there is **not enough** evidence to support the **alternative** hypothesis.

Decision making

- Small *P*-values provide stronger evidence against the null hypothesis.(H_0) i.e rejection of the null hypothesis.

  if p- value < 0.05 then reject the null hypothesis

- Small *P*-value $\Rightarrow$ strong evidence against H_0

- large *P*-values provide stronger evidence for the null hypothesis.(H_0) i.e acceptance of the null hypothesis.

  if p- value > 0.05 then accept the null hypothesis

Interpretation

Conventions

$P > 0.05 \Rightarrow$ non-significant evidence against H_0

$0.01 < P \leq 0.05 \Rightarrow$ significant evidence against H_0

$P \leq 0.01 \Rightarrow$ highly significant evidence against H_0

Examples

$P = 0.27 \Rightarrow$ non-significant evidence against H_0

$P = 0.01 \Rightarrow$ highly significant evidence against H_0

$P = 0.04 \Rightarrow$ significant evidence against H_0

Summary of the hypothesis

The Steps:

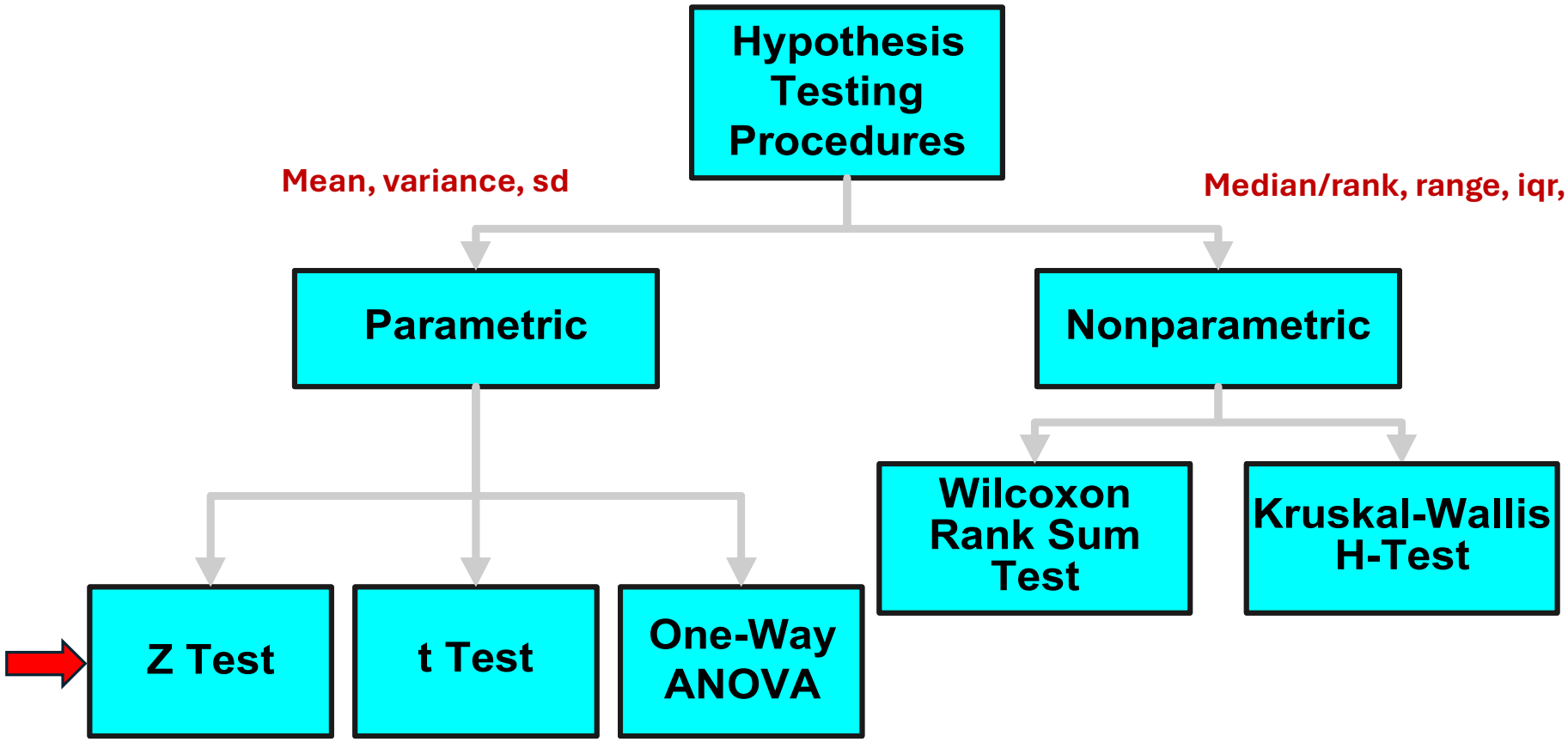
1. Define your hypotheses (null, alternative)
2. Specify your null distribution
3. Collect data from a sample
4. Calculate the p-value of what you observed (done by software)
5. Make a decision i.e Reject or fail to reject (~accept) the null hypothesis

