Data visualization in R Using ggplot2

Surya Lamichhane

01 March 2025

Contents

By the end of this lesson, you will be able to:

- Visualize data using ggplot function from ggplot2 package
- Describe and Analyze date using ggplot functions

Part 2. (Data vizulization in R using ggplot)

ggplot2 is one of the most used packages for data visualization in R and it builds plots in layers.

ggplot2 builds graphs in layers. It divides the plot into three parts:

- Data: It is the dataframe that contains data to be plotted.
- Aesthetics: These are the variables mapping to the visual properties of the plot.
- Geometry: It refers to the type of graph that is used, such as bar graph or histogram
- Optional parts:
  - XY- axex label
  - theme

For more details see the **link** [http://r-statistics.co/Top50-Ggplot2-Visualizations-MasterList-R-Code.html]

- **syntax**

```
ggplot(data) +
aes(x, y) +
geom_..() + # geometry type such point, hist, bar
optional_layer # such as theme, xy-labels

-OR- (using dplyr pyping)
data  %>%
ggplot()  %>%
aes(x, y) +
geom_..()
```

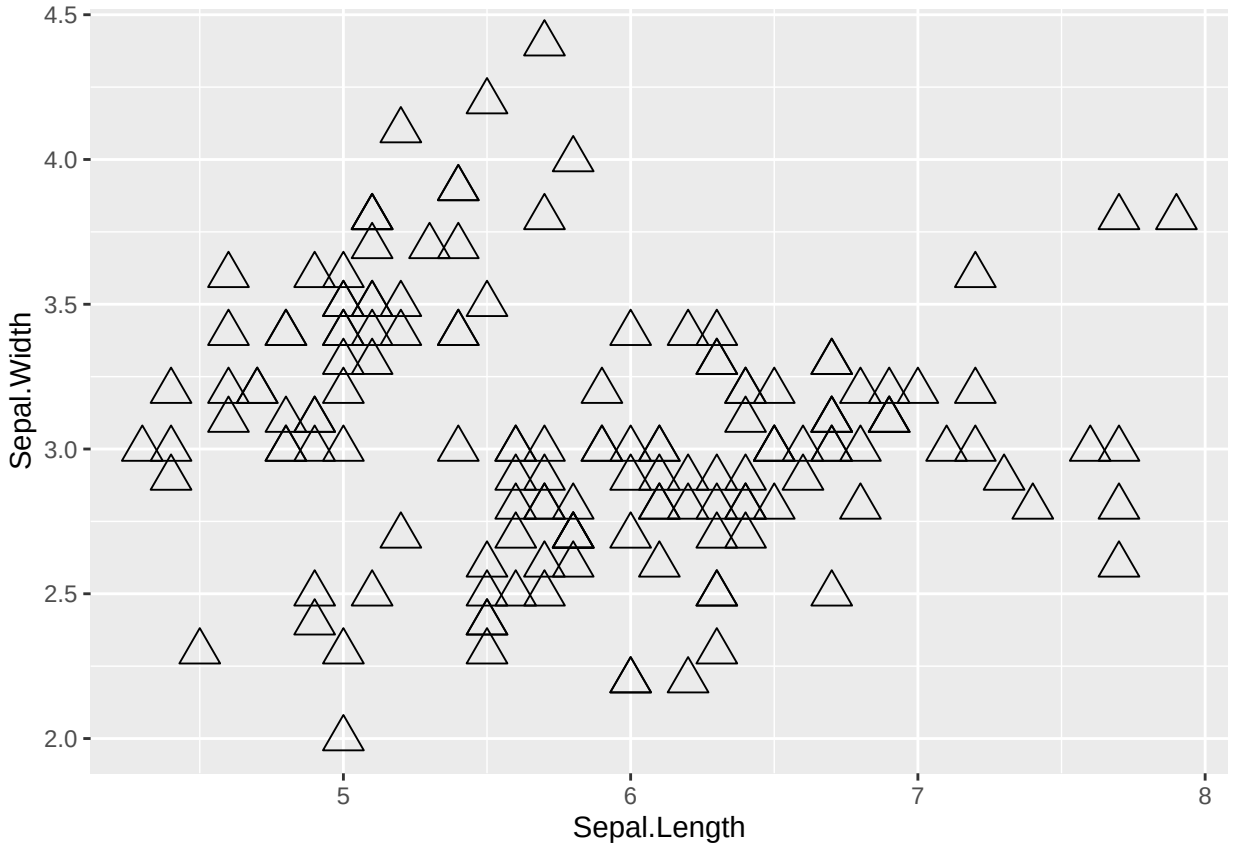
2.1 Scatter plot

```
# install.packages("ggplot2")
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
```

```
iris %>%
ggplot() +
  aes(x = Sepal.Length, y = Sepal.Width) +
  #geom_point(aes(size=Sepal.Length)) # size of points varies as values
  geom_point(size=5, shape=2)
```



