NCSSM miniterm R Tutorial

Andrew Soborowski

2023-01-17

This tutorial was heavily adapted from Bio 304, offered at Duke and written by Dr. Paul Magwene. The full course and notebooks are available on github at: https://bio304-class.github.io/bio304-book/

Basic R commands: math

```
10 + 5
```

```
## [1] 15
```

```
10 - 5
```

```
## [1] 5
```

```
10 / 5
```

```
## [1] 2
```

```
10 * 5
```

```
## [1] 50
```

```
10 ^ 5
```

```
## [1] 1e+05
```

```
10 ** 5
```

```
## [1] 1e+05
```

```
10 ^ 5 == 100000
```

```
## [1] TRUE
```

```
2 ^ 3
```

```
## [1] 8
```

```
10 * pi
```

```
## [1] 31.41593
```

```
10 %% 3 #Modulus division
```

```
## [1] 1
```

```
10 %/% 3 #Integer division
```

```
## [1] 3
```

```
1/0 #Inf
```

```
## [1] Inf
```

```r
-1/0 #Signed Inf
```

```
## [1] -Inf
```

Order of Operations Matters

```r
(10+2)/4-5
```

```
## [1] -2
```

```r
(10+2)/(4-5)
```

```
## [1] -12
```

R has built-in math functions

```r
abs(-3)   # absolute value
```

```
## [1] 3
```

```r
abs(3)
```

```
## [1] 3
```

```r
cos(pi/3) # cosine
```

```
## [1] 0.5
```

```r
sin(pi/3) # sine
```

```
## [1] 0.8660254
```

```r
log(10)   # natural logarithm
```

```
## [1] 2.302585
```

```r
log10(10) # log base 10
```

```
## [1] 1
```

```r
log2(10) # log base 2
```

```
## [1] 3.321928
```

```r
exp(1)    # exponential function
```

```
## [1] 2.718282
```

```r
sqrt(10)  # square root
```

```
## [1] 3.162278
```

```r
10 ^ 0.5  # same as square root
```

```
## [1] 3.162278
```

Variables

```r
x = 10
x <- 10

sin(x)
```

```
## [1] -0.5440211
```

```r
x = pi
sin(x)
```

```
## [1] 1.224647e-16
```

Logical values (Boolean values)

```r
10 < 9  # is 10 less than 9?
```

```
## [1] FALSE
```

```r
10 > 9  # is 10 greater than 9?
```

```
## [1] TRUE
```

```r
10 <= (5 * 2) # less than or equal to?
```

```
## [1] TRUE
```

```r
10 >= pi # greater than or equal to?
```

```
## [1] TRUE
```

```r
10 == 10 # equals?
```

```
## [1] TRUE
```

```r
10 != 10 # does not equal?
```

```
## [1] FALSE
```

A note on precision

```r
10 == (sqrt(10)^2)
```

```
## [1] FALSE
```

```r
4  == (sqrt(4)^2)
```

```
## [1] TRUE
```

```r
x = sqrt(10)^2
x
```

```
## [1] 10
```

```r
print(x, digits = 22)
```

```
## [1] 10.0000000000000177636
```

Logical operators

```r
x = TRUE
y = FALSE

!x        #Negation
```

```
## [1] FALSE
```

```r
x & y     #AND
```

```
## [1] FALSE
```

```r
x | y     #OR
```

```
## [1] TRUE
```

```
x | y     #OR
```

```
## [1] TRUE
```

```
xor(x,y)  #Exclusive OR
```

```
## [1] TRUE
```

Characters

```
x = "bob"
nchar(x)
```

```
## [1] 3
```

```
y = "jones"
paste(x,y, sep="")
```

```
## [1] "bobjones"
```

```
paste(x,y, sep=" ")
```

```
## [1] "bob jones"
```

```
x = "1"
y = "2"
#x + y  #This breaks. Why?
```

Vectors

```
x = c(2,4,6,8)
x
```

```
## [1] 2 4 6 8
```

```
length(x)
```

```
## [1] 4
```

```
typeof(x)
```

```
## [1] "double"
```

```
y = c(TRUE, TRUE, FALSE)
y
```

```
## [1]  TRUE  TRUE FALSE
```

```
typeof(y)
```

```
## [1] "logical"
```

```
length(y)
```

```
## [1] 3
```

```
z = c(1,3,5,7)

xz = c(x,z)
xz
```

```
## [1] 2 4 6 8 1 3 5 7
```

Vector Math

```r
x = c(2,4,6,8)
y = c(0,1,3,5)
#Vector and value: vectorized operation
x*2
```

```
## [1]  4  8 12 16
```

```r
x-pi
```

```
## [1] -1.1415927  0.8584073  2.8584073  4.8584073
```

```r
sin(x)
```

```
## [1]  0.9092974 -0.7568025 -0.2794155  0.9893582
```

```r
cos(x*pi)
```

```
## [1] 1 1 1 1
```

```r
#Vecotr and vector: elementwise operation
x+y
```

```
## [1]  2  5  9 13
```

```r
x*y
```

```
## [1]  0  4 18 40
```

```r
x/y
```

```
## [1] Inf 4.0 2.0 1.6
```

Stats and vectors

```r
data = c(7,7,6,2,9,9,7,4,10,5)

sum(data)
```

```
## [1] 66
```

```r
min(data)
```

```
## [1] 2
```

```r
max(data)
```

```
## [1] 10
```

```r
mean(data)
```

```
## [1] 6.6
```

```r
median(data)
```

```
## [1] 7
```

```r
var(data)
```

```
## [1] 6.044444
```

```r
sd(data)
```

```
## [1] 2.458545
```

```r
summary(data)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2.00    5.25    7.00    6.60    8.50   10.00
```

Vector Indexing: 1 based

```
x = c(2,4,6,8)
x[1]
```

```
## [1] 2
```

```
x[4]
```

```
## [1] 8
```

```
x[1:2]
```

```
## [1] 2 4
```

```
x[-1]
```

```
## [1] 4 6 8
```

```
x[c(1,4)]
```

```
## [1] 2 8
```

```
x[x>5]
```

```
## [1] 6 8
```

```
x[x<4 | x>8]
```

```
## [1] 2
```

```
x[x<=4 | x>8]
```

```
## [1] 2 4
```

```
x[2] = 5.5
x
```

```
## [1] 2.0 5.5 6.0 8.0
```

```
x[x>5] = 5
x
```

```
## [1] 2 5 5 5
```

DataFrames: our main data structure

```
#We normally make data frames from read in csv files but we can make them manually

age = c(30, 26, 21, 29, 25, 22, 28, 24, 23, 20)
sex = rep(c("M","F"), 5)
wt.in.kg = c(88, 76, 67, 66, 56, 74, 71, 60, 52, 72)
df = data.frame(age = age, sex = sex, wt = wt.in.kg)
df
```

```
##    age sex wt
## 1   30   M 88
## 2   26   F 76
## 3   21   M 67
## 4   29   F 66
## 5   25   M 56
```

```
## 6   22   F 74
## 7   28   M 71
## 8   24   F 60
## 9   23   M 52
## 10  20   F 72
```

DataFrame properties

```
dim(df)
```

```
## [1] 10  3
```

```
nrow(df)
```

```
## [1] 10
```

```
ncol(df)
```

```
## [1] 3
```

```
df[1]
```

```
##    age
## 1   30
## 2   26
## 3   21
## 4   29
## 5   25
## 6   22
## 7   28
## 8   24
## 9   23
## 10  20
```

```
df[1:2]
```

```
##    age sex
## 1   30   M
## 2   26   F
## 3   21   M
## 4   29   F
## 5   25   M
## 6   22   F
## 7   28   M
## 8   24   F
## 9   23   M
## 10  20   F
```

```
df["age"]
```

```
##    age
## 1   30
## 2   26
## 3   21
## 4   29
## 5   25
## 6   22
## 7   28
## 8   24
```

```
## 9   23
## 10  20
```

```
df[c("age","wt")]
```

```
##    age wt
## 1   30 88
## 2   26 76
## 3   21 67
## 4   29 66
## 5   25 56
## 6   22 74
## 7   28 71
## 8   24 60
## 9   23 52
## 10  20 72
```

DataFrame indexing

```
df[1,]
```

```
##   age sex wt
## 1  30   M 88
```

```
df[1:2,]
```

```
##   age sex wt
## 1  30   M 88
## 2  26   F 76
```

```
df[c(1,3,5),]
```

```
##   age sex wt
## 1  30   M 88
## 3  21   M 67
## 5  25   M 56
```

```
df[1,2]
```

```
## [1] "M"
```

```
df[1:3, 2:3]
```

```
##   sex wt
## 1   M 88
## 2   F 76
## 3   M 67
```

```
df[5:10, c("age", "wt")]
```

```
##    age wt
## 5   25 56
## 6   22 74
## 7   28 71
## 8   24 60
## 9   23 52
## 10  20 72
```

```
df["age"]
```

```
##    age
```

```
## 1   30
## 2   26
## 3   21
## 4   29
## 5   25
## 6   22
## 7   28
## 8   24
## 9   23
## 10  20
```

```
df[["age"]]
```

```
##  [1] 30 26 21 29 25 22 28 24 23 20
```

```
df$age
```

```
##  [1] 30 26 21 29 25 22 28 24 23 20
```

```
df[(df$wt >= 60 & df$wt <= 70),]
```

```
##   age sex wt
## 3  21   M 67
## 4  29   F 66
## 8  24   F 60
```

Installing and Using packages

```
#install.packages("tidyverse")
library(tidyverse)
```

```
## Warning: package 'ggplot2' was built under R version 4.3.1
```

```
## Warning: package 'dplyr' was built under R version 4.3.1
```

```
## Warning: package 'stringr' was built under R version 4.3.1
```

```
## Warning: package 'lubridate' was built under R version 4.3.1
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

Tibbles: a tidyverse extension to DataFrames

```
data = tibble(iris)
#Normally, reading in a file will automatically put it in a tibble
```

File Reading

```
#data = read_csv("file_path.csv")
#data = read_tsv("file_path.tsv")
```

dplyr: data transformations for tibbles Commands are chained together using “verbs” to manipulate data frames Each function accepts a tibble as its first arguement I highly reccomend using a cheat sheet for these

Select: subset columns

```
data
```

```
## # A tibble: 150 x 5
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##           <dbl>       <dbl>        <dbl>       <dbl> <fct>
##  1          5.1         3.5          1.4         0.2 setosa
##  2          4.9         3            1.4         0.2 setosa
##  3          4.7         3.2          1.3         0.2 setosa
##  4          4.6         3.1          1.5         0.2 setosa
##  5          5           3.6          1.4         0.2 setosa
##  6          5.4         3.9          1.7         0.4 setosa
##  7          4.6         3.4          1.4         0.3 setosa
##  8          5           3.4          1.5         0.2 setosa
##  9          4.4         2.9          1.4         0.2 setosa
## 10          4.9         3.1          1.5         0.1 setosa
## # i 140 more rows
```

```
select(data, Sepal.Length, Petal.Width)
```

```
## # A tibble: 150 x 2
##    Sepal.Length Petal.Width
##           <dbl>       <dbl>
##  1          5.1         0.2
##  2          4.9         0.2
##  3          4.7         0.2
##  4          4.6         0.2
##  5          5           0.2
##  6          5.4         0.4
##  7          4.6         0.3
##  8          5           0.2
##  9          4.4         0.2
## 10          4.9         0.1
## # i 140 more rows
```

A note on pipes

```
select(data, Sepal.Length, Petal.Width)
```

```
## # A tibble: 150 x 2
##    Sepal.Length Petal.Width
##           <dbl>       <dbl>
##  1          5.1         0.2
##  2          4.9         0.2
##  3          4.7         0.2
##  4          4.6         0.2
##  5          5           0.2
##  6          5.4         0.4
##  7          4.6         0.3
##  8          5           0.2
##  9          4.4         0.2
## 10          4.9         0.1
## # i 140 more rows
```

```r
data %>% select(Sepal.Length, Petal.Width)
```

```
## # A tibble: 150 x 2
##    Sepal.Length Petal.Width
##           <dbl>       <dbl>
##  1          5.1         0.2
##  2          4.9         0.2
##  3          4.7         0.2
##  4          4.6         0.2
##  5          5           0.2
##  6          5.4         0.4
##  7          4.6         0.3
##  8          5           0.2
##  9          4.4         0.2
## 10          4.9         0.1
## # i 140 more rows
```

Filter: search data for conditions

```r
data %>% filter(Sepal.Length > 4.9)
```

```
## # A tibble: 128 x 5
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##           <dbl>       <dbl>        <dbl>       <dbl> <fct>
##  1          5.1         3.5          1.4         0.2 setosa
##  2          5           3.6          1.4         0.2 setosa
##  3          5.4         3.9          1.7         0.4 setosa
##  4          5           3.4          1.5         0.2 setosa
##  5          5.4         3.7          1.5         0.2 setosa
##  6          5.8         4            1.2         0.2 setosa
##  7          5.7         4.4          1.5         0.4 setosa
##  8          5.4         3.9          1.3         0.4 setosa
##  9          5.1         3.5          1.4         0.3 setosa
## 10          5.7         3.8          1.7         0.3 setosa
## # i 118 more rows
```

```r
data %>% filter(Sepal.Length > 4.9, Species == "setosa")
```

```
## # A tibble: 30 x 5
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##           <dbl>       <dbl>        <dbl>       <dbl> <fct>
##  1          5.1         3.5          1.4         0.2 setosa
##  2          5           3.6          1.4         0.2 setosa
##  3          5.4         3.9          1.7         0.4 setosa
##  4          5           3.4          1.5         0.2 setosa
##  5          5.4         3.7          1.5         0.2 setosa
##  6          5.8         4            1.2         0.2 setosa
##  7          5.7         4.4          1.5         0.4 setosa
##  8          5.4         3.9          1.3         0.4 setosa
##  9          5.1         3.5          1.4         0.3 setosa
## 10          5.7         3.8          1.7         0.3 setosa
## # i 20 more rows
```