Methods of hypothesis testing

- Parametric methods are those methods that are used when the population is normally distributed or can be approximately equal to normal when the size is large. Examples include mean, standard deviation....etc

- Non-parametric methods are used for analysis when the population is not normally distributed, and we cannot approximate them to normal. Methods include median.....etc

Hypothesis testing methods



Shapiro–Wilk test for normality: RStudio

- A Shapiro-Wilk: Test for normality

The null hypothesis H_o: Normally distributed data.

Alternative hypothesis H_a: Not normally distributed data.

Perform the test in R:

Decision: You accept the null/fail to reject the null when your p-value is **greater** than 0.05.

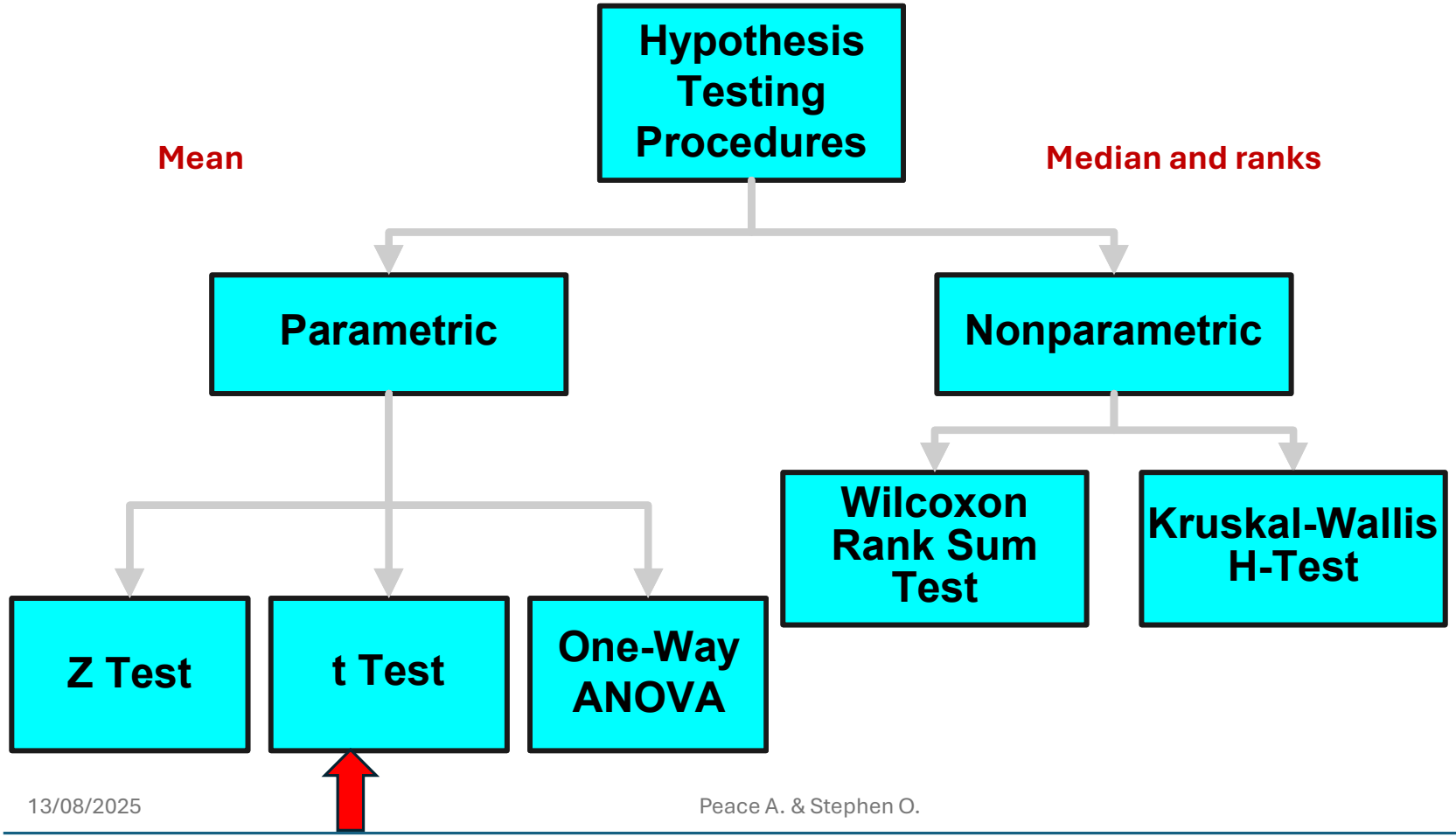
For example, P-value is greater than 0.05, conclude normality.