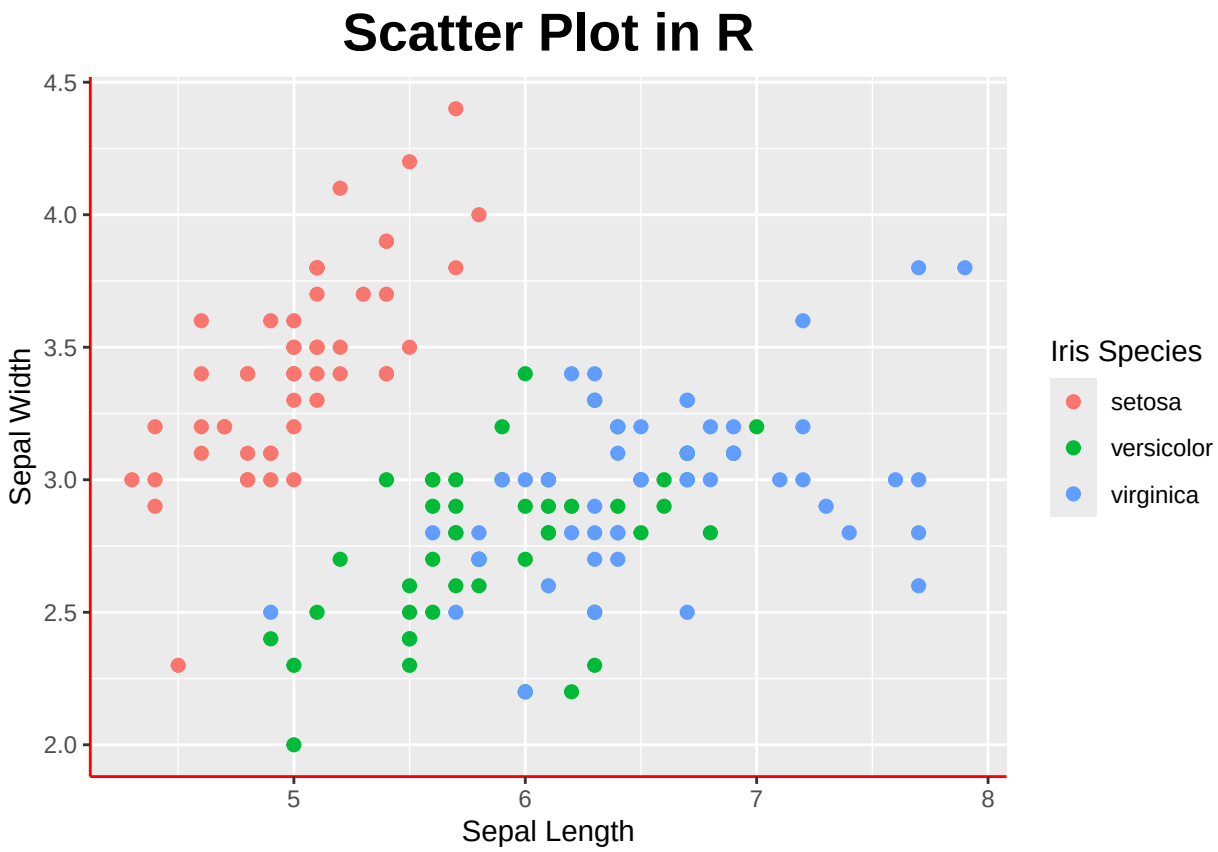
2.1.1. Label in the scatter plot

```
ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species)) + # Points color by Species
  geom_point() +
  xlab("Sepal Length") +               # X-axis label
  ylab("Sepal Width")  +              # Y-axis label
  theme(axis.line = element_line(colour = "black", # Changes the default theme
                                 linewidth =0.5))
```