```r
data %>% filter(Sepal.Length > 4.9 | Species == "setosa")
```

```
## # A tibble: 148 x 5
```

```
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##           <dbl>       <dbl>        <dbl>       <dbl> <fct>
##  1          5.1         3.5          1.4         0.2 setosa
##  2          4.9         3            1.4         0.2 setosa
##  3          4.7         3.2          1.3         0.2 setosa
##  4          4.6         3.1          1.5         0.2 setosa
##  5          5           3.6          1.4         0.2 setosa
##  6          5.4         3.9          1.7         0.4 setosa
##  7          4.6         3.4          1.4         0.3 setosa
##  8          5           3.4          1.5         0.2 setosa
##  9          4.4         2.9          1.4         0.2 setosa
## 10          4.9         3.1          1.5         0.1 setosa
## # i 138 more rows
```

Mutate: make new columns

```
data %>% mutate(Length.Ratio = Petal.Length/Sepal.Length)
```

```
## # A tibble: 150 x 6
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species Length.Ratio
##           <dbl>       <dbl>        <dbl>       <dbl> <fct>           <dbl>
##  1          5.1         3.5          1.4         0.2 setosa          0.275
##  2          4.9         3            1.4         0.2 setosa          0.286
##  3          4.7         3.2          1.3         0.2 setosa          0.277
##  4          4.6         3.1          1.5         0.2 setosa          0.326
##  5          5           3.6          1.4         0.2 setosa          0.28
##  6          5.4         3.9          1.7         0.4 setosa          0.315
##  7          4.6         3.4          1.4         0.3 setosa          0.304
##  8          5           3.4          1.5         0.2 setosa          0.3
##  9          4.4         2.9          1.4         0.2 setosa          0.318
## 10          4.9         3.1          1.5         0.1 setosa          0.306
## # i 140 more rows
```

```
data_ratio = data %>% mutate(Length.Ratio = Petal.Length/Sepal.Length)
```

Arrange: sort by column values

```
data %>% arrange(Sepal.Length)
```

```
## # A tibble: 150 x 5
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##           <dbl>       <dbl>        <dbl>       <dbl> <fct>
##  1          4.3         3            1.1         0.1 setosa
##  2          4.4         2.9          1.4         0.2 setosa
##  3          4.4         3            1.3         0.2 setosa
##  4          4.4         3.2          1.3         0.2 setosa
##  5          4.5         2.3          1.3         0.3 setosa
##  6          4.6         3.1          1.5         0.2 setosa
##  7          4.6         3.4          1.4         0.3 setosa
##  8          4.6         3.6          1           0.2 setosa
##  9          4.6         3.2          1.4         0.2 setosa
## 10          4.7         3.2          1.3         0.2 setosa
## # i 140 more rows
```

```
data %>% arrange(Sepal.Length, Sepal.Width)
```

```
## # A tibble: 150 x 5
```

```
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##           <dbl>       <dbl>        <dbl>       <dbl> <fct>
##  1          4.3         3            1.1         0.1 setosa
##  2          4.4         2.9          1.4         0.2 setosa
##  3          4.4         3            1.3         0.2 setosa
##  4          4.4         3.2          1.3         0.2 setosa
##  5          4.5         2.3          1.3         0.3 setosa
##  6          4.6         3.1          1.5         0.2 setosa
##  7          4.6         3.2          1.4         0.2 setosa
##  8          4.6         3.4          1.4         0.3 setosa
##  9          4.6         3.6          1           0.2 setosa
## 10          4.7         3.2          1.3         0.2 setosa
## # i 140 more rows
```

```
data %>% arrange(desc(Sepal.Length))
```

```
## # A tibble: 150 x 5
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##           <dbl>       <dbl>        <dbl>       <dbl> <fct>
##  1          7.9         3.8          6.4         2   virginica
##  2          7.7         3.8          6.7         2.2 virginica
##  3          7.7         2.6          6.9         2.3 virginica
##  4          7.7         2.8          6.7         2   virginica
##  5          7.7         3            6.1         2.3 virginica
##  6          7.6         3            6.6         2.1 virginica
##  7          7.4         2.8          6.1         1.9 virginica
##  8          7.3         2.9          6.3         1.8 virginica
##  9          7.2         3.6          6.1         2.5 virginica
## 10          7.2         3.2          6           1.8 virginica
## # i 140 more rows
```

Group_by: create groupings within variables

```
data %>% group_by(Species)
```

```
## # A tibble: 150 x 5
## # Groups:   Species [3]
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##           <dbl>       <dbl>        <dbl>       <dbl> <fct>
##  1          5.1         3.5          1.4         0.2 setosa
##  2          4.9         3            1.4         0.2 setosa
##  3          4.7         3.2          1.3         0.2 setosa
##  4          4.6         3.1          1.5         0.2 setosa
##  5          5           3.6          1.4         0.2 setosa
##  6          5.4         3.9          1.7         0.4 setosa
##  7          4.6         3.4          1.4         0.3 setosa
##  8          5           3.4          1.5         0.2 setosa
##  9          4.4         2.9          1.4         0.2 setosa
## 10          4.9         3.1          1.5         0.1 setosa
## # i 140 more rows
```

```
data %>% group_by(Sepal.Length > 5, Species)
```

```
## # A tibble: 150 x 6
## # Groups:   Sepal.Length > 5, Species [6]
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species `Sepal.Length > 5`
##           <dbl>       <dbl>        <dbl>       <dbl> <fct>   <lgl>
```

```
##  1          5.1         3.5          1.4         0.2 setosa  TRUE
##  2          4.9         3            1.4         0.2 setosa  FALSE
##  3          4.7         3.2          1.3         0.2 setosa  FALSE
##  4          4.6         3.1          1.5         0.2 setosa  FALSE
##  5          5           3.6          1.4         0.2 setosa  FALSE
##  6          5.4         3.9          1.7         0.4 setosa  TRUE
##  7          4.6         3.4          1.4         0.3 setosa  FALSE
##  8          5           3.4          1.5         0.2 setosa  FALSE
##  9          4.4         2.9          1.4         0.2 setosa  FALSE
## 10          4.9         3.1          1.5         0.1 setosa  FALSE
## # i 140 more rows
```

Summarize: apply functions to variables

```
data %>% summarize(mean(Sepal.Length))
```

```
## # A tibble: 1 x 1
##   `mean(Sepal.Length)`
##                  <dbl>
## 1                 5.84
```

```
data %>% summarize(mean(Sepal.Length), sd(Sepal.Length))
```

```
## # A tibble: 1 x 2
##   `mean(Sepal.Length)` `sd(Sepal.Length)`
##                  <dbl>              <dbl>
## 1                 5.84              0.828
```

Combining Summarize and Group_by

```
data %>% summarize(mean(Sepal.Length))
```

```
## # A tibble: 1 x 1
##   `mean(Sepal.Length)`
##                  <dbl>
## 1                 5.84
```

```
data %>% group_by(Species) %>%
  summarize(mean(Sepal.Length))
```

```
## # A tibble: 3 x 2
##   Species    `mean(Sepal.Length)`
##   <fct>                     <dbl>
## 1 setosa                     5.01
## 2 versicolor                 5.94
## 3 virginica                  6.59
```

```
data %>% group_by(Sepal.Length > 5, Species) %>%
  summarize(mean(Sepal.Length), sd(Sepal.Length))
```

```
## `summarise()` has grouped output by 'Sepal.Length > 5'. You can override using
## the `.groups` argument.
```

```
## # A tibble: 6 x 4
## # Groups:   Sepal.Length > 5 [2]
##   `Sepal.Length > 5` Species    `mean(Sepal.Length)` `sd(Sepal.Length)`
##   <lgl>              <fct>                     <dbl>              <dbl>
## 1 FALSE              setosa                     4.76             0.221
## 2 FALSE              versicolor                 4.97             0.0577
```

```
## 3 FALSE             virginica                 4.9            NA
## 4 TRUE              setosa                    5.31            0.223
## 5 TRUE              versicolor                6.00            0.467
## 6 TRUE              virginica                 6.62            0.593
```

One large example

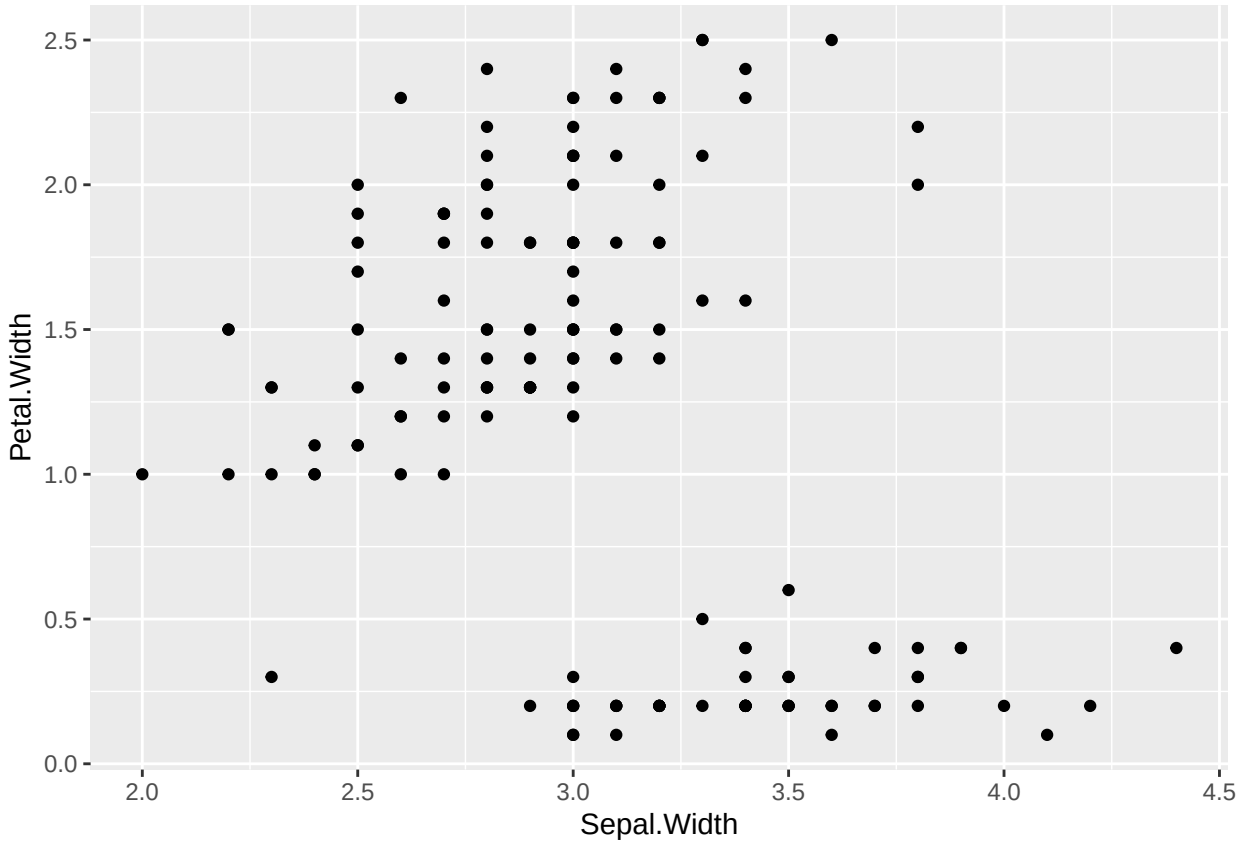
```
data %>% mutate(Sepal.Ratio = Sepal.Length/Sepal.Width, Petal.Ratio = Petal.Length/Petal.Width) %>%
  select(Sepal.Ratio, Petal.Ratio, Species) %>%
  filter(Petal.Ratio > 1, Sepal.Ratio > 1) %>%
  group_by(Species) %>%
  summarize(mean(Sepal.Ratio),mean(Petal.Ratio), sd(Sepal.Ratio), sd(Petal.Ratio)) %>%
  arrange(`mean(Sepal.Ratio)`)
```

