

R – INTERMEDIATE

Digital Scholarship HUB

University of Illinois at Chicago

digitalscholarship@uic.edu

TABLE OF CONTENTS

DEFINE A FUNCTION.....	1
➤ BUILT IN FUNCTIONS.....	1
➤ USER DEFINED FUNCTIONS.....	2
DATA VISUALIZATION.....	3
➤ BAR PLOT.....	4
➤ HISTOGRAM.....	7
➤ PIE CHART.....	8
➤ BOX PLOT.....	9
➤ SCATTER PLOT.....	13
HYPOTHESIS TESTING.....	14
➤ CONFIDENCE INTERVAL.....	14
➤ P-Value.....	15
➤ T-TEST.....	16
➤ ANOVA.....	18
➤ CORRELATION TEST.....	20
REFERENCES.....	21

DEFINE A FUNCTION

Function are lines of codes which are executed in a sequential order in order to perform a certain task. In other words, it is a set of statements which are executed together to accomplish a certain task. R like any programming language, has many built in functions (also called pre-defined functions) but also allows users to create their own functions which are known as user defined functions.

BUILT IN FUNCTIONS

These are functions which are built inside R which are readily available for all users with access to R.

E.g.: `seq()` - Print a sequence of numbers
`mean()` - Find the mean value of a set of numbers
`sum()` - Find the sum of a set of numbers

```
> seq(25,35)
[1] 25 26 27 28 29 30 31 32 33 34 35
> mean(25:35)
[1] 30
> sum(25:35)
[1] 330
```

You can refer [Link1](#) & [Link2](#) for some of the most frequently used built in R functions.

USER DEFINED FUNCTION

These are functions which are manually defined by user and are not available in R by default. Users can create a function based on their own requirements. These functions are useful when a block of code is required to be performed repetitively.

Syntax

```
function_name <- function(argument1, argument2, .....)  
{  
  Body of the function (or) statements  
  return()  
}
```

Calling the function

```
function_name()
```

<pre>> add_two_numbers <- function(a,b) + { + c <- a+b + return(c) + } > add_two_numbers(1,2) [1] 3 > result <- add_two_numbers(1,2) > result [1] 3</pre>	<pre>> print_hello <- function(){ + print("Hello") + print("Function successfully called") + } > print_hello() [1] "Hello" [1] "Function successfully called"</pre>
--	--

DATA VISUALIZATION

In R data can be visualized using the basic **plot** function. The syntax for the plot command is as below:

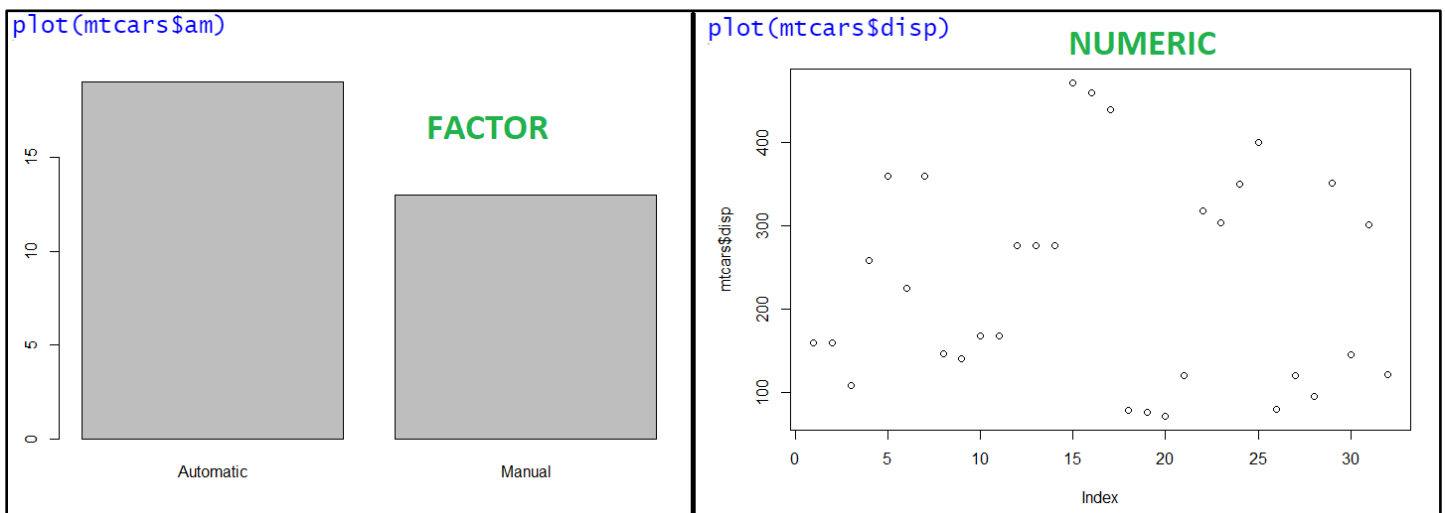
plot(data)

Before proceeding ahead kindly load the **mtcars** data using the below commands

```
library(datasets)
data("mtcars")
mtcars$cyl <- as.factor(mtcars$cyl)
mtcars$vs <- as.factor(mtcars$vs)
mtcars$am <- as.factor(mtcars$am)
mtcars$gear <- as.factor(mtcars$gear)
mtcars$carb <- as.factor(mtcars$carb)
levels(mtcars$am) <- c(levels(mtcars$am), "Automatic", "Manual")
mtcars$am[mtcars$am == 0] <- "Automatic"
mtcars$am[mtcars$am == 1] <- "Manual"
mtcars$am <- factor(mtcars$am)
str(mtcars)
```