Shapiro–Wilk test: RStudio

```
> shapiro.test(crop_recommendation$yield_kg_ha)

        Shapiro-Wilk normality test

data:  crop_recommendation$yield_kg_ha
W = 0.59525, p-value < 2.2e-16
```

P-values < 0.05, therefore, reject the null and conclude that yield is not normally distributed.

Hypothesis testing methods



Independent t-test

- The t-test checks whether the mean is equal in two groups.
- There must be a single observation from each participant from ***two independent groups***
- The observation from the second group is independent from the first since they come from different subjects.
- Comparing the **difference between two means** to a **distribution of differences between mean**.

Procedure for the independent t-test

- A t-test: Test for equality of means in two groups

  The null hypothesis H_o: The two means are equal

  Alternative hypothesis H_a: The two means are not equal

  Perform the test in R:

  Decision: You accept the null/fail to reject the null when your p-value is **greater** than 0.05.

  For example, P-value is greater than 0.05, conclude that the two means are equal.

T-Test in R/RStudio

```
        Welch Two Sample t-test

data:  Pesticide_kg by year
t = 0.46684, df = 165.94, p-value = 0.6412
alternative hypothesis: true difference in means between group 2024 and group 2025 is not equal
to 0
95 percent confidence interval:
 -4.166860  6.747599
sample estimates:
mean in group 2024 mean in group 2025
          50.95935           49.66898
```