```
## # A tibble: 3 x 5
##   Species    `mean(Sepal.Ratio)` `mean(Petal.Ratio)` `sd(Sepal.Ratio)`
##   <fct>                    <dbl>               <dbl>             <dbl>
## 1 setosa                    1.47                6.91             0.119
## 2 versicolor                2.16                3.24             0.229
## 3 virginica                 2.23                2.78             0.247
## # i 1 more variable: `sd(Petal.Ratio)` <dbl>
```

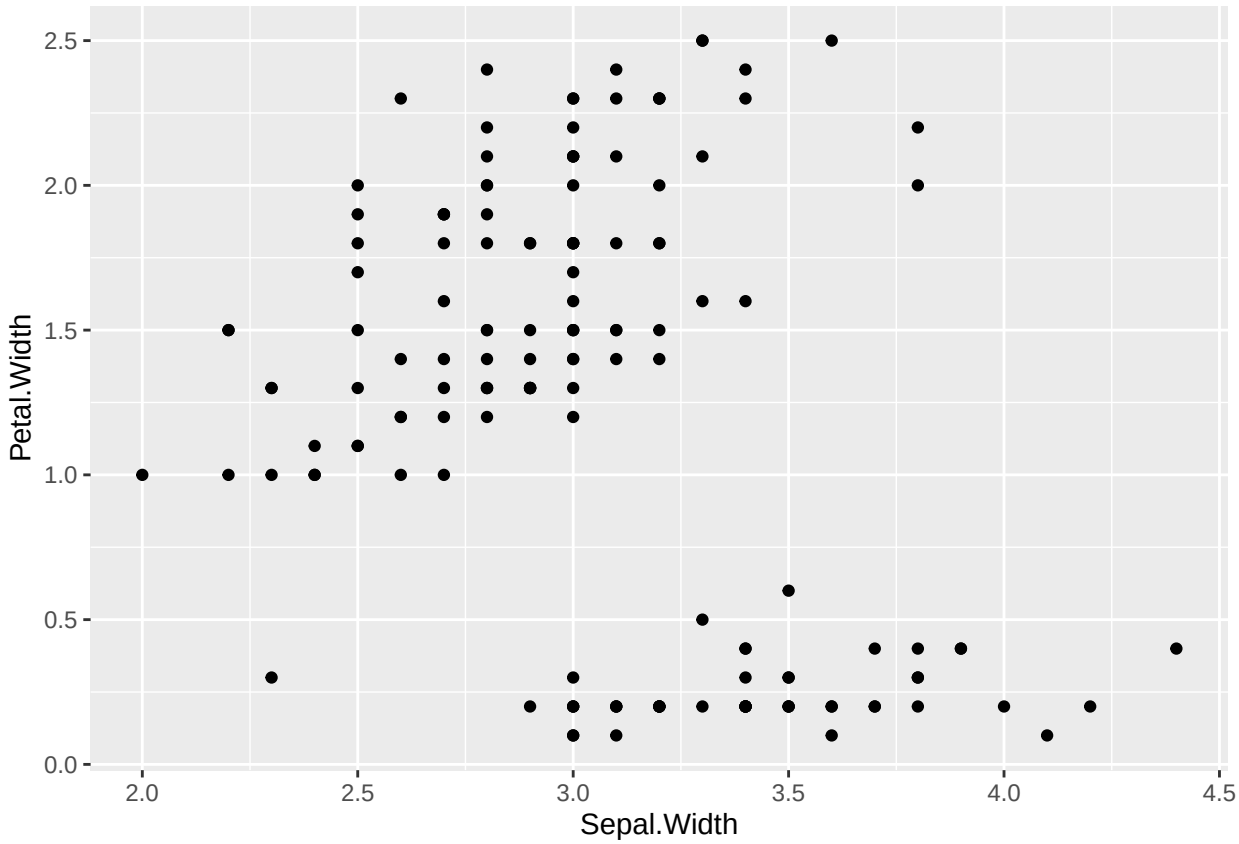
There are many more dplyr sub-functions to use and combinations to discover. I reccomend experimenting, reviewing cheat sheets, and consulting stack-overflow if stuck.

GGplot: tidyverse plotting functions

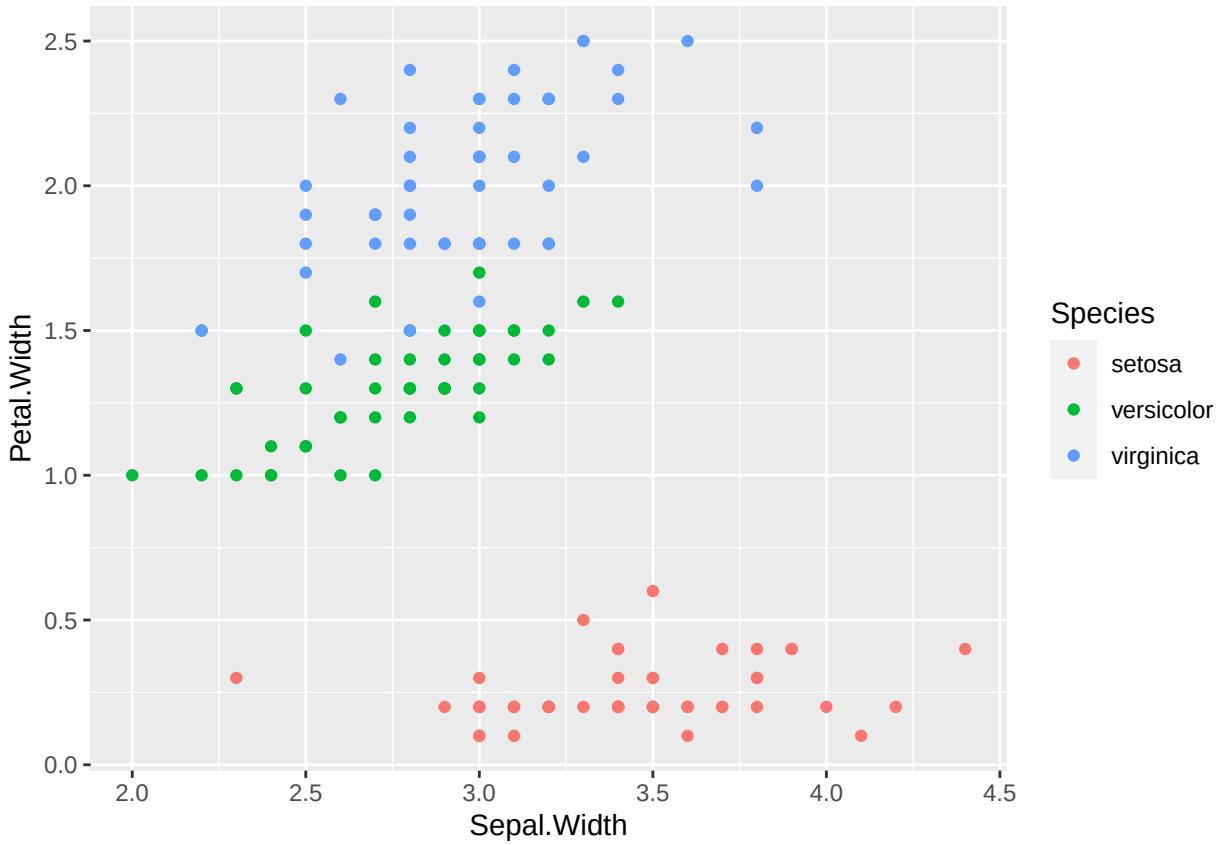
```
data %>% ggplot() +
  geom_point(aes(x=Sepal.Width, y=Petal.Width))
```



```r
#Or equivalenty
data %>% ggplot(aes(x=Sepal.Width, y=Petal.Width)) +
  geom_point()
```

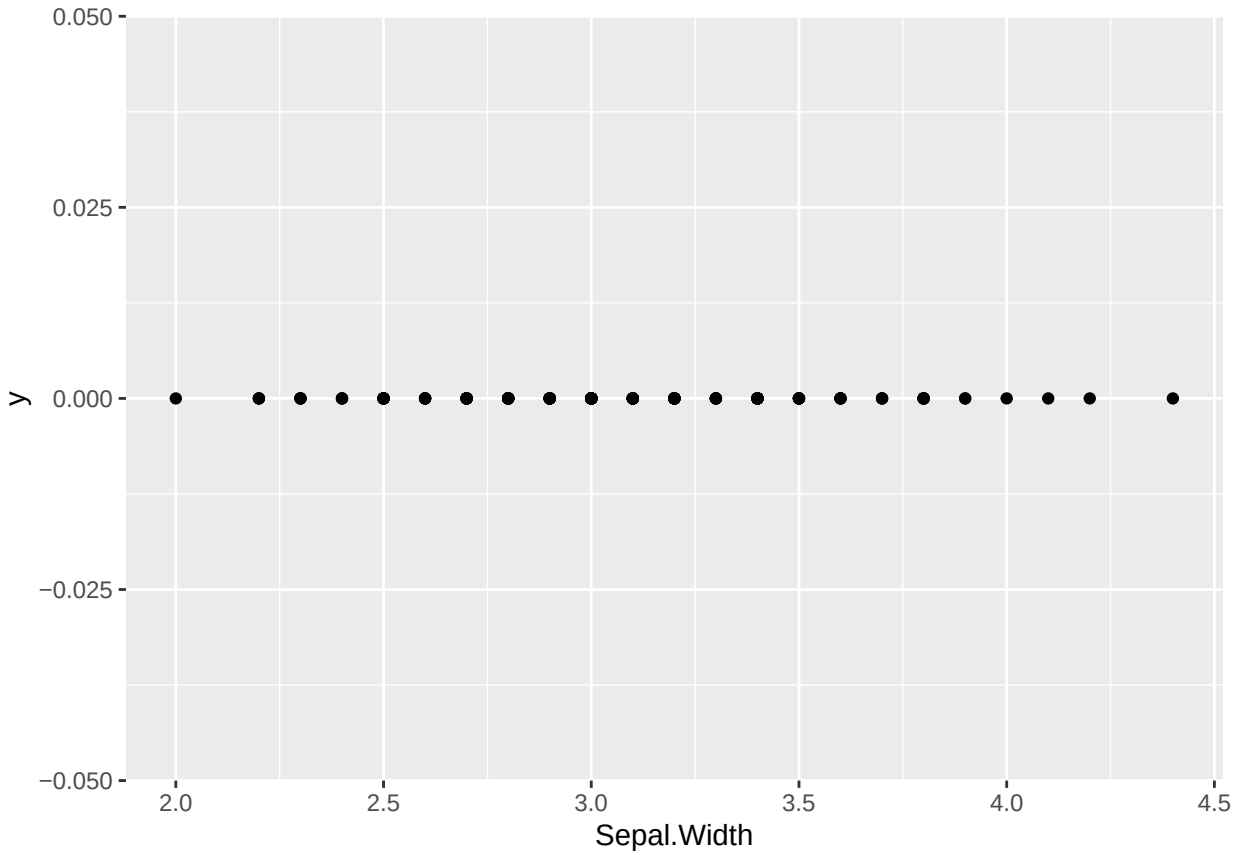



```r
ggplot(data, aes(x=Sepal.Width, y=Petal.Width, color = Species)) +
  geom_point()
```

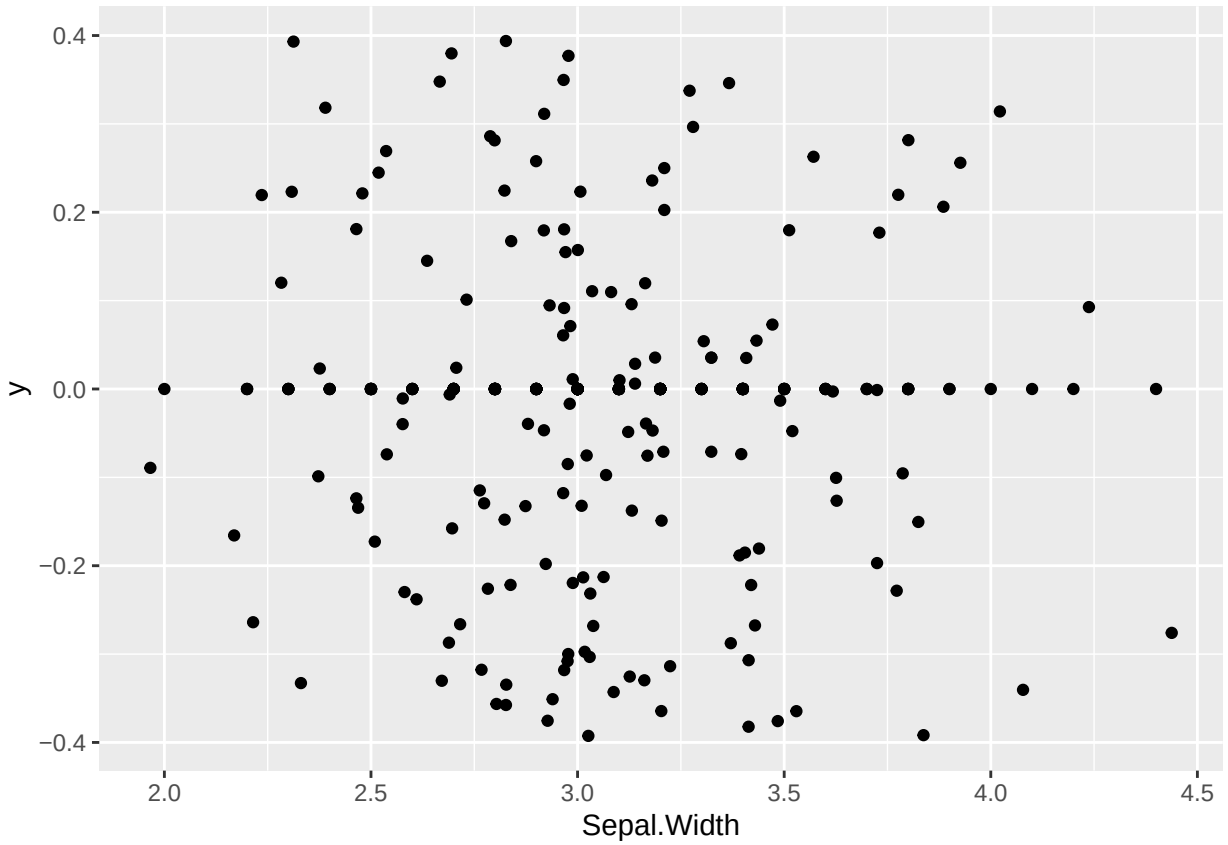


Jittering

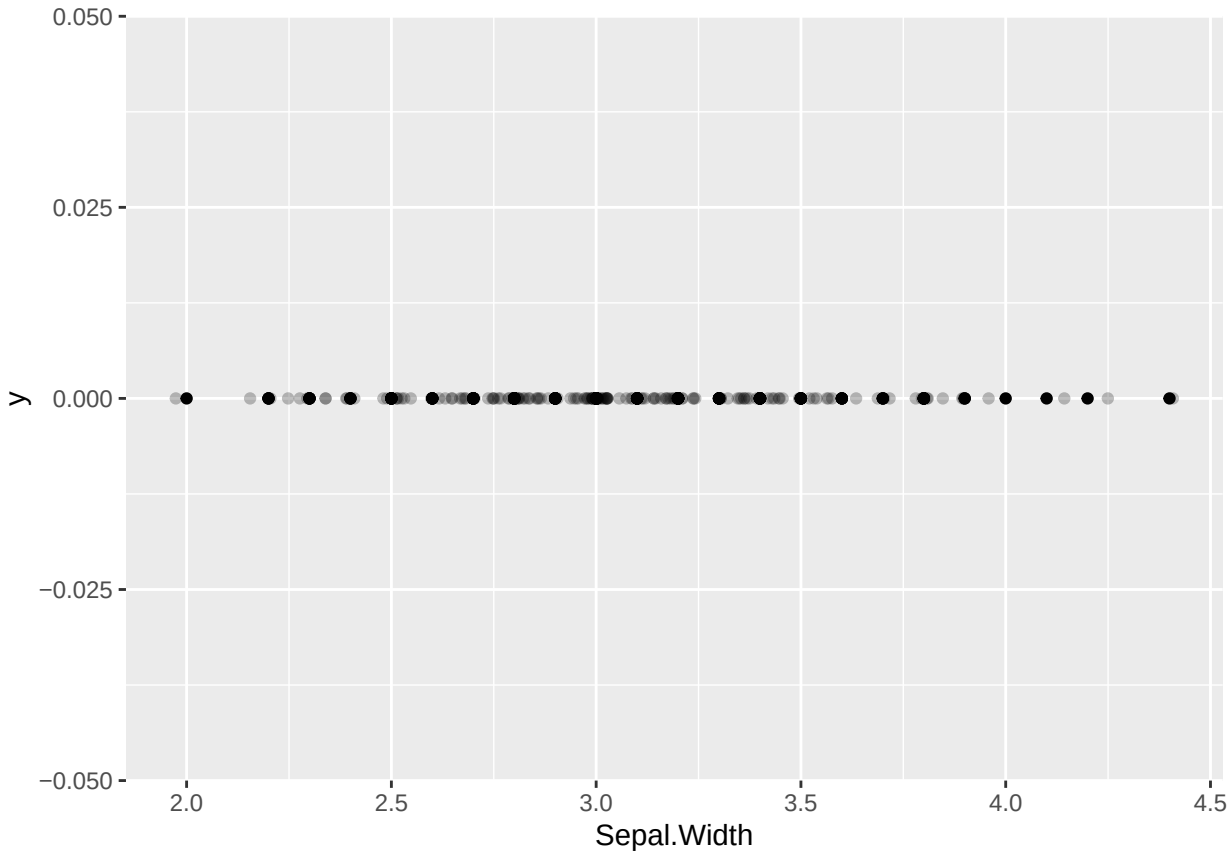
```
data %>% ggplot(aes(x=Sepal.Width, y=0)) +
  geom_point()
```