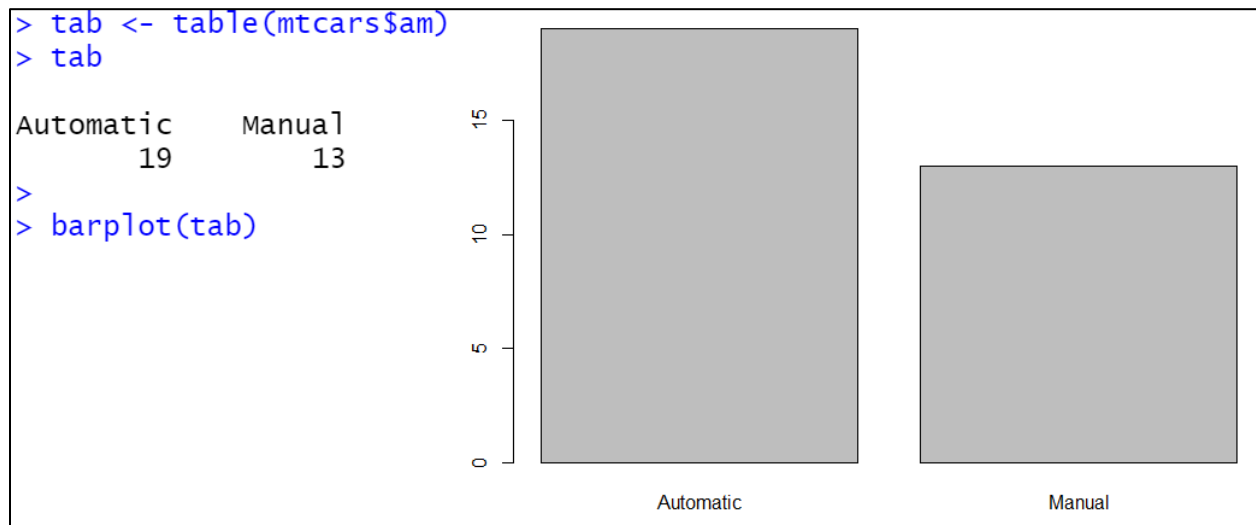
The plot function creates the plots as below based on data type:

Plot	Data type
Bar plot	Factor
Scatter plot	Numeric



BARPLOT

To create a barplot in R we can use the **barplot()** command, but the data passed to the function must be of a table type which contains the count of each factor.

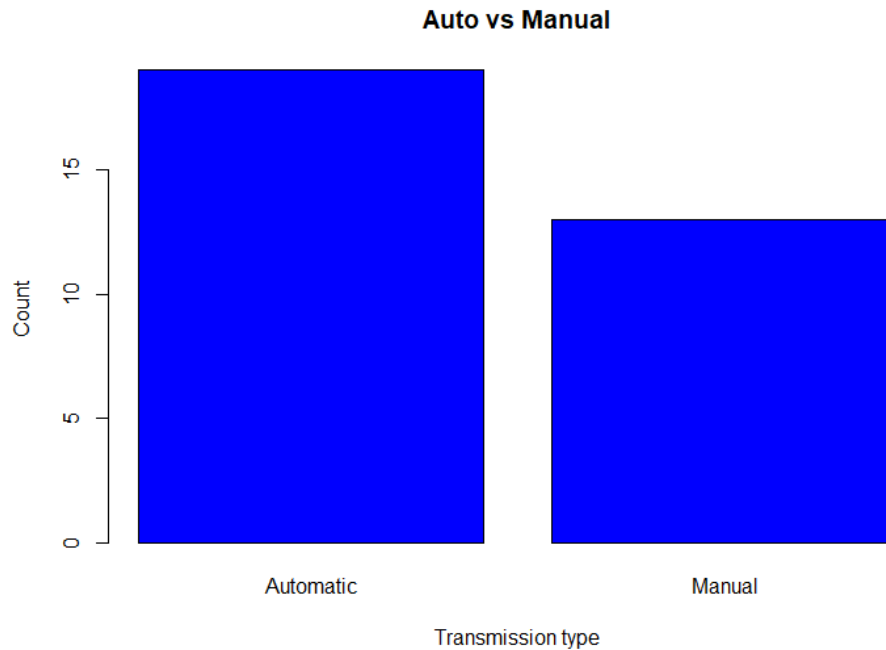


Syntax:

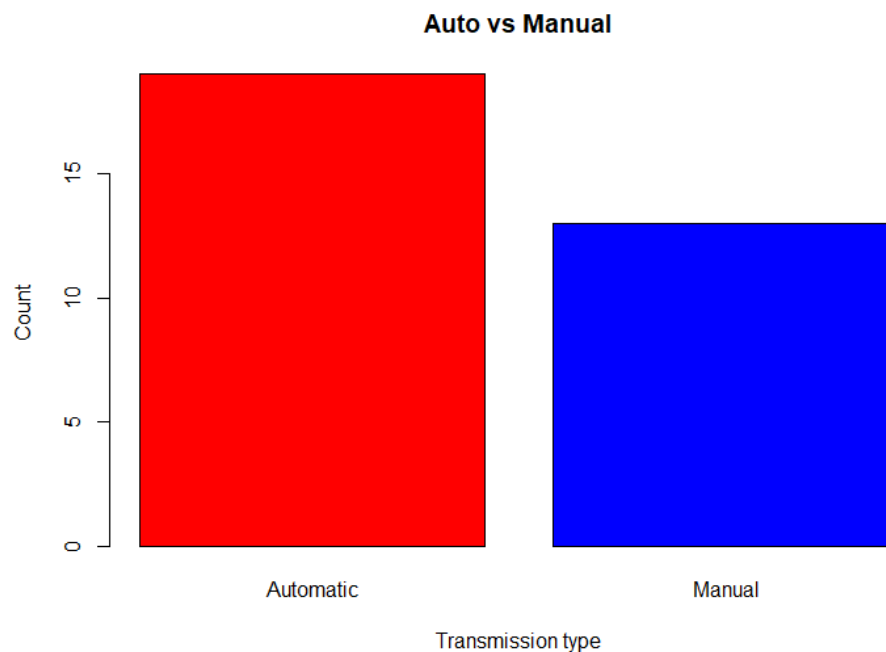
You can create a better plot using the below arguments.

barplot(data,	<i>Data in the form of a table</i>
main = ,	<i>The heading of the plot</i>
xlab = ,	<i>The x-axis label</i>
ylab = ,	<i>The y-axis label</i>
col =)	<i>Color of the bar</i>

```
# same color for all bars  
barplot(tab, main = "Auto vs Manual",  
        xlab = "Transmission type", ylab = "Count",  
        col="blue")
```



```
# Multiple colors  
barplot(tab, main = "Auto vs Manual",  
        xlab = "Transmission type", ylab = "Count",  
        col=c("red", "blue"))
```



To compare the counts of two different factors the above syntax says the same, only the data passed into the **barplot()** function is now a table of two factors.