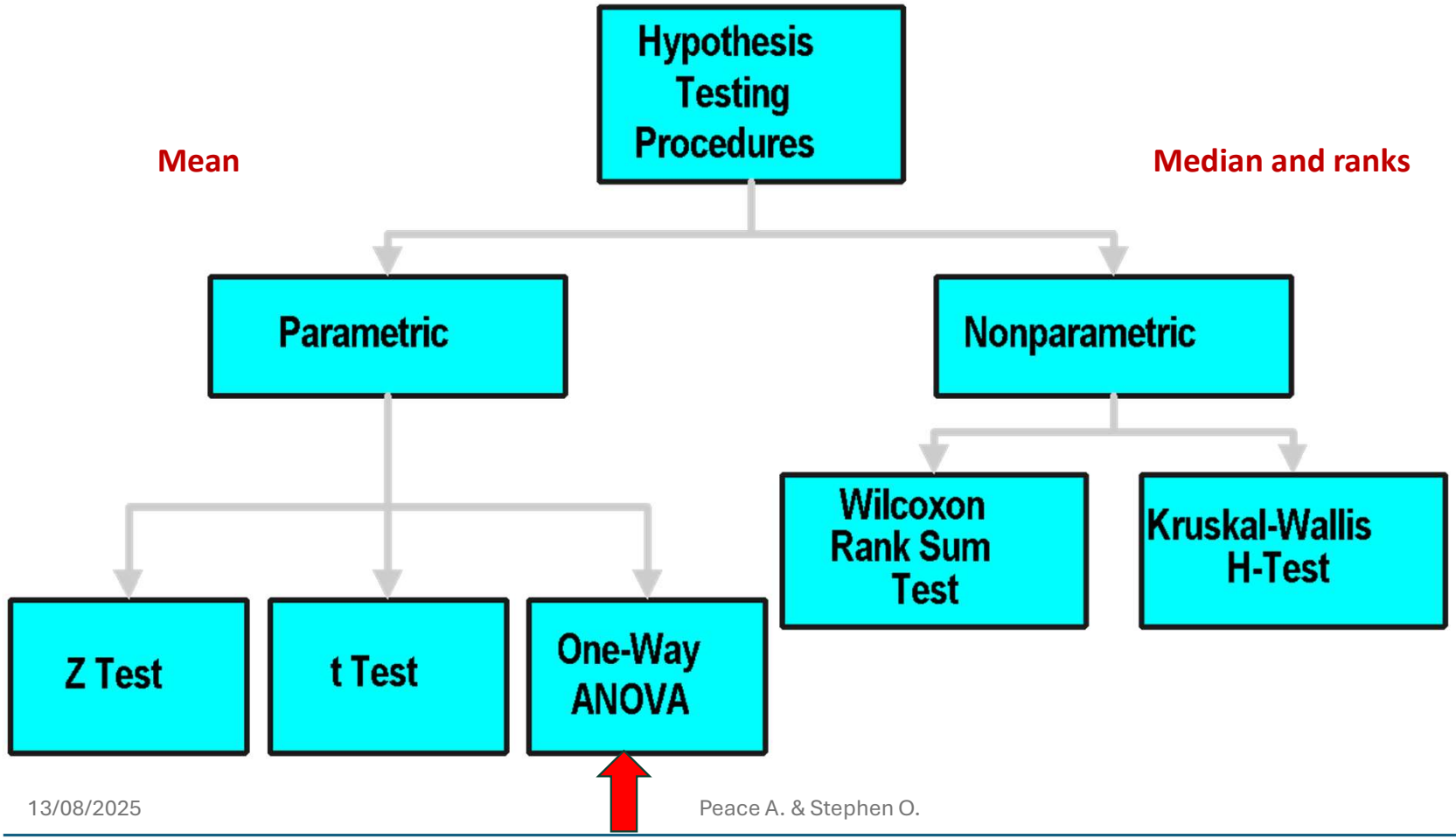
T-statistics

mean Pesticide in 2024

mean Pesticide of 2025

P-value > 0.05, therefore, accept the null and conclude that the means are the same

Hypothesis testing methods



Analysis of Variance (ANOVA)

- ANOVA, or Analysis of Variance, is a test used to determine differences between research results from three or more unrelated samples or groups.

- It checks whether the variation in data is large enough to cause differences in the mean.
- ANOVA is a bivariate test where one variable is numeric and the second one is categorical.
- The categorical variable has three or more groups.

Note: Variance = the square of the standard deviation.

ANOVA model Assumption:

- Independence (the variables are independent)
- Normality (variables are normally distributed)
- Homoscedasticity (the variables have the same variance).

Hypothesis: What Anova does.

At its simplest, ANOVA tests the following hypotheses:

H_0: The means of all the groups are equal

i.e $\mu_1 = \mu_2 = \mu_3 \ldots\ldots = \mu_n$

H_a: The means of all the groups are not equal

i.e $\mu_1 \neq \mu_2 \neq \mu_3 \ldots\ldots \neq \mu_n$

What ANOVA cannot do.

- ANOVA cannot tell which groups are different i.e when the alternative hypothesis has been accepted ($\mu_1 \neq \mu_2 \neq \mu_3 \ldots\ldots \neq \mu_n$)

- If the means are different, then the Post-hoc test of mean differences is required to determine which means are different from each other.

Levels of ANOVA

- One-Way
- Factorial (Two-Way, Three-Way) ANOVA
- Repeated measures ANOVA

One way ANOVA

- Requires two variables:

➢ One factor/categorical variable with three or more categories/groups e.g group

➢ One dependent quantitative variable that is Normally distributed e.g Pesticide_kg

Pesticide_kg	SoilType
63.508326	Loamy
44.314471	Loamy
97.427769	Sandy
47.758094	Clay
51.608282	Loamy
43.484168	Loamy
16.130919	Sandy
44.736932	Clay
54.640739	Loamy
80.018452	Clay
59.839143	Sandy