```
data %>% ggplot(aes(x=Sepal.Width, y=0)) +
  geom_point() +
  geom_jitter()
```

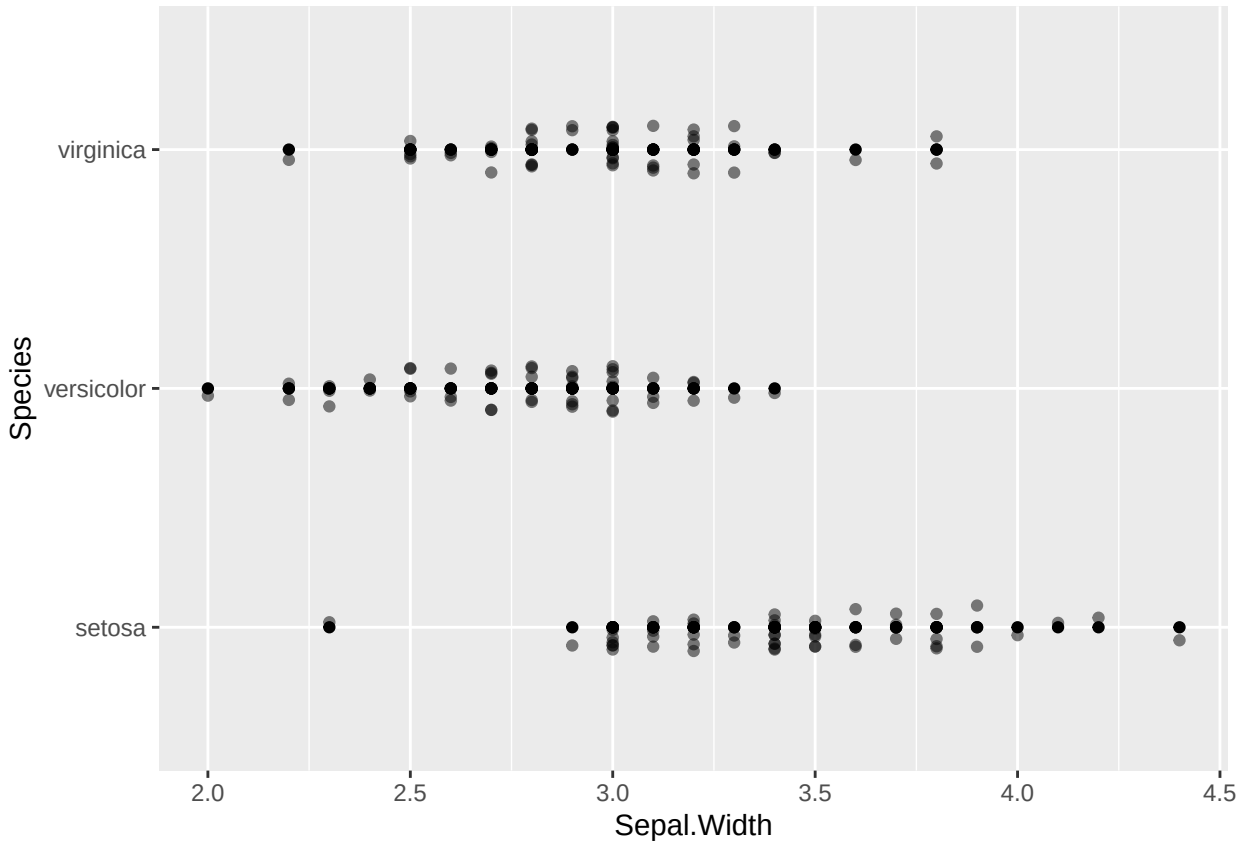



```
data %>% ggplot(aes(x=Sepal.Width, y=0)) +
  geom_point() +
  geom_jitter(width=.05, height=0, alpha=.25)
```

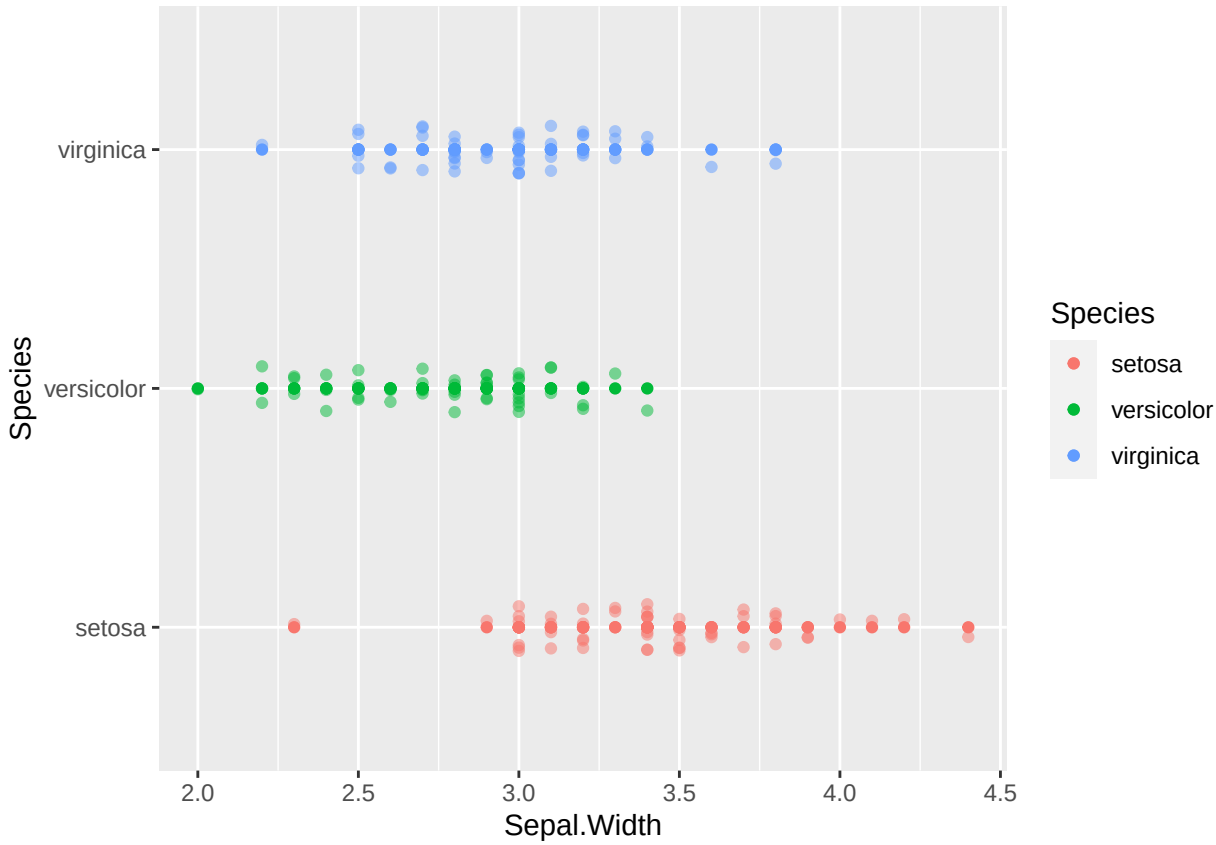


Catagorical plotting

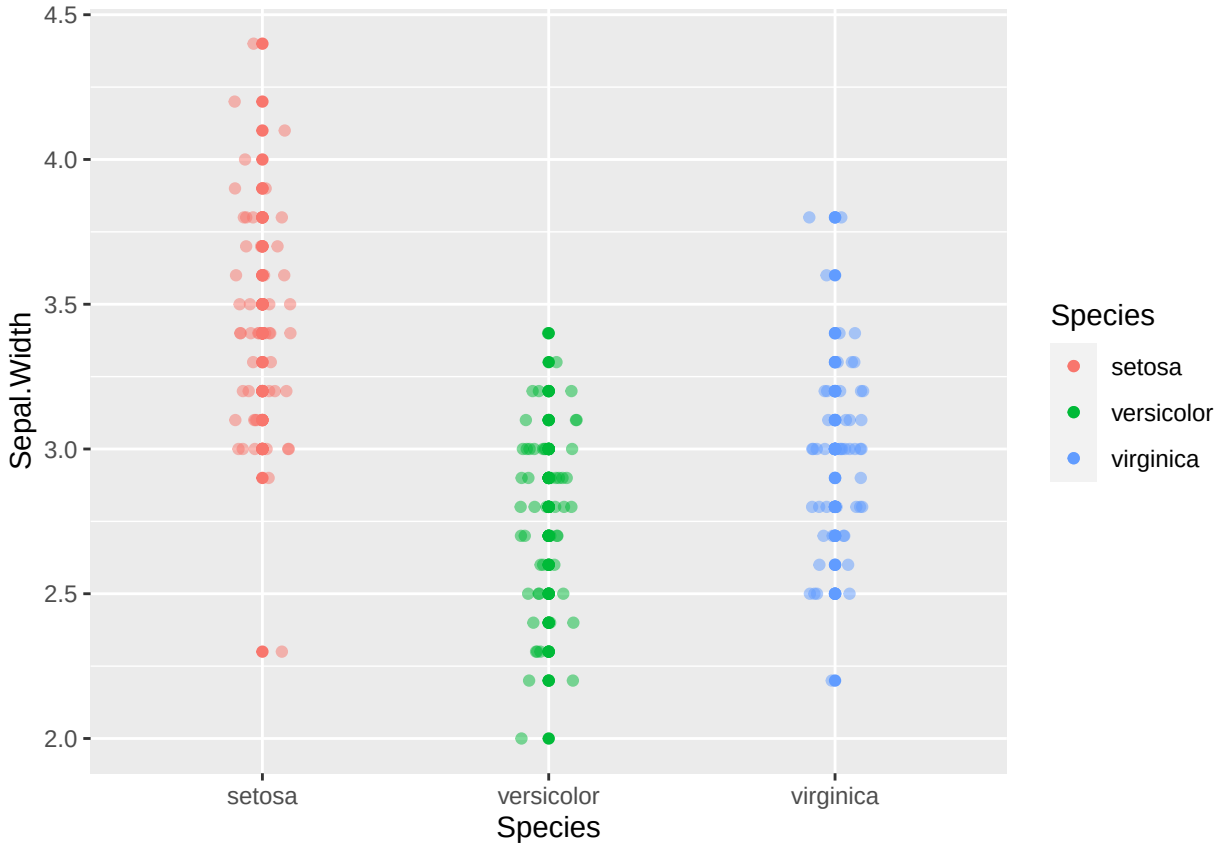
```
data %>% ggplot(aes(x=Sepal.Width, y=Species)) +
  geom_point() +
  geom_jitter(width = 0, height=.1, alpha=.5)
```