2.1.2. Plot Title and Change legend name

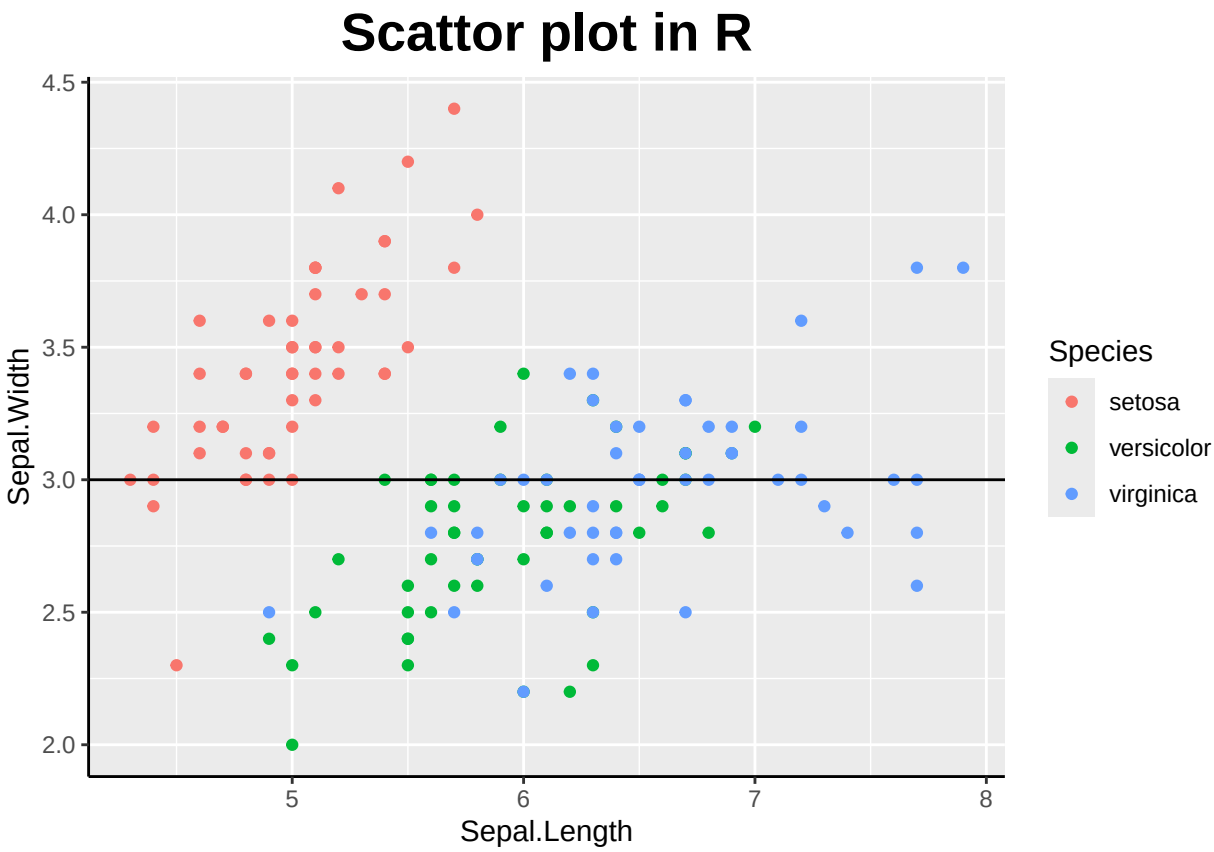
```
library(ggplot2)

ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species)) +
  geom_point(size = 2) +  # Increased point size for better visibility
  ggtitle("Scatter Plot in R") +
  scale_color_discrete(name = "Iris Species") +  # Correct way to rename legend
  xlab("Sepal Length") +
  ylab("Sepal Width") +
  theme(
    axis.line = element_line(colour = "red", size = 0.5),  # Keep red xy-axes
    plot.title = element_text(hjust = 0.5, size = 20, face = "bold"),  # Centered and bold title
  )
```

```
## Warning: The `size` argument of `element_line()` is deprecated as of ggplot2 3.4.0.
## i Please use the `linewidth` argument instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



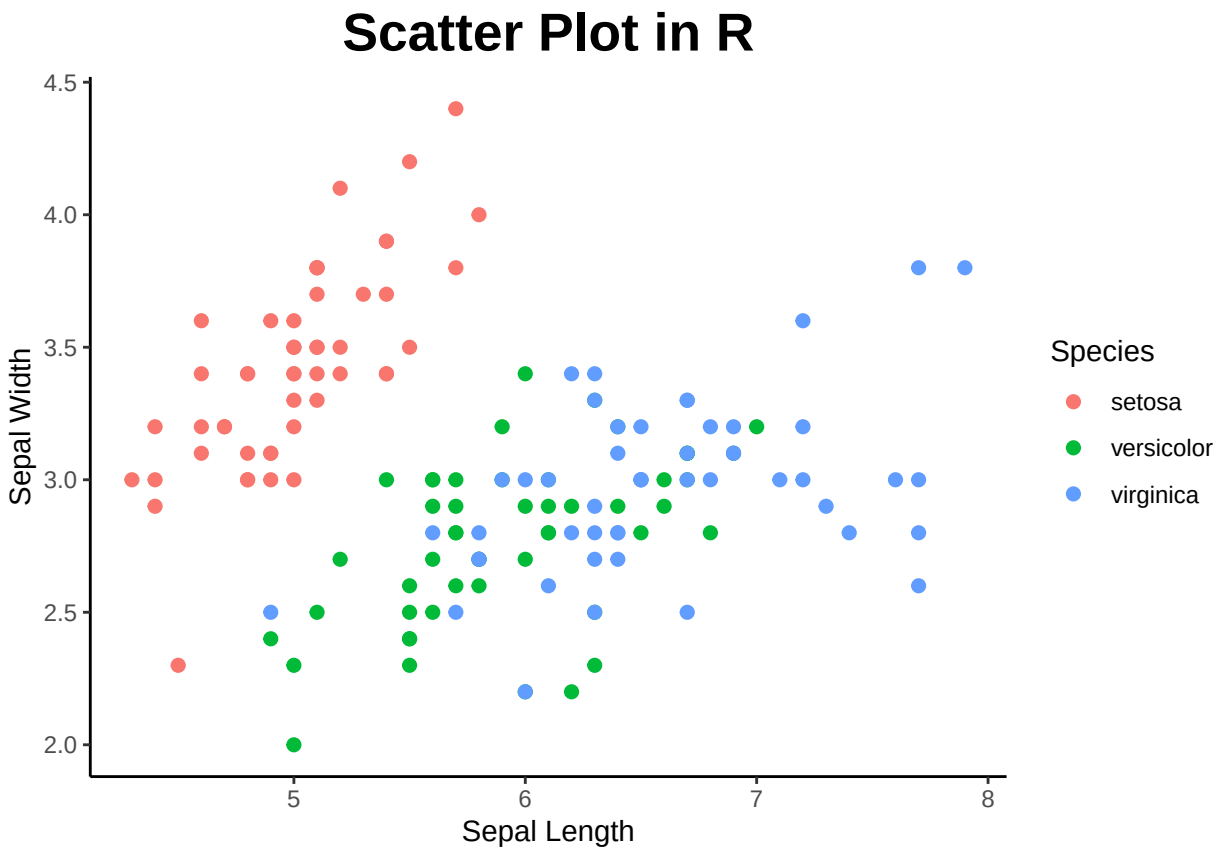
2.1.3. Straigh line using abline()