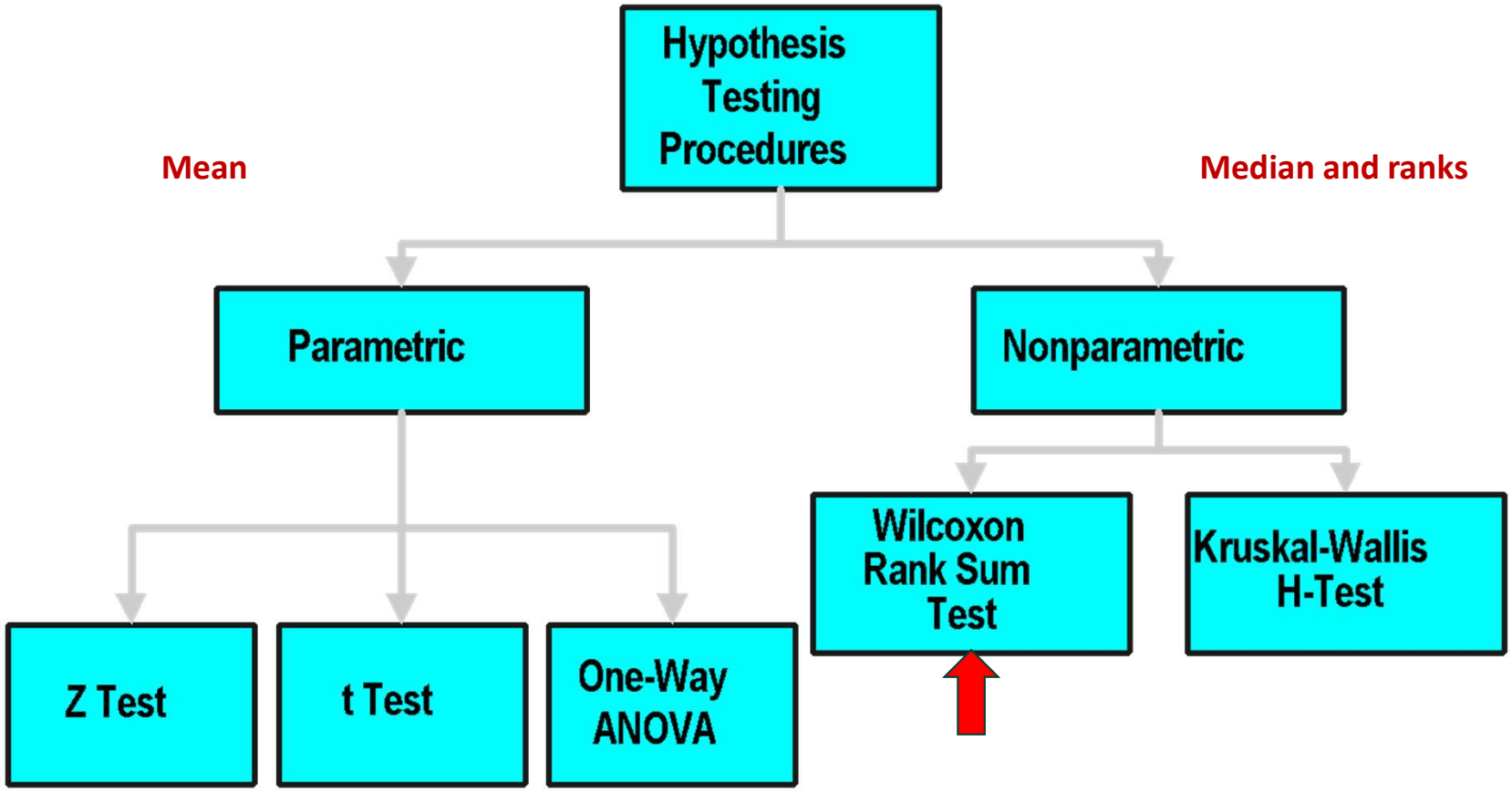
ANOVA output using aov function

```
> ANOVA <- aov(Pesticide_kg~SoilType, data=Kenya_agri_disease_spatial)
> summary(ANOVA)
             Df Sum Sq Mean Sq F value Pr(>F)
SoilType      2   1083   541.4   1.501  0.225
Residuals   197  71061   360.7
>
```

Degrees of freedom = k-1
i.e 3-1 = 2

P-value > 0.05 is significant i.e fail to reject the null hypothesis and conclude that the mean of pesticide is the same in the different soil types.