```
data %>% ggplot(aes(x=Sepal.Width, y=Species, colour=Species)) +
  geom_point() +
  geom_jitter(width = 0, height=.1, alpha=.5)
```



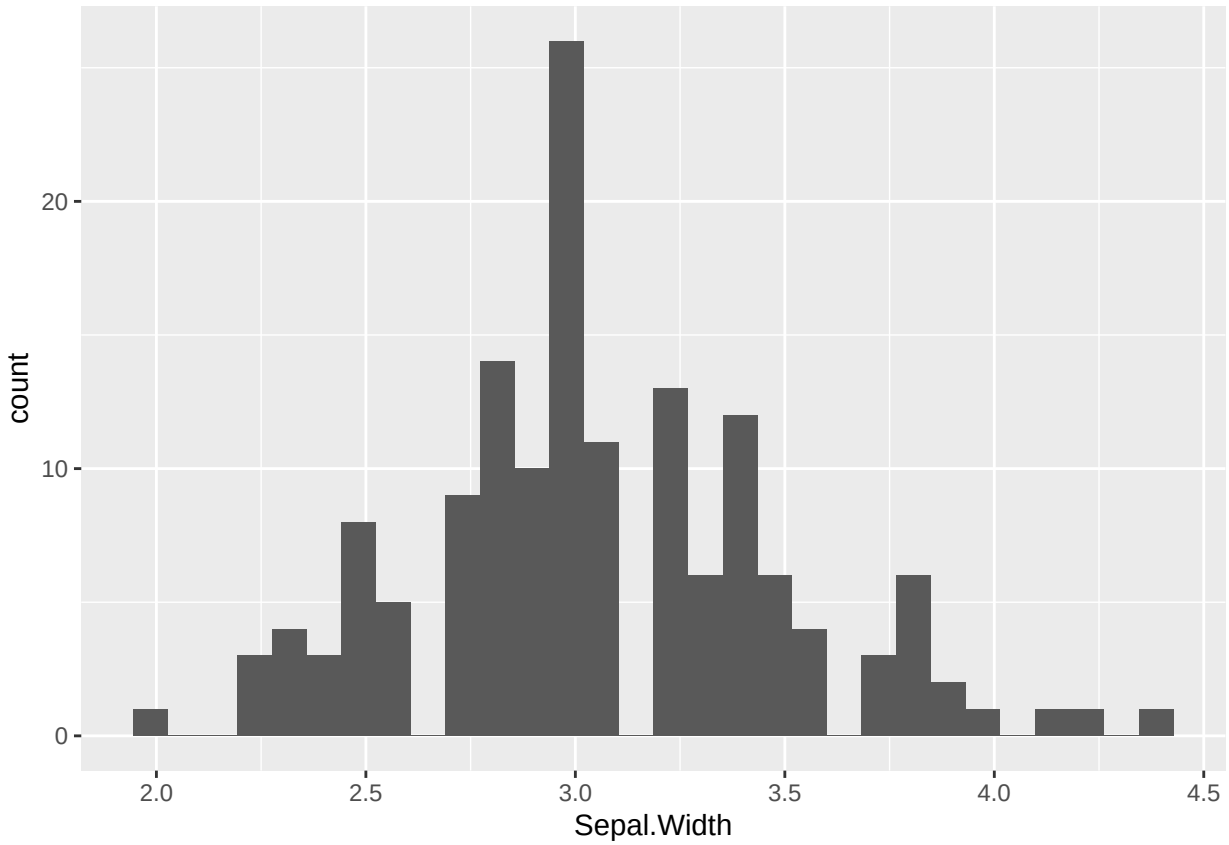

```
data %>% ggplot(aes(x=Sepal.Width, y=Species, colour=Species)) +
  geom_point() +
  geom_jitter(width = 0, height=.1, alpha=.5) +
  coord_flip()
```



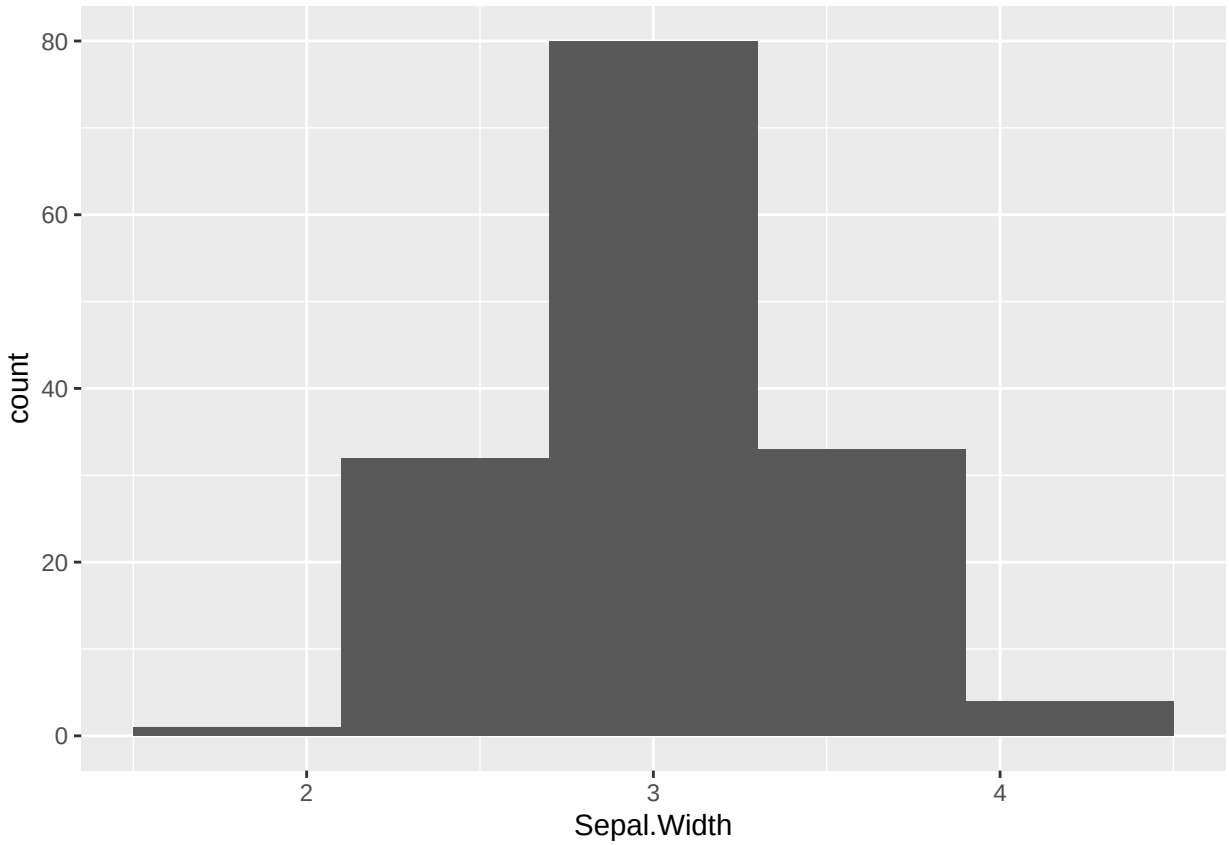
Histograms

```
data %>% ggplot(aes(x=Sepal.Width)) +
  geom_histogram()
```

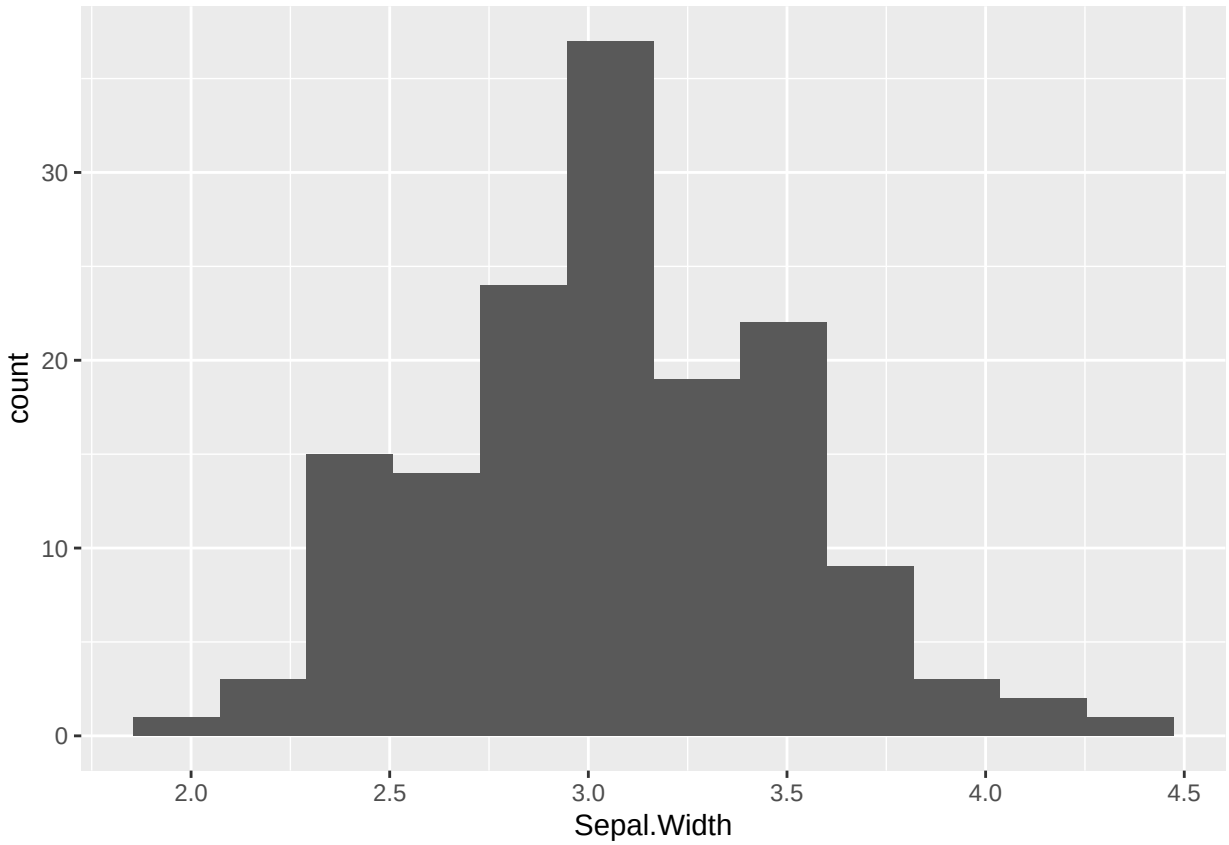
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



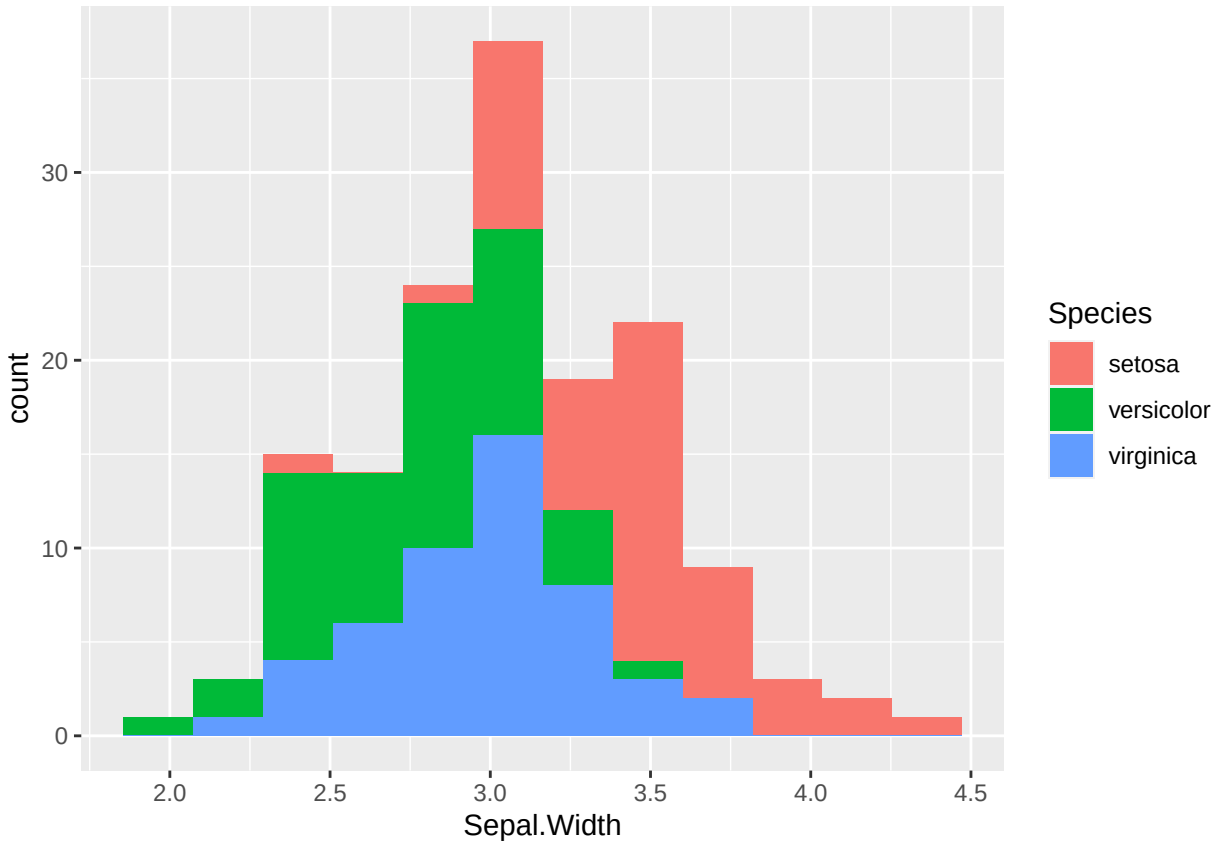