The **legend()** function can be used to include a legend in the plots but **MUST** be run immediately after the plot command.

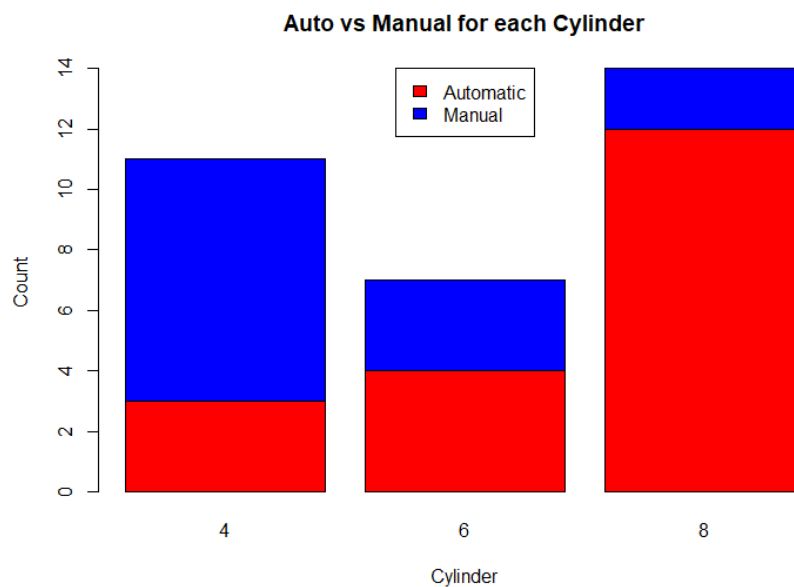
Syntax:

Legend (*location_of_legend*, *legend = legend_keys*, *fill = colors_of_the_keys*)

```
# Two factors
tab <- table(mtcars$am,mtcars$cyl)
tab
      4   6   8
Automatic 3  4 12
Manual    8  3  2

barplot(tab, col = c("red","blue"),
        main = "Auto vs Manual for each cylinder",
        xlab = "cylinder", ylab ="count")

legend("top", legend = c("Automatic", "Manual"), fill = c("red", "blue"))
```

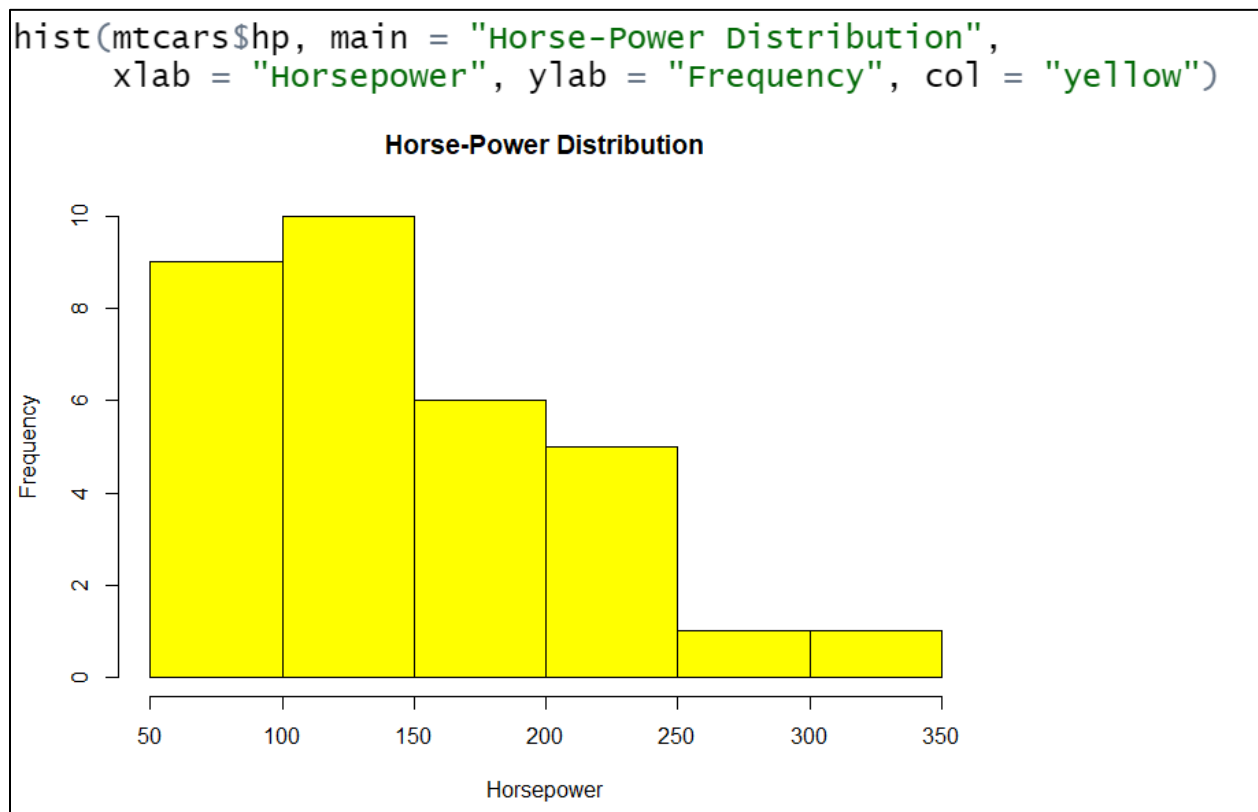


HISTOGRAM

A histogram is similar to a barplot but is used for numeric data types. Instead of counting the occurrence of each individual value, a histogram divides the data into bins (buckets/ranges) depending on the entire data range and displays the count.

Syntax:

```
hist(data, main = "Plot Heading", xlab = "X-Axis Label", ylab = "Y-Axis Label", col = "Bar color")
```



From the plot we can determine:

Horsepower range in dataset - 50 to 350

Cars with horsepower between 50 to 100 - 9

Cars with horsepower between 100 to 150 - 10

Thus, a histogram gives an idea about the distribution of the numeric variable in our dataset.