Hypothesis testing methods



Wilcoxon rank sum test

Wilcoxon rank sum test is the non-parametric test of equality of two medians.

- It is also referred to as Man Whitney's U test.
- Does not require normally distributed populations
- May be applied to ordinal data
- Assumptions
  - Independent samples
  - At least ordinal data

Wilcoxon rank sum test hypothesis

Hypotheses:

Null hypothesis:

The medians of values for each group are equal.

Alternative hypothesis:

The medians of values for each group are not equal.

Wilcoxon rank sum test

P-values < 0.05, therefore, not normally distributed

#check for normality

shapiro.test(crop_recommendation$rainfall)

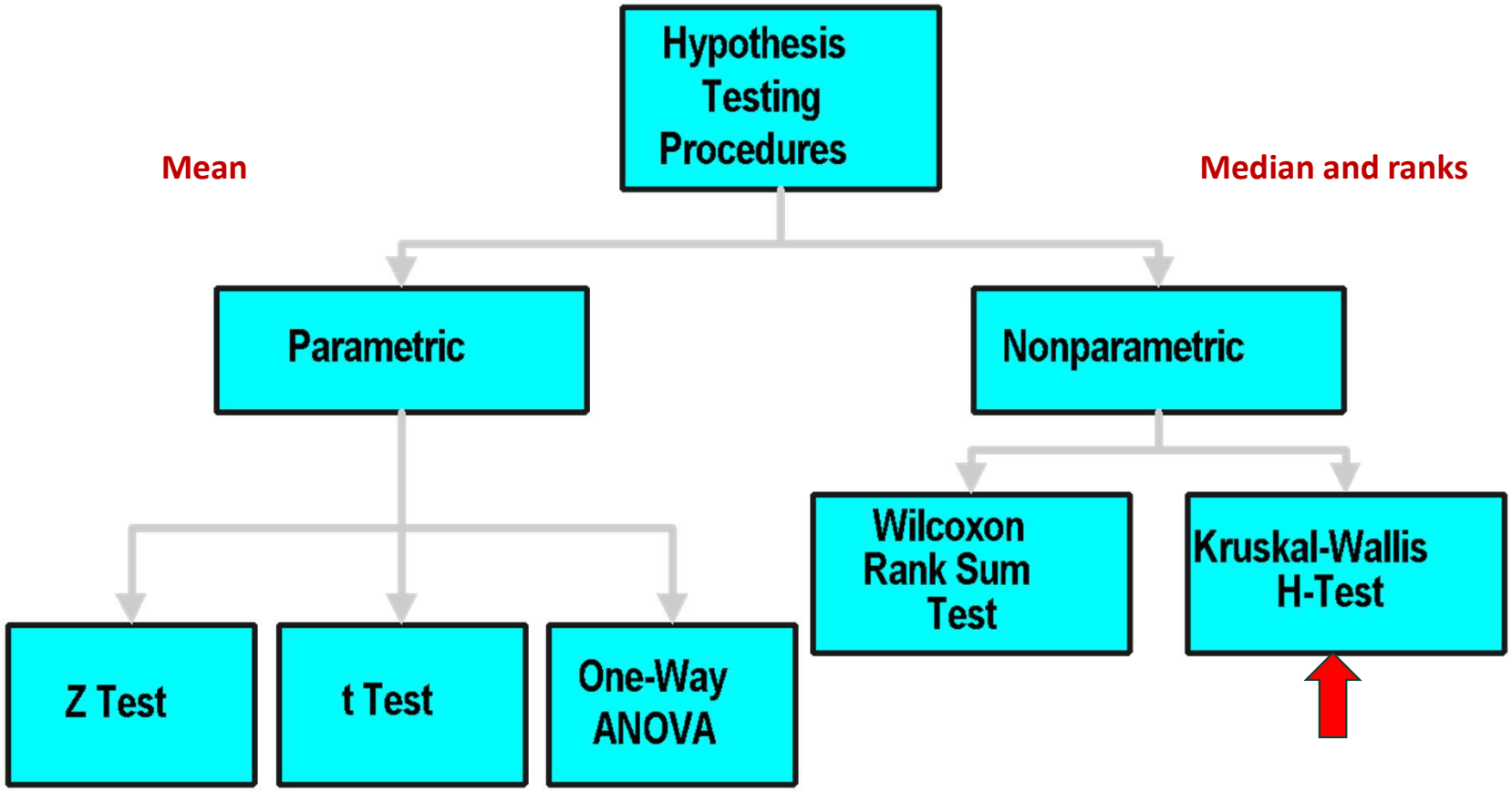
```
> shapiro.test(crop_recommendation$rainfall)

        Shapiro-Wilk normality test

data:  crop_recommendation$rainfall
W = 0.95392, p-value = 6.279e-07

> #perform wilcoxon test
> wilcox.test(rainfall ~ yield_category, data = crop_recommendation)

        Wilcoxon rank sum test with continuity correction

data:  rainfall by yield_category
W = 7628, p-value = 0.3828
alternative hypothesis: true location shift is not equal to 0
```