```
ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species)) +
  geom_point() +
  geom_abline(intercept = 3, slope =  0 ) + # draw a horizontal line
  ggtitle("Scattor plot in R") +
  scale_color_discrete(name = "Species") +  # Legend title
  xlab("Sepal.Length") +              # X-axis label
  ylab("Sepal.Width")  +             # Y-axis label
  theme(axis.line = element_line(color = "black", size = 0.5), # Changes the default theme (xy-axes)
        plot.title = element_text(hjust=0.5, size = 20, face = "bold")# Assign title on center
)
```



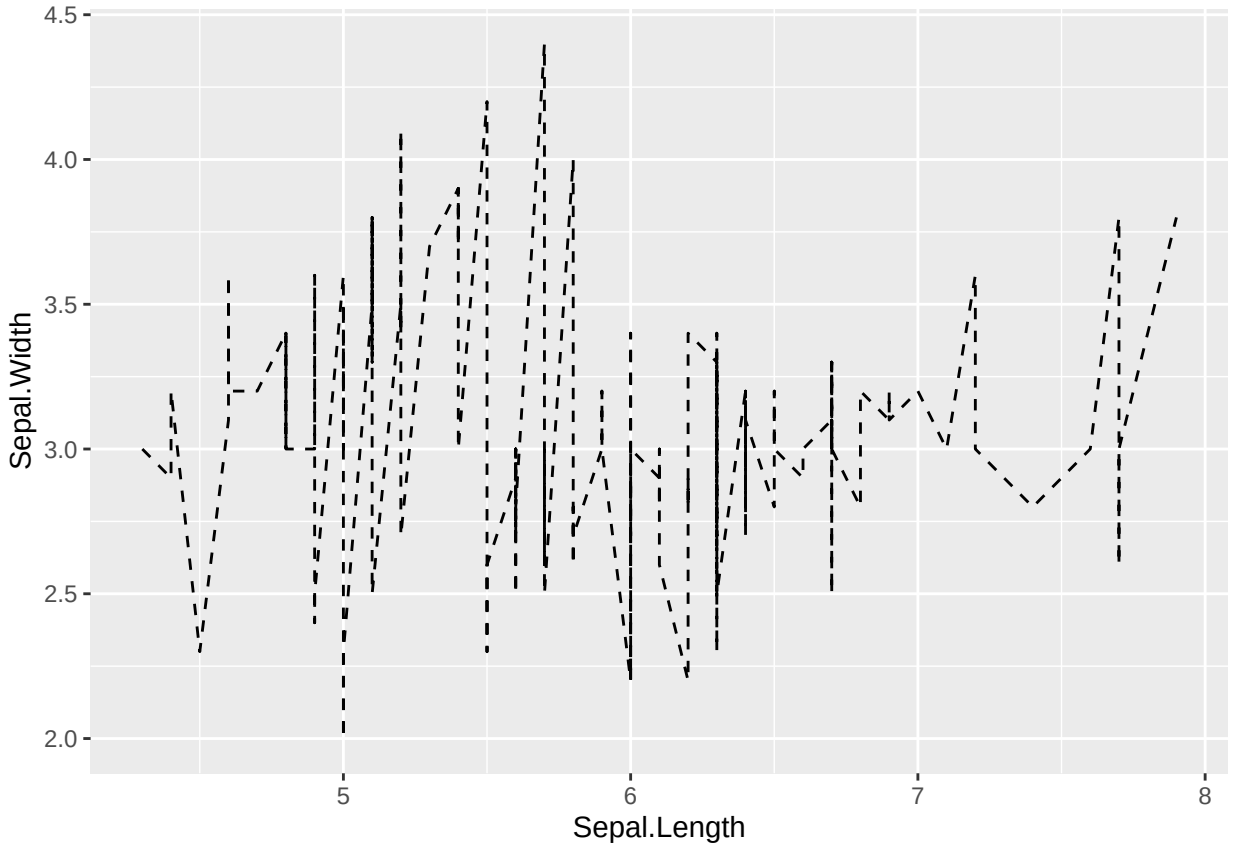
2.1.4. Remove the grids and change background to white