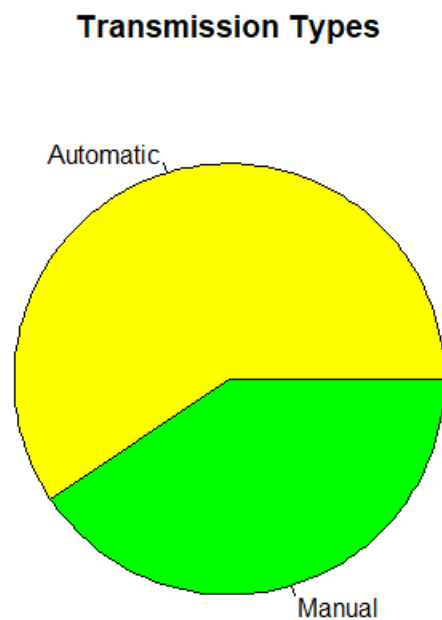
PIE CHART

Pie-Chart is used for the same conditions as a bar chart but is mostly when it is required to show dominance of one or two particular value(s) in comparison with others.

Syntax:

<code>pie(data,</code>	<i>Data in form of a table</i>
<code>col = ,</code>	<i>Colors of the pie slices</i>
<code>main =)</code>	<i>Plot heading</i>

```
# PIE CHART  
pie(table(mtcars$am), col=c("yellow","green"),  
     main = "Transmission Types")
```



BOX PLOT

A Box Plot is used to understand the distribution of numerical type data in the data set. It helps to understand how data is grouped, if data is skewed and identify the outliers in the data.

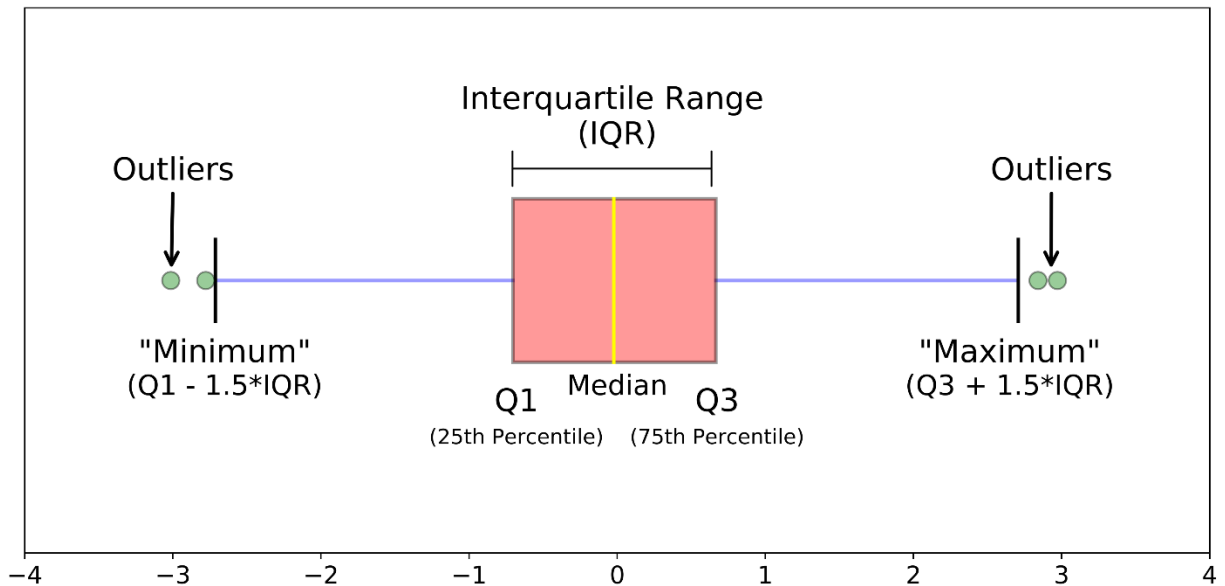


Image Source - [towardsdatascience](https://towardsdatascience.com/)

A boxplot represents data in the below categories:

Minimum - Mean – (2 X Standard Deviation)

Q1 - 25th Percentile

Median - 50th Percentile

Q3 - 75th Percentile

Maximum - Mean + (2 X Standard Deviation)

Outlier - The 0.7% of the data which are more than 2 Standard deviations away from mean

Kindly find a video explaining in detail percentiles [here](#).

A Box Plot can be related to a normal distribution as shown below:

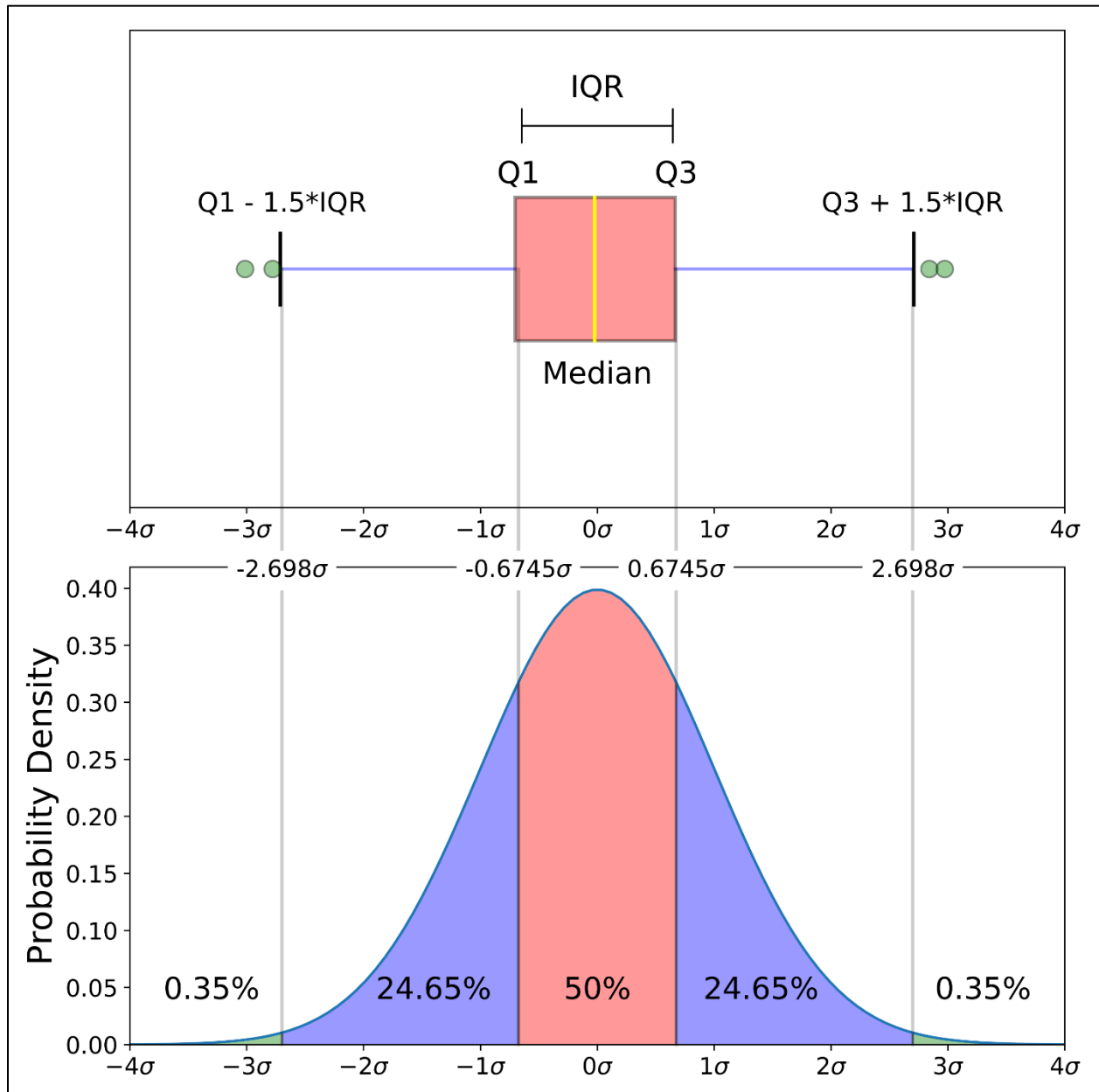
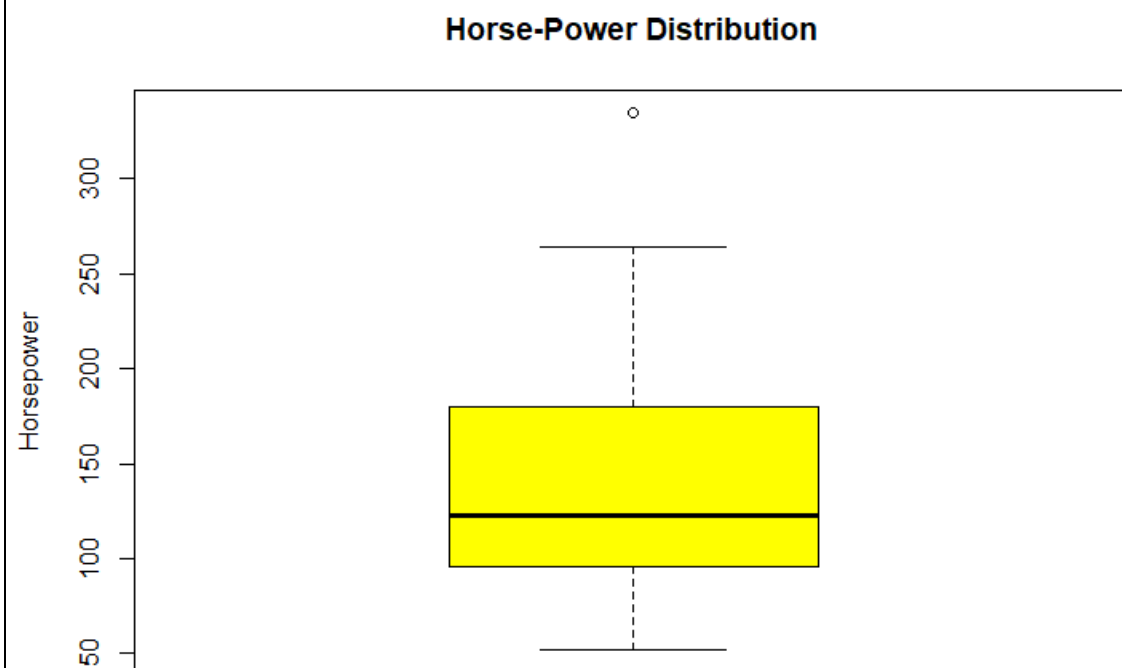


Image Source - [towardsdatascience](https://towardsdatascience.com/)

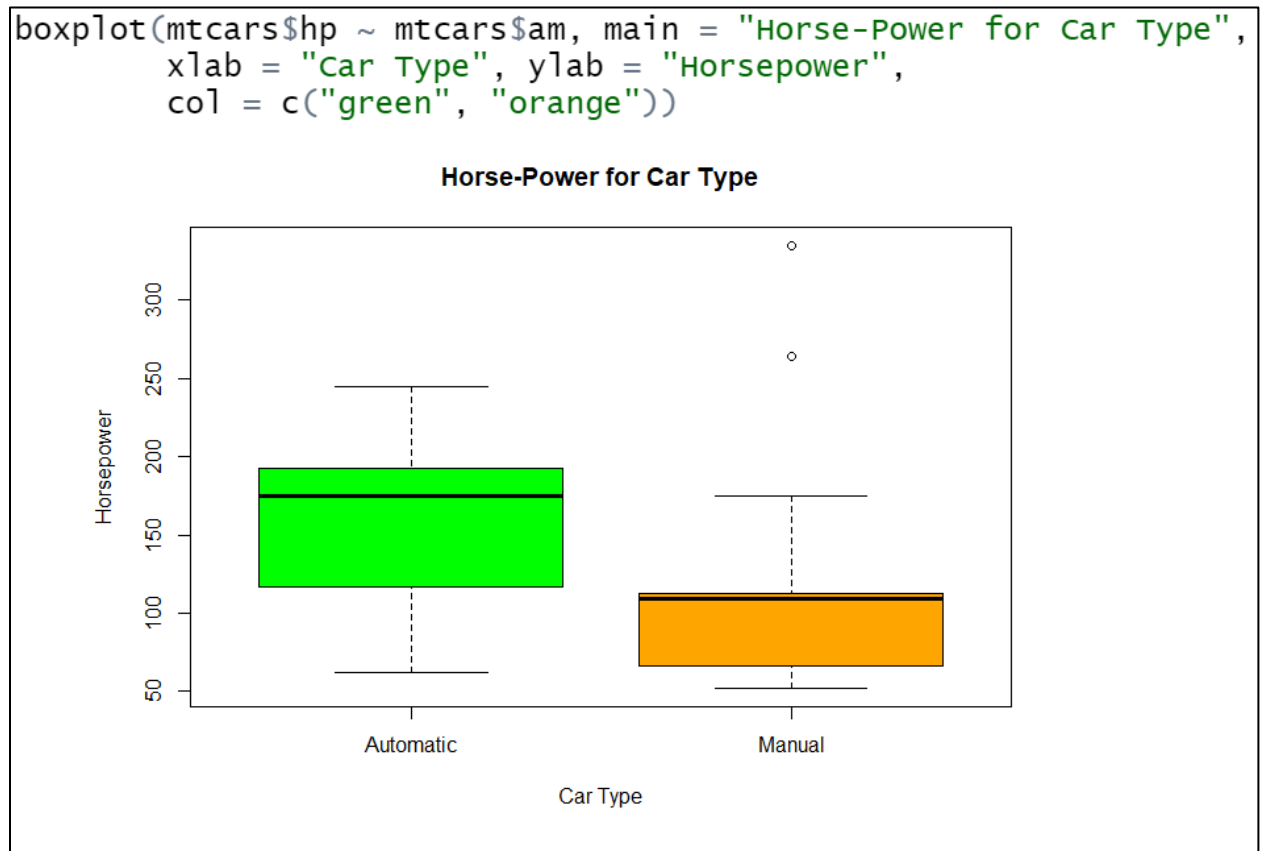
Syntax:

`boxplot(data,` *Numerical data (or) column for a data frame*
 `main =,` *Plot Title*
 `ylab = ,` *Y-Axis Label*
 `col =)` *Color of the plot*

```
boxplot(mtcars$hp, main = "Horse-Power Distribution",  
        ylab = "Horsepower", col = "yellow")
```



Boxplot can also be used to analyze the distribution of data across various factors in a column.

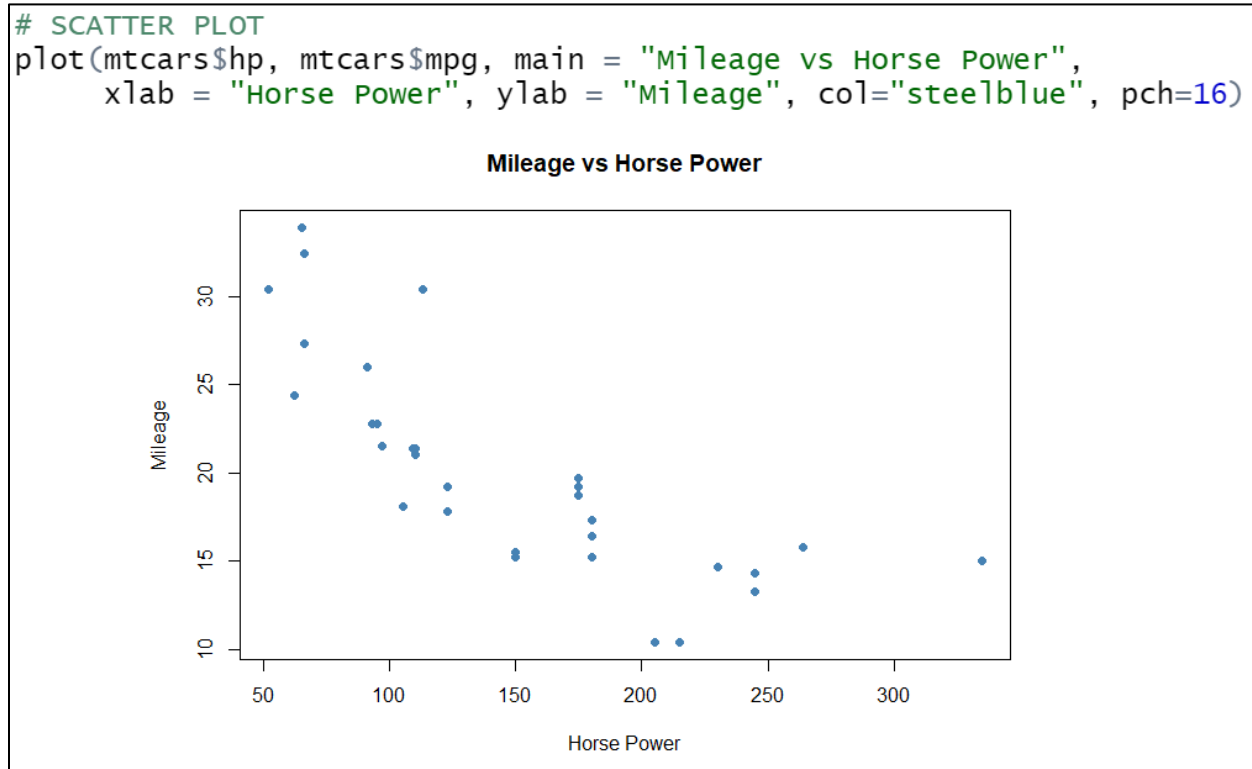


SCATTER PLOT

Scatter plot can be used to identify if any relationship exists between two numeric variables.

Syntax:

<code>plot(Numerical_column1,</code>	<i>Column of numerical data type along X-Axis</i>
<code> Numerical_column2,</code>	<i>Column of numerical data type along Y-Axis</i>
<code> main = ,</code>	<i>Plot Title</i>
<code> xlab = ,</code>	<i>X-Axis Label</i>
<code> ylab = ,</code>	<i>Y-Axis Label</i>
<code> col =)</code>	<i>Plot Color</i>



It can be seen from the plot that as the Horse-Power of the car increased the car Mileage decreases.

HYPOTHESIS TESTING

Hypothesis testing is performed to identify if there is a relationship between the attributes (columns) of our data set. Hypothesis testing is only used to confirm that there is a relation between the attributes considered but does not define the nature of the relationship.