```
data %>% ggplot(aes(x=Sepal.Width)) +
  geom_histogram(bins=5)
```



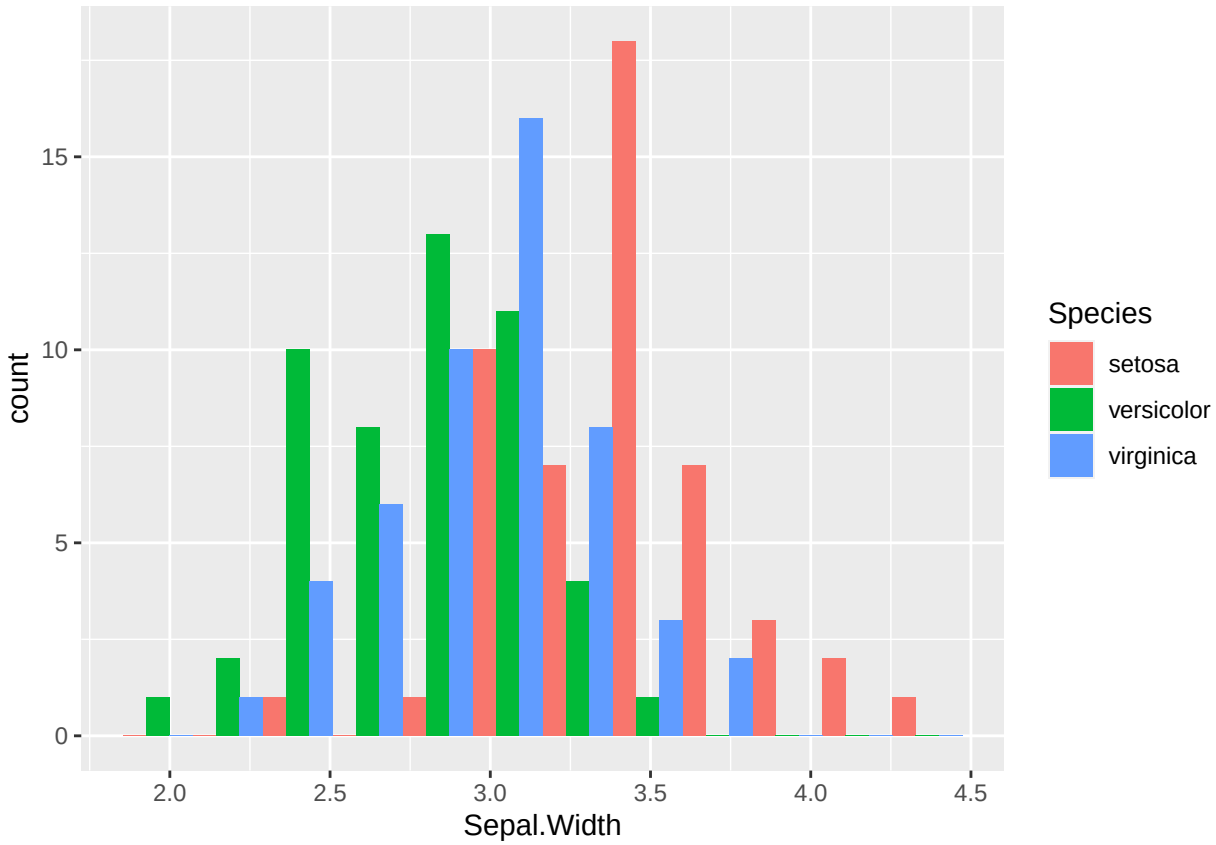

```
data %>% ggplot(aes(x=Sepal.Width)) +
  geom_histogram(bins=12)
```



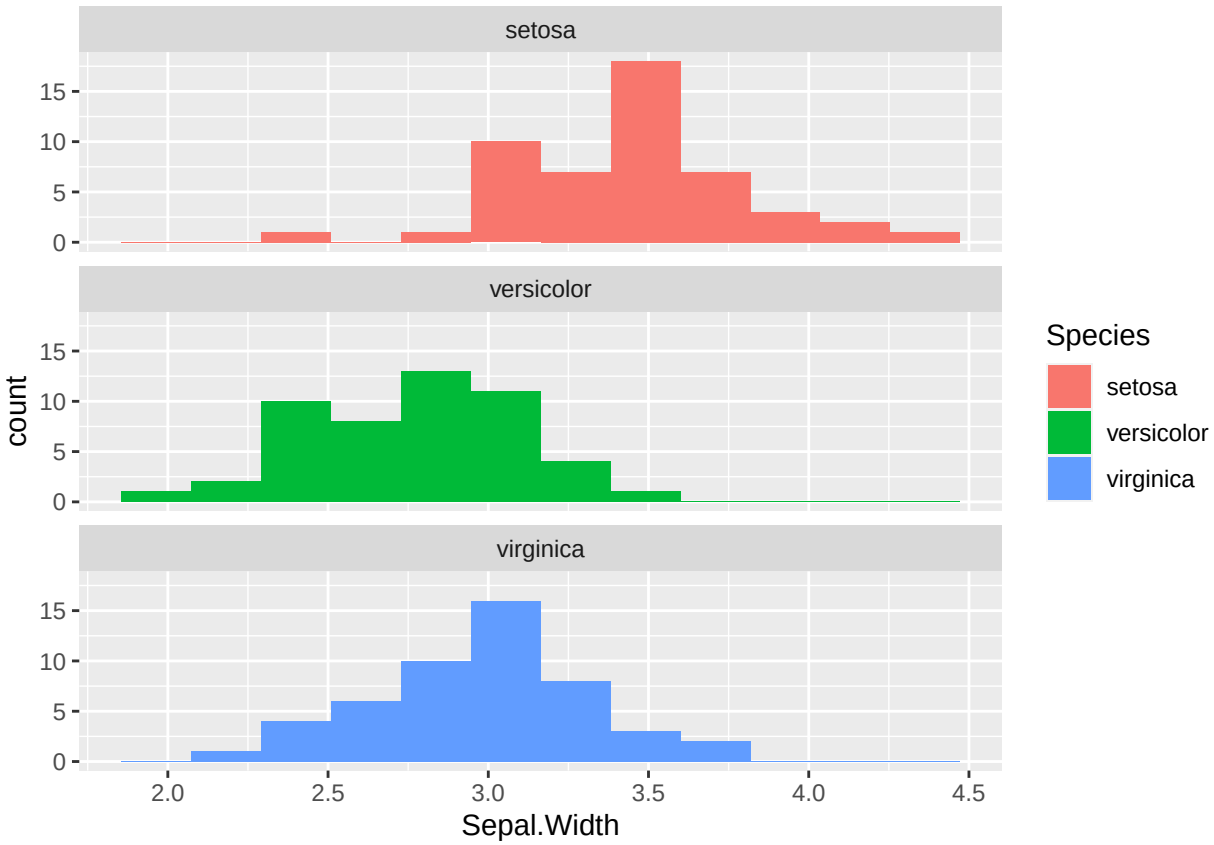

```
data %>% ggplot(aes(x=Sepal.Width, fill=Species)) +
  geom_histogram(bins=12)
```




```
data %>% ggplot(aes(x=Sepal.Width, fill=Species)) +
  geom_histogram(bins=12, position="dodge")
```

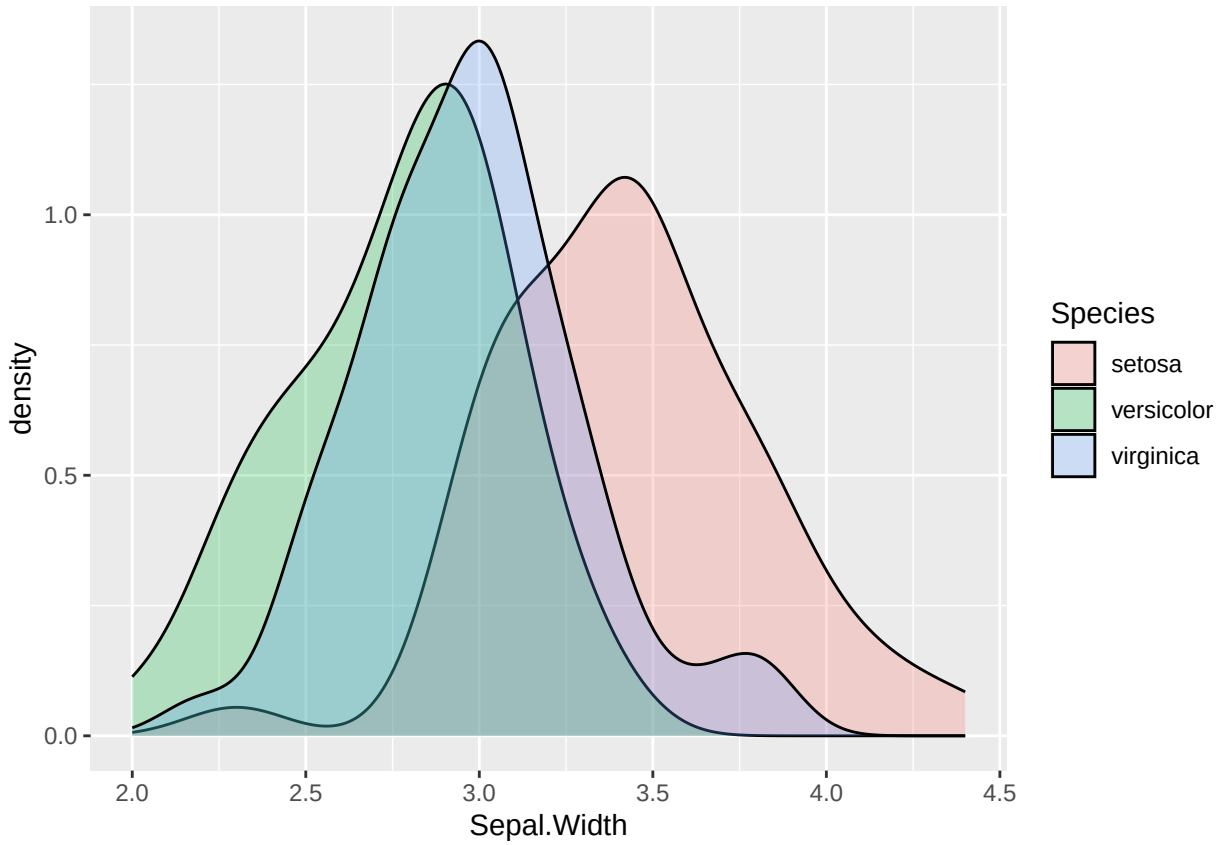



```
data %>% ggplot(aes(x=Sepal.Width, fill=Species)) +
  geom_histogram(bins=12) +
  facet_wrap(~Species, ncol=1)
```

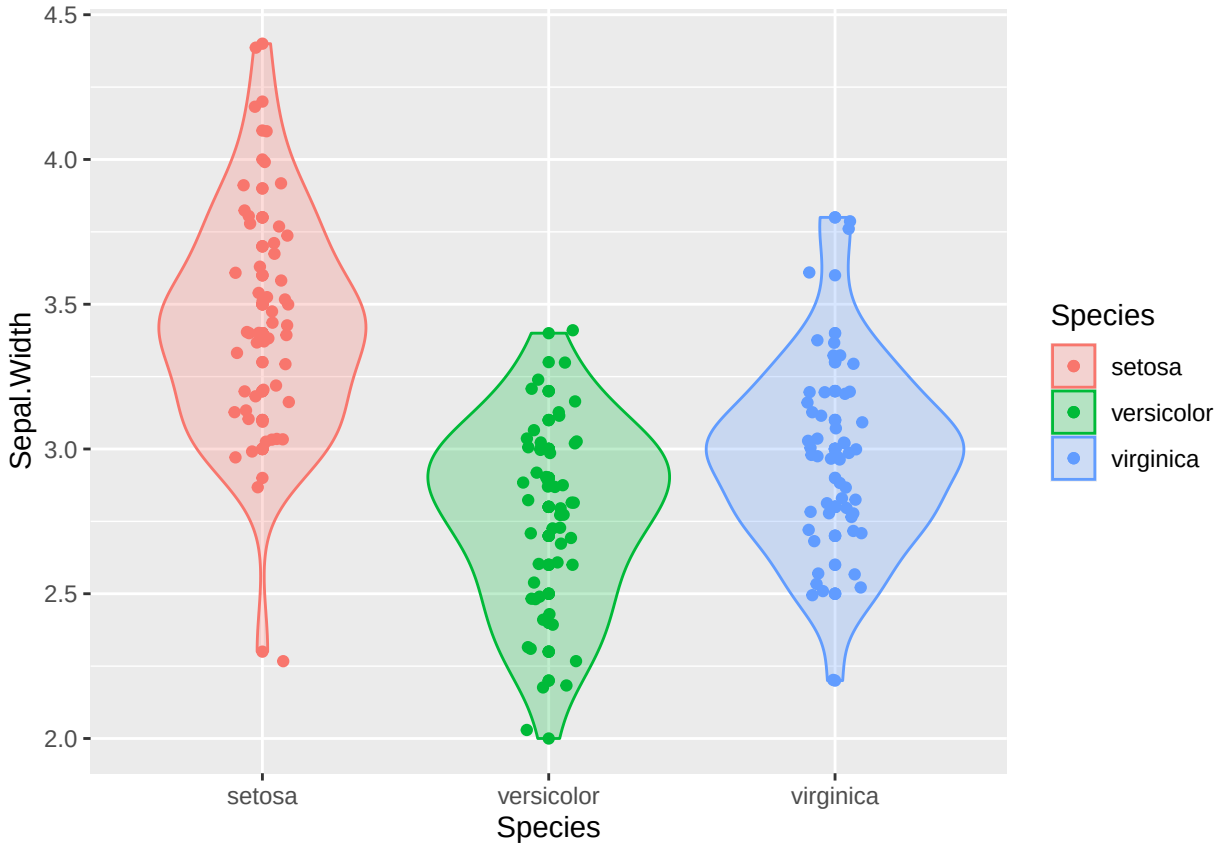


Density and Violin Plots