```
ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species)) +
  geom_point(size = 2) +  # Adds scatter points with color by Species
  ggtitle("Scatter Plot in R") +
  scale_color_discrete(name = "Species") +  # Legend title
  xlab("Sepal Length") +
  ylab("Sepal Width") +
  theme(
    axis.line = element_line(colour = "black", size = 0.5),  # Make xy-axes black
    plot.title = element_text(hjust = 0.5, face = "bold", size = 20),  # Center title, bold, increase s
    panel.background = element_rect(fill = "white"),  # White background
    panel.grid.major = element_blank(),  # Remove major grid lines
    panel.grid.minor = element_blank()   # Remove minor grid lines
  )
```

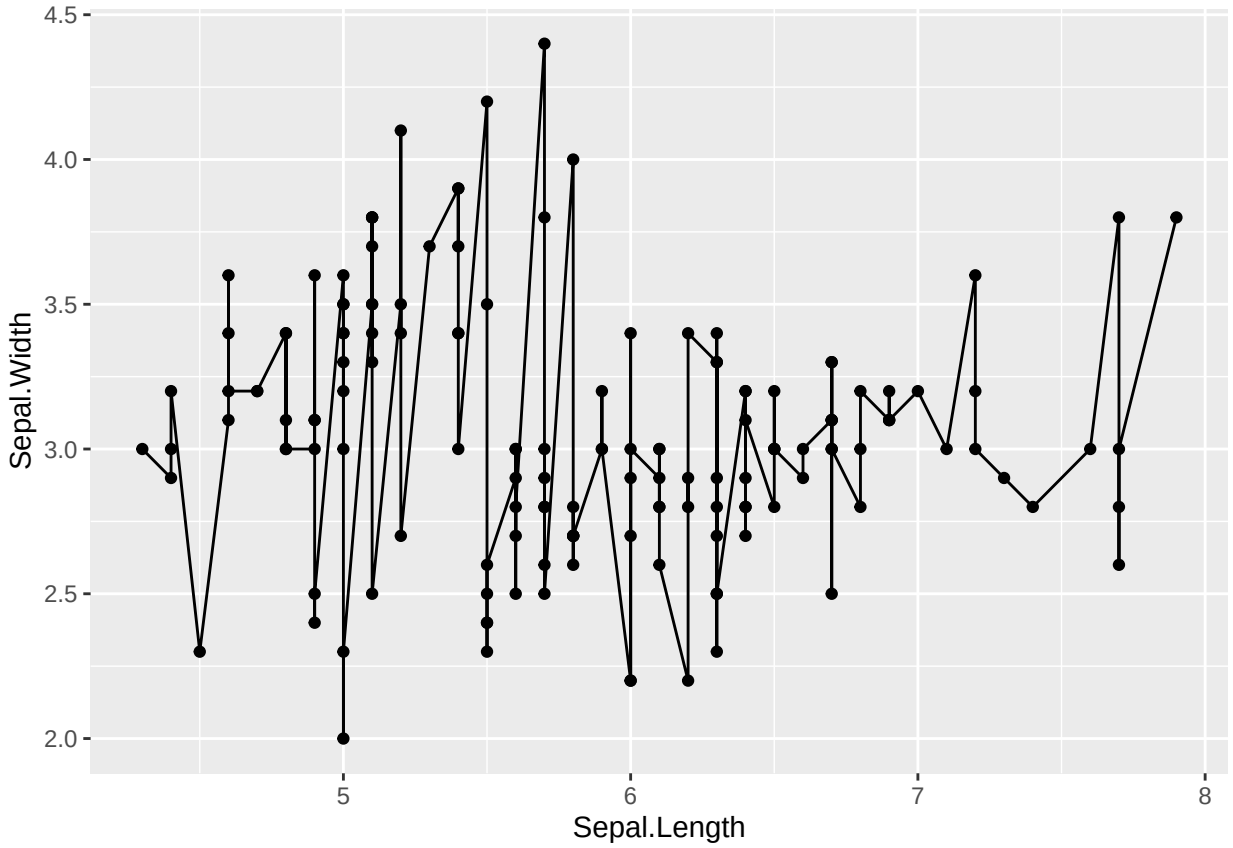


2.2. Line plots in R

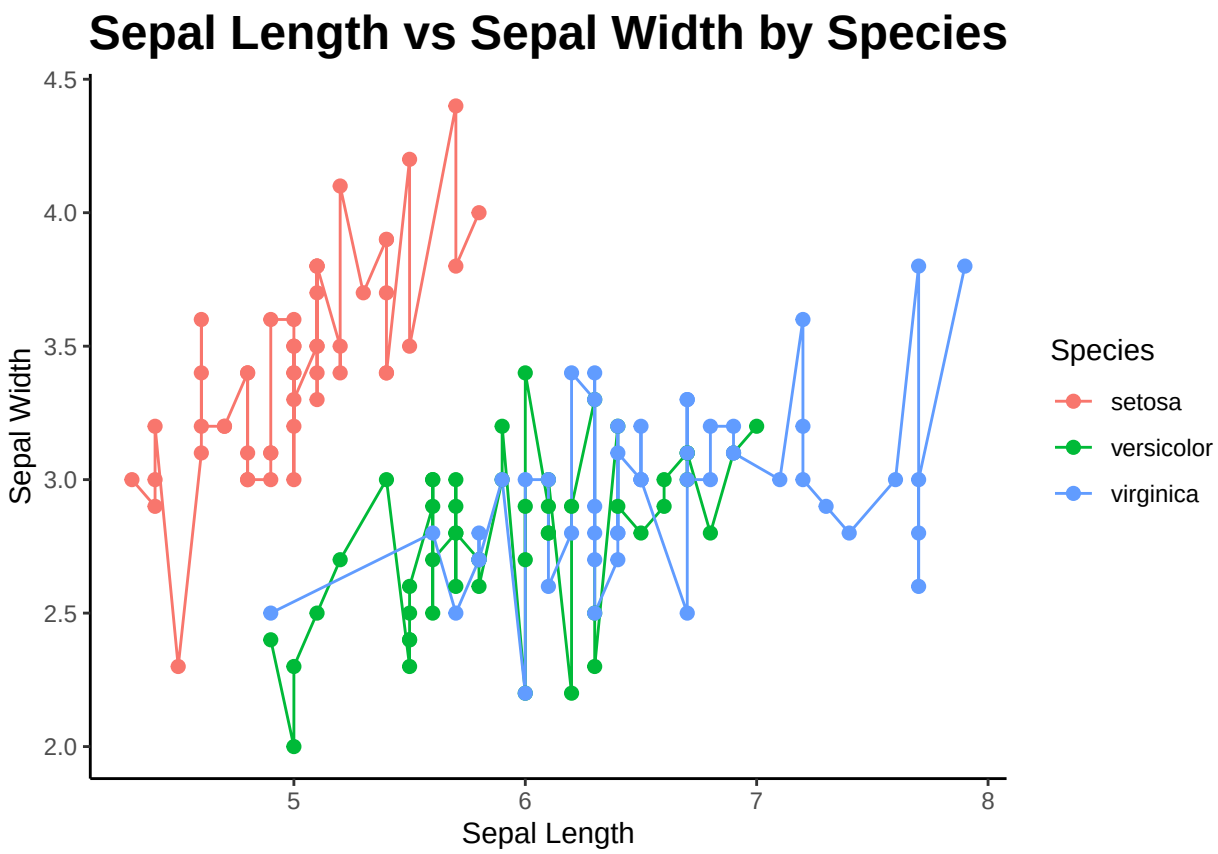
```
# Change the line type
ggplot(data=iris, aes(x = Sepal.Length, y = Sepal.Width)) +
  geom_line(linetype = "dashed")
```