#perform Wilcoxon ranksum test for equality of medians

wilcox.test(rainfall ~ yield_category, data = crop_recommendation)

P-values > 0.05, therefore, we accept the null hypothesis that the median rainfall is similar for high and low yield.

Hypothesis testing methods



Kruskal-Wallis *H-Test*

- Tests the equality of medians in more than two groups.
- Corresponds to ANOVA for more than two means.
- Uses chi square (χ^2) distribution with $p - 1$ df

Kruskal-Wallis H-Test, example

Hypotheses:

Null hypothesis:

The medians of values for each group are equal.

Alternative hypothesis:

The medians of values for each group are not equal.

Kruskal: RStudio

#perform a normality test

#check for normality
shapiro.test(crop_recommendation$rainfall)

```
> kruskal.test(rainfall ~ Crop, data = crop_recommendation)

        Kruskal-Wallis rank sum test

data:  rainfall by Crop
Kruskal-Wallis chi-squared = 180.1, df = 7, p-value < 2.2e-16
```

#perform a Kruskal Wallis test of equality of medians.
kruskalTest(rainfall ~ Crop, data = crop_recommendation)

P-values < 0.05, therefore, medians are different.

Post hoc test for non-parametric comparisons

```
##perform dunns' post hoc multiple comparison test
install.packages("FSA")
library(FSA)
dunnTest(rainfall ~ Crop, data = crop_recommendation, method = "holm")
```

Comparison	Z	P.unadj	P.adj
chickpea - coffee	-6.9202262	4.509229e-12	1.082215e-10
chickpea - cotton	-0.3653969	7.148152e-01	1.000000e+00
coffee - cotton	6.5548294	5.570554e-11	1.225522e-09
chickpea - maize	-5.0337370	4.810094e-07	9.620187e-06
coffee - maize	1.8864892	5.922906e-02	2.961453e-01
cotton - maize	-4.6683402	3.036428e-06	5.465571e-05
chickpea - potato	-4.3689565	1.248416e-05	1.872625e-04
coffee - potato	2.5512698	1.073312e-02	1.180643e-01
cotton - potato	-4.0035596	6.239647e-05	8.111541e-04
maize - potato	0.6647806	5.061909e-01	1.000000e+00
chickpea - rice	-9.2437041	2.381085e-20	6.667039e-19
coffee - rice	-2.3234778	2.015350e-02	1.612280e-01
cotton - rice	-8.8783072	6.788555e-19	1.765024e-17
maize - rice	-4.2099670	2.554080e-05	3.575712e-04
potato - rice	-4.8747476	1.089477e-06	2.070006e-05
chickpea - sugarcane	-9.0001061	2.254996e-19	6.088488e-18
coffee - sugarcane	-2.0798799	3.753655e-02	2.252193e-01
cotton - sugarcane	-8.6347093	5.887659e-18	1.471915e-16
maize - sugarcane	-3.9663691	7.297583e-05	8.757099e-04
potato - sugarcane	-4.6311497	3.636409e-06	6.181895e-05
rice - sugarcane	0.2435979	8.075422e-01	8.075422e-01
chickpea - wheat	-2.4889756	1.281117e-02	1.153006e-01
coffee - wheat	4.4312506	9.368813e-06	1.499010e-04
cotton - wheat	-2.1235788	3.370538e-02	2.359377e-01
maize - wheat	2.5447614	1.093524e-02	1.093524e-01
potato - wheat	1.8799808	6.011069e-02	2.404428e-01
rice - wheat	6.7547284	1.431031e-11	3.291370e-10
sugarcane - wheat	6.5111305	7.458731e-11	1.566333e-09

significant

Not significant

Predictive Analytics

Regression

Classification

Regression

- Regression analysis is a very widely used statistical tool to establish a relationship model between two variables.
- One of these variable is called predictor variable whose value is gathered through experiments.
- The other variable is called response variable whose value is derived from the predictor variable.