```
data %>% ggplot(aes(x=Sepal.Width, fill=Species)) +
  geom_density(alpha=.25)
```

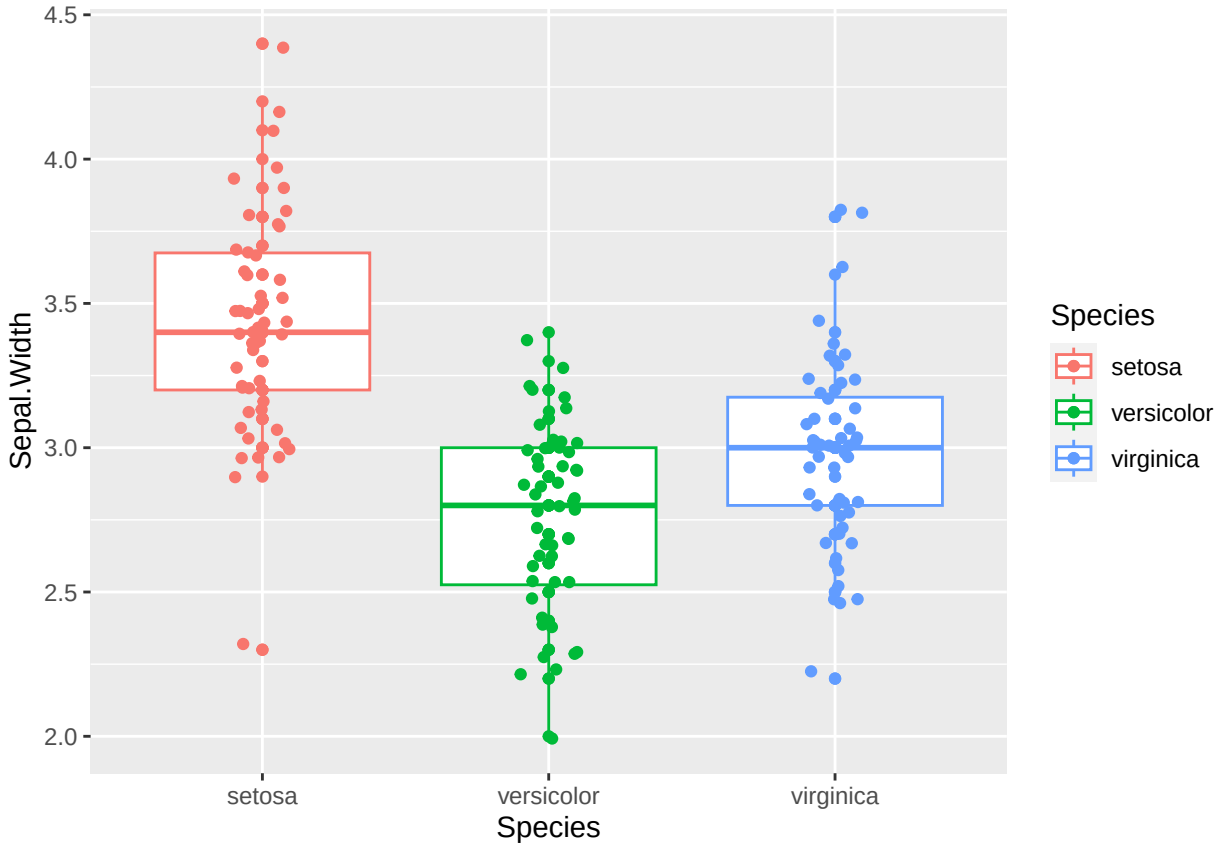



```
data %>% ggplot(aes(x=Species, y=Sepal.Width, colour = Species, fill=Species)) +
  geom_violin(alpha=.25) +
  geom_point() + geom_jitter(width = 0.1)
```



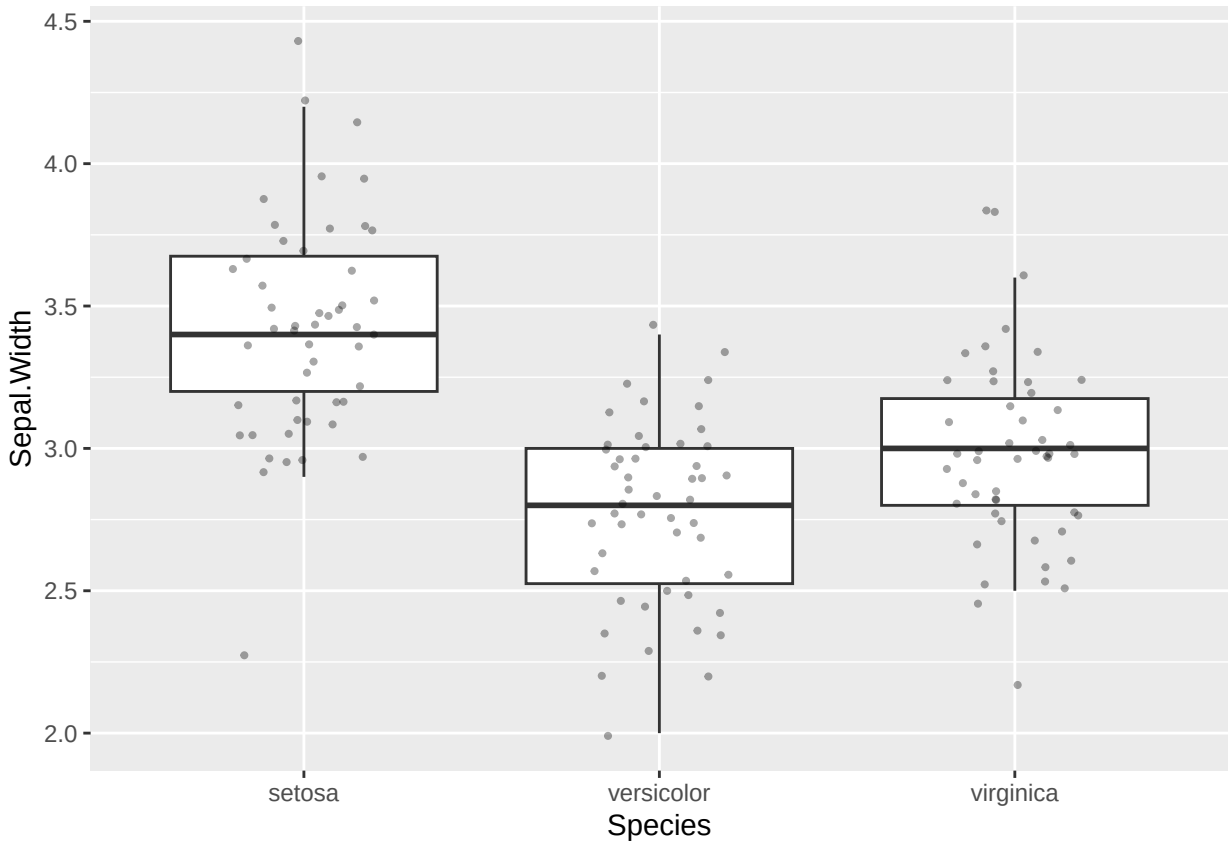
Boxplots

```
data %>% ggplot(aes(x=Species, y=Sepal.Width, colour=Species)) +
  geom_boxplot()+
  geom_point() + geom_jitter(width = 0.1)
```

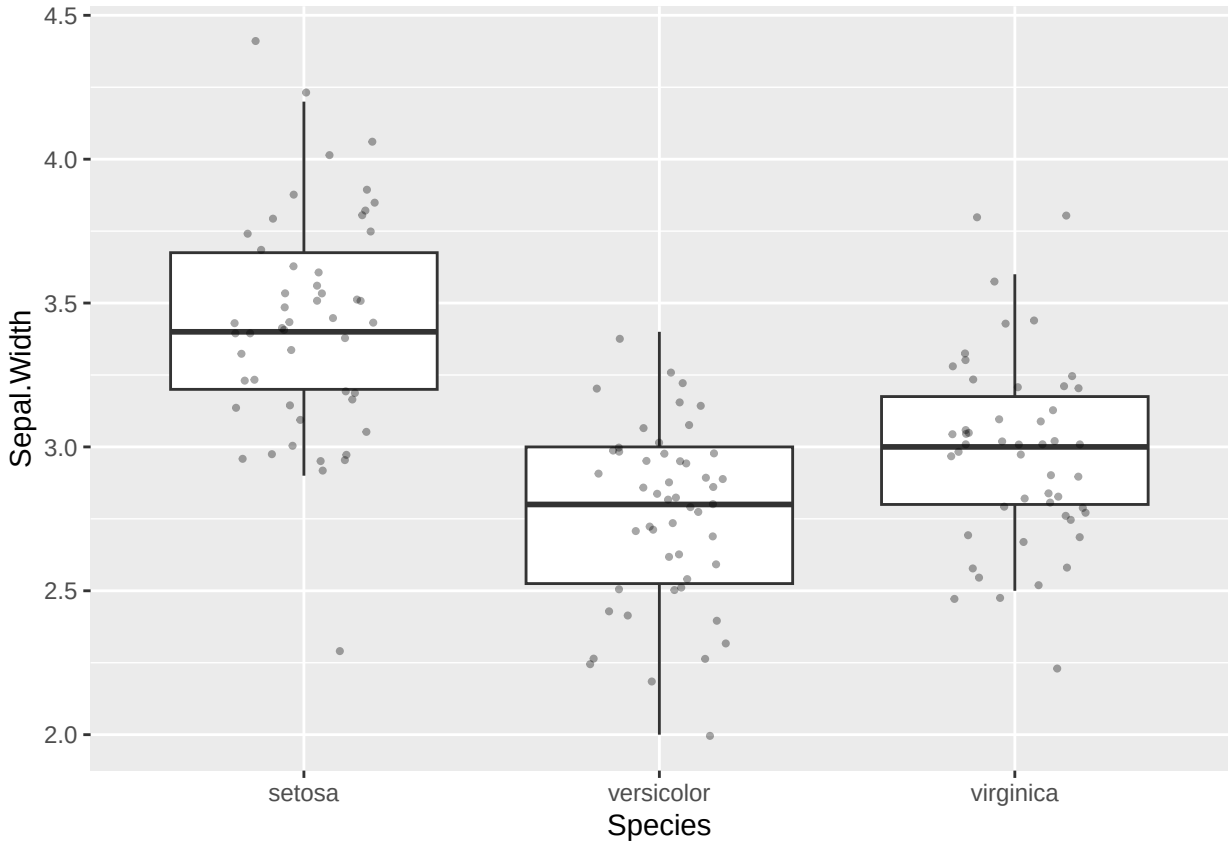


Combination Plots

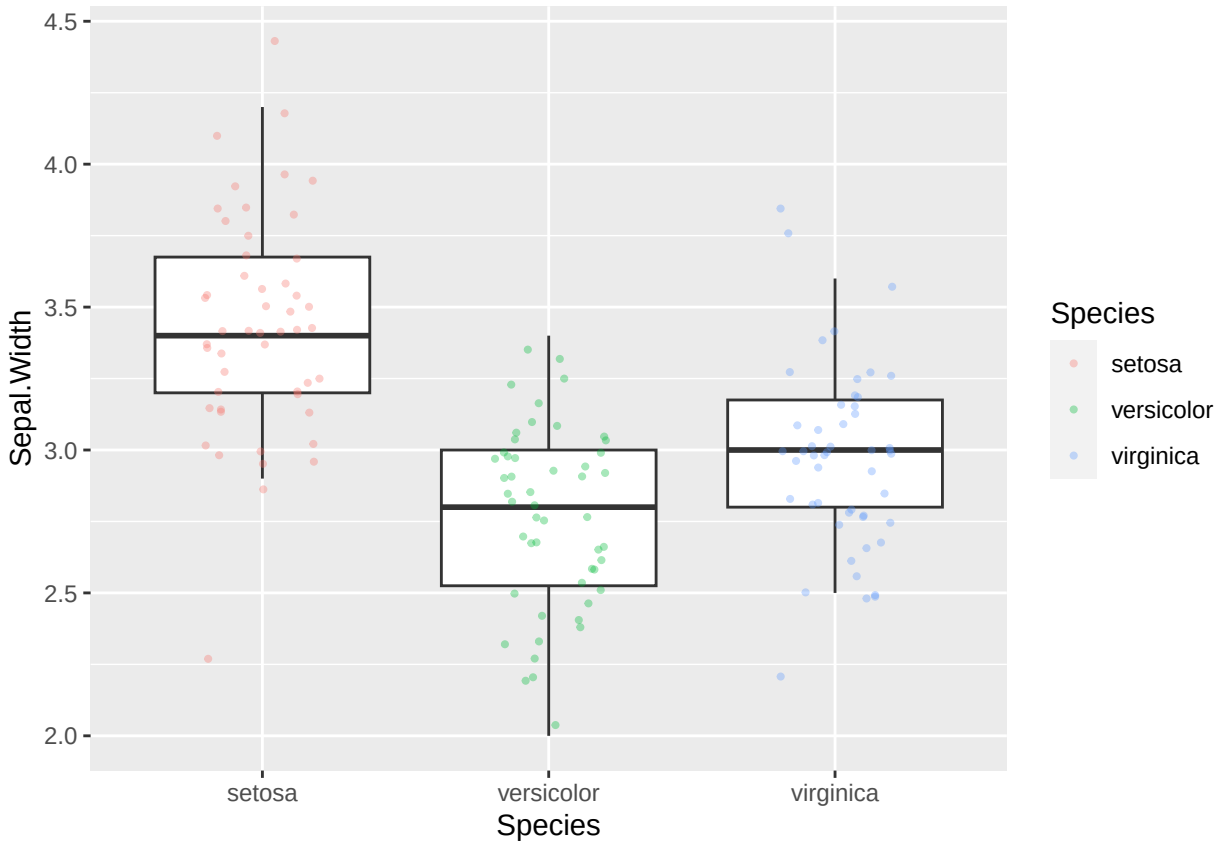
```
data %>% ggplot(aes(x=Species, y=Sepal.Width)) +
  geom_boxplot(outlier.shape=NA) +
  geom_jitter(width=.2, height=.05, alpha=.35, size=.75)
```



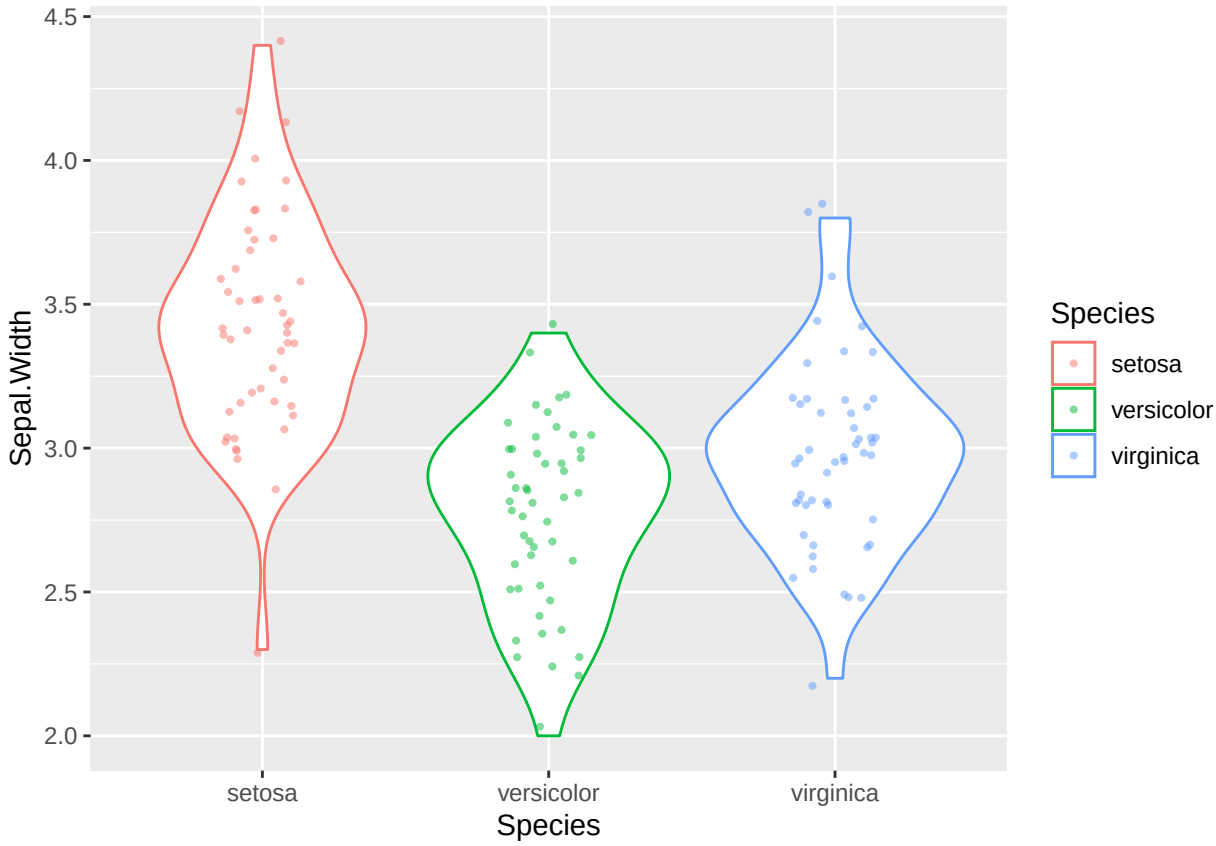