```
# add points
ggplot(data=iris, aes(x = Sepal.Length, y = Sepal.Width)) +
  geom_line(linetype = "solid")+
  geom_point()
```




```
# Line plot grouped by Species

ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species)) +
  geom_line() +  # Line plot grouped by Species
  geom_point(size = 2) +  # Scatter points with color by Species
  ggtitle("Sepal Length vs Sepal Width by Species") +  # Title
  scale_color_discrete(name = "Species") +  # Properly label legend
  xlab("Sepal Length") +
  ylab("Sepal Width") +
  theme(
    axis.line = element_line(colour = "black", size = 0.5),  # Make xy-axes black, size is thickness
    plot.title = element_text(hjust = 0.5, size = 18, face = "bold"),  # Centered bold title
    legend.position = "right",  # Position legend to the right
    panel.background = element_rect(fill = "white"),  # White background
    panel.grid.major = element_blank(),  # Remove major grid lines
    panel.grid.minor = element_blank()   # Remove minor grid lines
  )
```



2.3. Bar plot in R

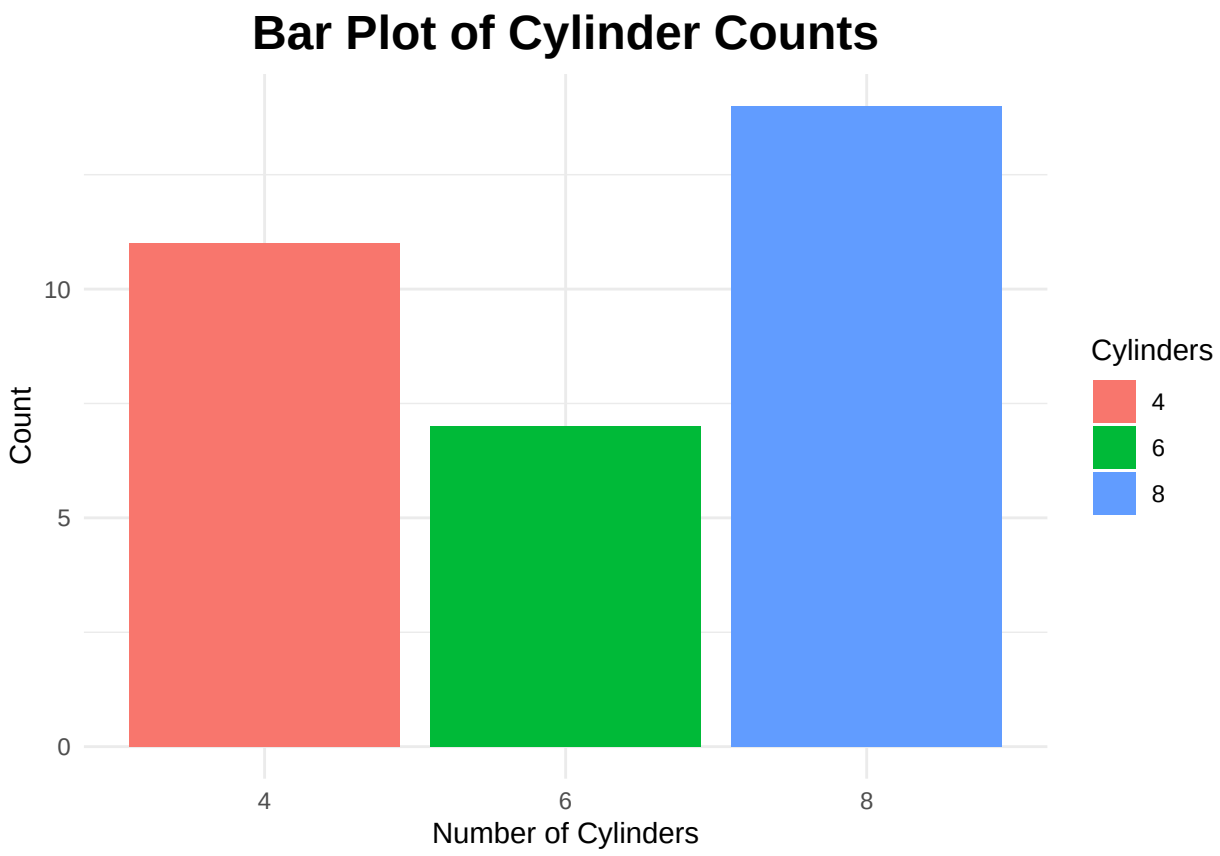
```
str(mtcars)
```