To perform hypothesis testing, we first initially form 2 different hypotheses:

- Null Hypothesis – No difference between data considered
- Alternate Hypothesis – There is a difference between the data considered

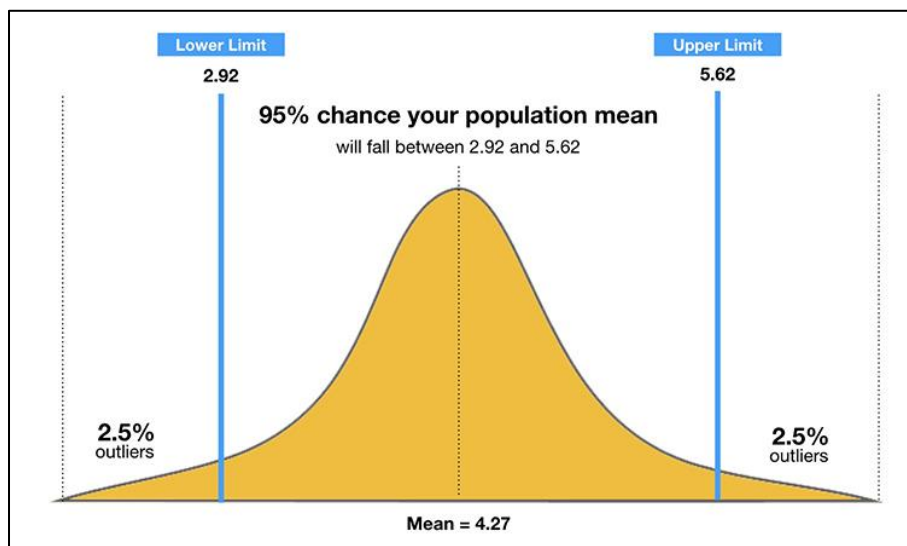
For example, if we want to perform a hypothesis testing to check if a car's transmission type (Manual or Automatic) has an impact on the price of a car then our hypothesis would be:

- Null Hypothesis – There is no difference in the price range of a car based on its type of transmission
- Alternate Hypothesis – There is a statistical difference in the price range of a car based on its type of transmission

Kindly find a video explaining hypothesis testing in more detail [here](#).

CONFIDENCE INTERVAL

The confidence interval determines the range of values which the true mean lies. For example, if data is collected regarding the height of men then, a 95% confidence interval provides the range of height within which the true mean of all men's height lie.



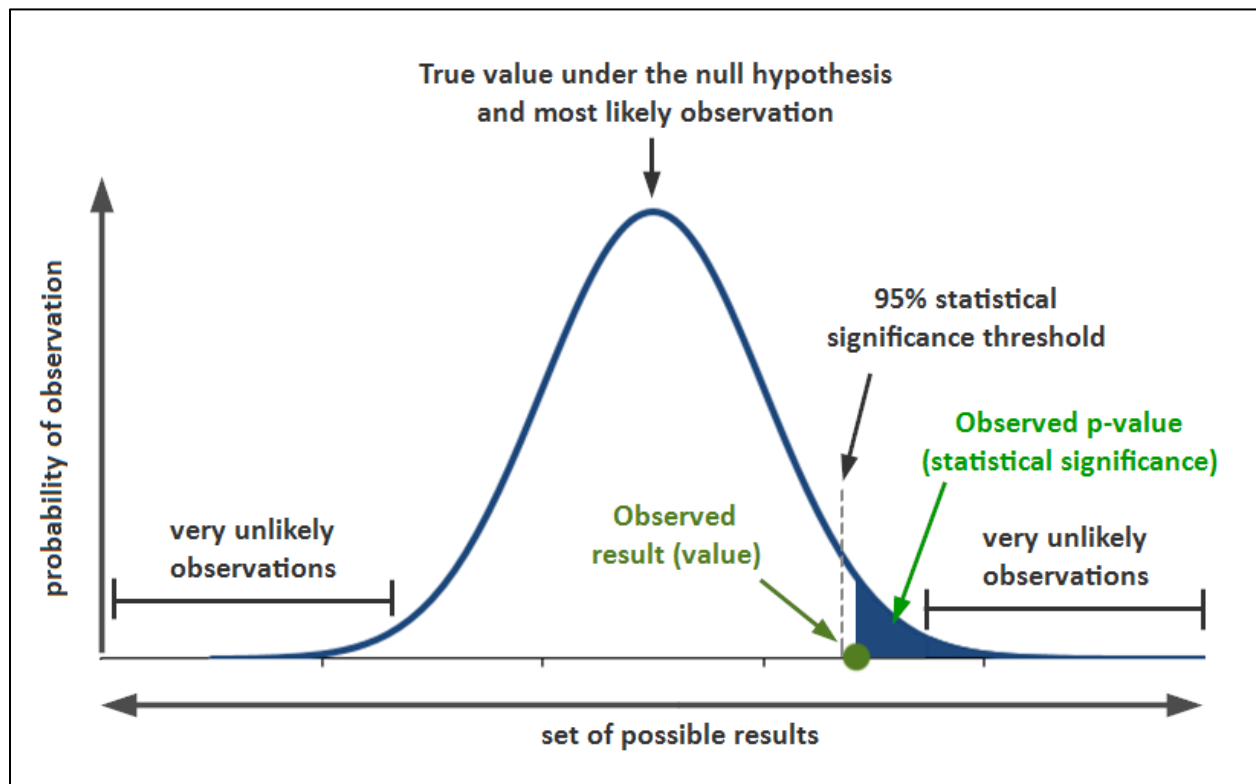
P-Value

The p-value of a test provides the probability of gaining results in the extreme cases under the assumption that the null hypothesis is correct. If a p-value is large, then the probability of such a result is very high and if p-value is low then the probability of such a result is very low under the considered null hypothesis.

We chose a significance value to determine when to reject the null hypothesis. Conventionally, 0.05 is chosen as the significance level such that if p-value is less than 0.05 then we reject the null hypothesis and accept the alternate hypothesis.

You can refer to the below links to learn more in detail:

- Confidence Interval – [MathisFun](#), [YouTube](#)
- P-value – [StatsDirect](#), [YouTube](#), [YouTube2](#), [Towards Data Science](#)



T-TEST

The t-test is used to run a hypothesis testing on one (or) two levels of same factor.