```
#Note that this is equivalent to writing
data %>% ggplot() +
  geom_boxplot(aes(x=Species, y=Sepal.Width), outlier.shape=NA) +
  geom_jitter(aes(x=Species, y=Sepal.Width), width=.2, height=.05, alpha=.35, size=.75)
```




```
#Be mindful of what is shared between plots, for example, we may only want to colour one layer
data %>% ggplot(aes(x=Species, y=Sepal.Width)) +
  geom_boxplot(outlier.shape=NA) +
  geom_jitter(aes(colour=Species),width=.2, height=.05, alpha=.35, size=.75)
```

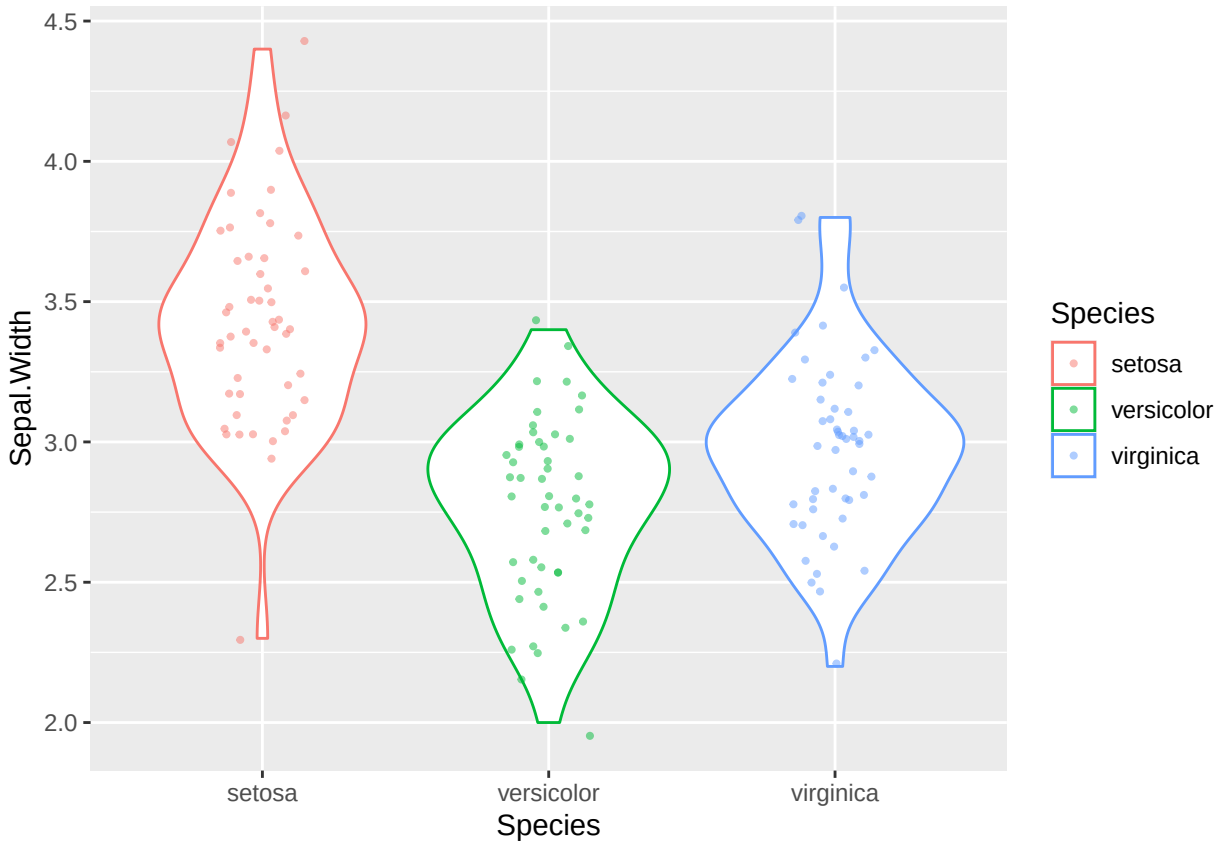



```
#Violin plus jitter plot
data %>% ggplot(aes(x=Species, y=Sepal.Width, color=Species)) +
  geom_violin() +
  geom_jitter(width=.15, height=.05, alpha=.5, size=.75)
```



Layering and titles

```
#We can save plots and layers as variables
plot = data %>% ggplot(aes(x=Species, y=Sepal.Width, colour=Species))
violin_layer = geom_violin()
jitter_layer = geom_jitter(width=.15, height=.05, alpha=.5, size=.75)

plot+violin_layer+jitter_layer
```




```
plot+violin_layer+jitter_layer +
  labs(x="Species", y="Sepal Width (cm)", title="Sepal Width Distributions")
```

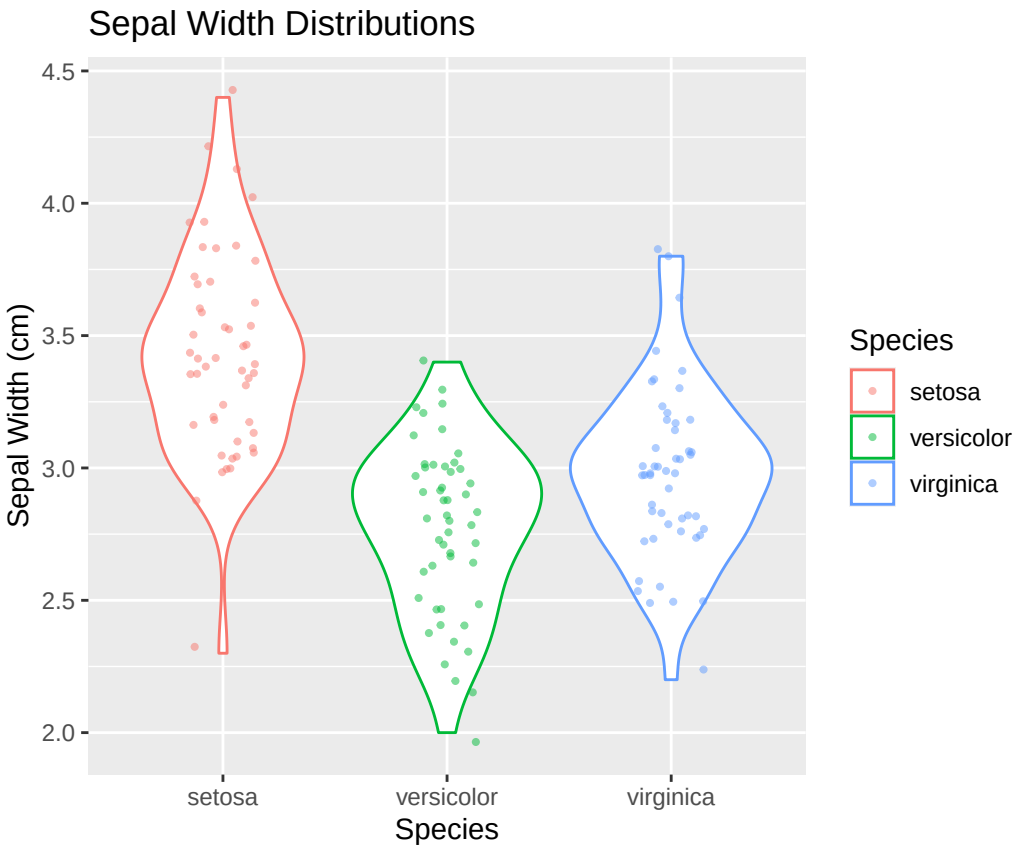
Themes All parts of a ggplot can be tweaked, but there are some themes for overall styles to use

```
full_plot = plot + violin_layer + jitter_layer + labs(x="Species", y="Sepal Width (cm)", title="Sepal W

full_plot
```

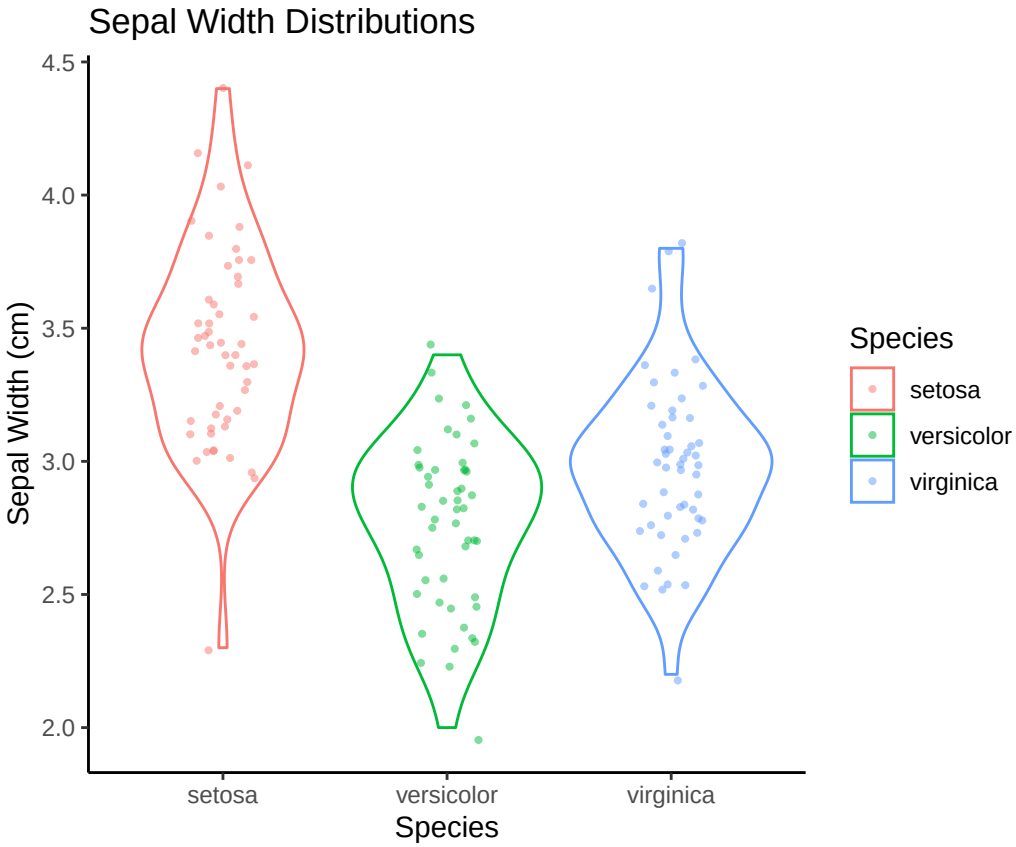

```
full_plot + theme_bw()
```


```
full_plot + theme_classic()
```


```
#You can see more options using RStudio's autocomplete or at https://ggplot2.tidyverse.org/reference/gg

full_plot + theme(aspect.ratio=1)
```




```
full_plot + theme_classic() + theme(aspect.ratio=1)
```



This is just a small peak at some of the things you can do with ggplot2. There are lots of additional plots to use, graphical options to tweak, and display options to add. Extensive documentation is located on the ggplot website, ggplot cheat sheets, and stack overflow.