- It is specified by a mathematical function $y = ax+b$
- **y** is the response variable.
- **x** is the predictor variable.
- **a** and **b** are constants which are called the coefficients.

Establishing linear regression

- Gathering a sample of observed values of independent and corresponding dependent variables.
- Create a relationship model using the **lm()** function in R.
- Find the coefficients from the model created and create the mathematical equation using these.
- Get a summary of the relationship model to know the average error in prediction. Also called **residuals**.
- To predict the outcome of new observations, use the **predict()** function in R.

Using the lm() function

- The lm() function establishes a relationship between a dependent and independent variable.

- Simple linear regressions have only one predictor variable
- Example

#Simple Linear regression (Predict milk yield)

lin_mod <- lm(milk_yield_l~body_weight_kg, data = Data)

#model summary

summary(lin_mod)

```
lm(formula = milk_yield_l ~ body_weight_kg, data = Data)

Residuals:
    Min      1Q  Median      3Q     Max 
-46.807 -11.384  -0.439  11.356  60.973 

Coefficients:
               Estimate Std. Error t value Pr(>|t|)    
(Intercept)    90.14077    6.13697   14.69   <2e-16 ***
body_weight_kg  0.61945    0.01219   50.81   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 16.67 on 318 degrees of freedom
Multiple R-squared:  0.8904,	Adjusted R-squared:   0.89 
F-statistic:  2582 on 1 and 318 DF,  p-value: < 2.2e-16
```

Multiple linear regression

- Multiple linear regressions have more than one predictor variable
- Example

#Multiple Linear regression (Predict milk yield)

```
lin_mod <- lm(milk_yield_l~body_weight_kg+temp_c, data = Animal_data)
```

#model summary

```
summary(lin_mod)
```

```
Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)    96.67341    7.31920  13.208   <2e-16 ***
body_weight_kg  0.62258    0.01231  50.576   <2e-16 ***
temp_c         -0.32801    0.20149  -1.628    0.105
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 16.62 on 317 degrees of freedom
Multiple R-squared:  0.8913,    Adjusted R-squared:  0.8906
F-statistic:  1299 on 2 and 317 DF,  p-value: < 2.2e-16
```

Multiple linear regression

```
lm(formula = milk_yield_l ~ feed_intake_kg + age_months + body_weight_kg +
    pasture_quality_index + lactation_days + herd_size + temp_c +
    humidity_pct + protein_pct_feed + energy_density_mjkg + vet_visits_per_year +
    parasite_index, data = Data)

Residuals:
     Min       1Q   Median       3Q      Max
-14.1838  -3.3098   0.1194   3.4181  15.5439

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)            8.987559   4.578683   1.963   0.0506 .
feed_intake_kg         0.758120   0.100976   7.508 6.57e-13 ***
age_months            -0.427099   0.033543 -12.733  < 2e-16 ***
body_weight_kg         0.600582   0.004139 145.120  < 2e-16 ***
pasture_quality_index  0.359638   0.030014  11.982  < 2e-16 ***
lactation_days         0.485212   0.010530  46.078  < 2e-16 ***
herd_size             -0.178202   0.015824 -11.261  < 2e-16 ***
temp_c                -0.329981   0.068315  -4.830 2.16e-06 ***
humidity_pct           0.223439   0.031837   7.018 1.44e-11 ***
protein_pct_feed       0.056098   0.157114   0.357   0.7213
energy_density_mjkg    0.175923   0.206867   0.850   0.3958
vet_visits_per_year   -0.145657   0.477071  -0.305   0.7603
parasite_index         0.697821   0.102918   6.780 6.15e-11 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5.098 on 307 degrees of freedom
Multiple R-squared:  0.9901,    Adjusted R-squared:  0.9897
F-statistic:  2557 on 12 and 307 DF,  p-value: < 2.2e-16
```