Syntax

t.test(Factor 1,	Values of the first factor
Factor 2,	Values of the second factor
alternative =)	Check if factor 1 mean is smaller or greater than factor 2 (optional)

To determine if the transmission type of a car has an impact on its mileage

```
> Auto_mileage <- mtcars[mtcars$am == "Automatic", "mpg"]
> Manual_mileage <- mtcars[mtcars$am == "Manual", "mpg"]
> t.test(Auto_mileage, Manual_mileage)
```

welch Two sample t-test




```
data: Auto_mileage and Manual_mileage
t = -3.7671, df = 18.332, p-value = 0.001374
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -11.280194 -3.209684
sample estimates:
mean of x mean of y
 17.14737  24.39231
```

Since the p-value is less than 0.05 we can accept the alternate hypothesis that there is a difference in the mileage of a car based on its transmission type.

To determine if the transmission type of a car has an impact on its horsepower

```
> Auto_hp <- mtcars[mtcars$am == "Automatic", "hp"]
> Manual_hp <- mtcars[mtcars$am == "Manual", "hp"]
> t.test(Auto_hp, Manual_hp)
```

welch Two sample t-test



```
data: Auto_hp and Manual_hp
t = 1.2662, df = 18.715, p-value = 0.221
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -21.87858  88.71259
sample estimates:
mean of x mean of y
 160.2632  126.8462
```

Since the p-value is greater than 0.05 we can accept the null hypothesis that there is no difference in the horsepower of a car based on its transmission type.

We can use the **alternative** parameter to determine if the first factor under consideration has a higher mean compared to the second factor.

```
> t.test(Auto_mileage, Manual_mileage, alternative = "less")

welch Two sample t-test

data: Auto_mileage and Manual_mileage
t = -3.7671, df = 18.332, p-value = 0.0006868
alternative hypothesis: true difference in means is less than 0
95 percent confidence interval:
 -Inf -3.913256
sample estimates:
mean of x mean of y
 17.14737  24.39231

> t.test(Auto_mileage, Manual_mileage, alternative = "greater")

welch Two sample t-test

data: Auto_mileage and Manual_mileage
t = -3.7671, df = 18.332, p-value = 0.9993
alternative hypothesis: true difference in means is greater than 0
95 percent confidence interval:
 -10.57662      Inf
sample estimates:
mean of x mean of y
 17.14737  24.39231
```

From the test and the resulting p-value(s) we can verify that a car with an Automatic transmission has a lower mileage in comparison to manual transmission car in the dataset.

ANOVA (Analysis of Variance)

The ANOVA test is performed to run hypothesis testing on a factor with more than two levels.

In our **mtcars** dataset the “cylinder” attribute has three levels while “carburetors” attribute has six levels.

```
> levels(mtcars$cyl)
[1] "4" "6" "8"
> levels(mtcars$carb)
[1] "1" "2" "3" "4" "6" "8"
```

Syntax

```
aov(Numerical_column_name ~ Categorical_column_name, data = dataframe_name)
```

```
TukeyHSD(ANOVA_output)
```

The initial anova test only provides a result stating if there is an overall difference. To check for difference between each individual level in the factor we use the **TukeyHSD** function.

```
> mileage.aov <- aov(mpg~cyl, data=mtcars)
> summary(mileage.aov)
```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
cyl	2	824.8	412.4	39.7	4.98e-09 ***
Residuals	29	301.3	10.4		

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> TukeyHSD(mileage.aov)
```

Tukey multiple comparisons of means
95% family-wise confidence level

Fit: aov(formula = mpg ~ cyl, data = mtcars)

```
$cyl
```

	diff	lwr	upr	p adj
6-4	-6.920779	-10.769350	-3.0722086	0.0003424
8-4	-11.563636	-14.770779	-8.3564942	0.0000000
8-6	-4.642857	-8.327583	-0.9581313	0.0112287

```

> horsepower.aov <- aov(hp~carb, data=mtcars)
> summary(horsepower.aov)
              Df Sum Sq Mean Sq F value    Pr(>F)
carb              5   90319    18064    8.476 7.31e-05 ***
Residuals        26   55408     2131
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> TukeyHSD(horsepower.aov)
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = hp ~ carb, data = mtcars)

$carb
      diff          lwr          upr        p adj
2-1   31.2  -38.6970658  101.0971  0.7429980
3-1   94.0   -3.8754692  191.8755  0.0650833
4-1  101.0   31.1029342  170.8971  0.0018434
6-1   89.0  -62.6280249  240.6280  0.4809394
8-1  249.0   97.3719751  400.6280  0.0003888
3-2   62.8  -30.5672469  156.1672  0.3347215
4-2   69.8    6.3694463  133.2306  0.0248797
6-2   57.8  -90.9578343  206.5578  0.8357649
8-2  217.8   69.0421657  366.5578  0.0015865
4-3    7.0  -86.3672469  100.3672  0.9998994
6-3   -5.0 -168.7769853  158.7770  0.9999988
8-3  155.0   -8.7769853  318.7770  0.0713126
6-4  -12.0 -160.7578343  136.7578  0.9998557
8-4  148.0   -0.7578343  296.7578  0.0517459
8-6  160.0  -40.5850229  360.5850  0.1760952

```

CORRELATION TEST

The correlation test is used to run a hypothesis testing on two different numerical attributes.

Syntax

`cor.test(Numerical_Attribute_1, Numerical_Attribute_2)`

```
> cor.test(mtcars$mpg, mtcars$hp)

        Pearson's product-moment correlation

data:  mtcars$mpg and mtcars$hp
t = -6.7424, df = 30, p-value = 0.0000001788
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.8852686 -0.5860994
sample estimates:
          cor
-0.7761684

> cor.test(mtcars$qsec, mtcars$wt)

        Pearson's product-moment correlation

data:  mtcars$qsec and mtcars$wt
t = -0.97191, df = 30, p-value = 0.3389
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.4933536  0.1852649
sample estimates:
          cor
-0.1747159
```

From the results we can determine that there exists a relationship between mileage and horsepower but no relationship between the car's weight and quarter mile time (qsec).

REFERENCES

Wikipedia (2021, January 14). *P-value*. Retrieved from <https://en.wikipedia.org/wiki/P-value>