```
## 'data.frame':    32 obs. of  11 variables:
##  $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
##  $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
##  $ disp: num  160 160 108 258 360 ...
##  $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
##  $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
##  $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
##  $ qsec: num  16.5 17 18.6 19.4 17 ...
##  $ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
##  $ am  : num  1 1 1 0 0 0 0 0 0 0 ...
##  $ gear: num  4 4 4 3 3 3 3 4 4 4 ...
##  $ carb: num  4 4 1 1 2 1 4 2 2 4 ...
```

2.3.1. Creating a Bar Plot with Factor Variables in

```
mtcars %>%
  ggplot(aes(x = factor(cyl), fill = factor(cyl))) +  # x varaible must be factor, y is not required wi
  geom_bar() +
```

```
  scale_fill_discrete(name = "Cylinders") +  # Change Legend title
  ggtitle("Bar Plot of Cylinder Counts") +
  xlab("Number of Cylinders") +
  ylab("Count") +
  theme_minimal()+
 theme(
    #axis.line = element_line(colour = "black", size = 0.5),  # Make xy-axes black, size is thickness
    plot.title = element_text(hjust = 0.5, size = 18, face = "bold"),  # Centered bold title
  )
```



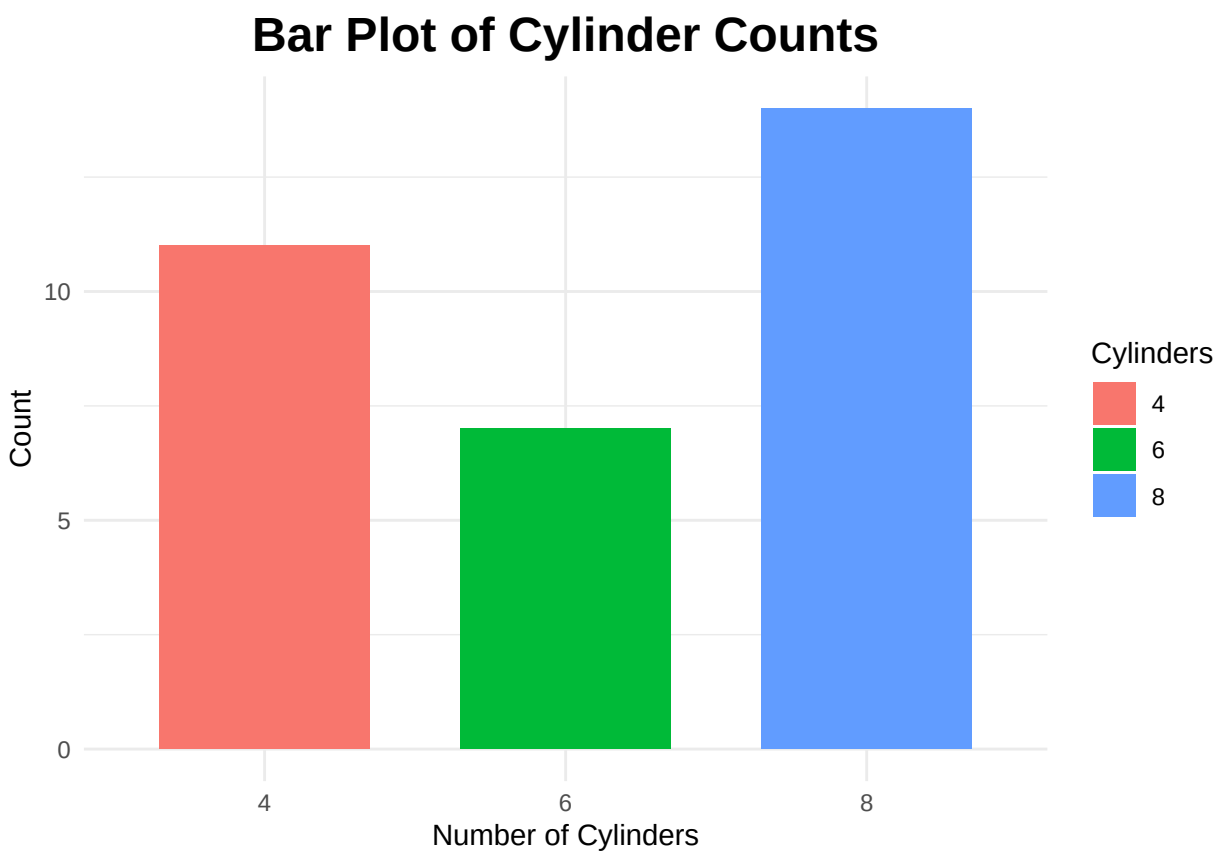
2.3.2. Creating a Bar Plot when we know the frequency (counts) of the labels

```
df = mtcars %>%
  group_by(cyl)%>%
  summarise(count = n())
df
```

```
## # A tibble: 3 x 2
##     cyl count
##   <dbl> <int>
## 1     4    11
## 2     6     7
## 3     8    14
```

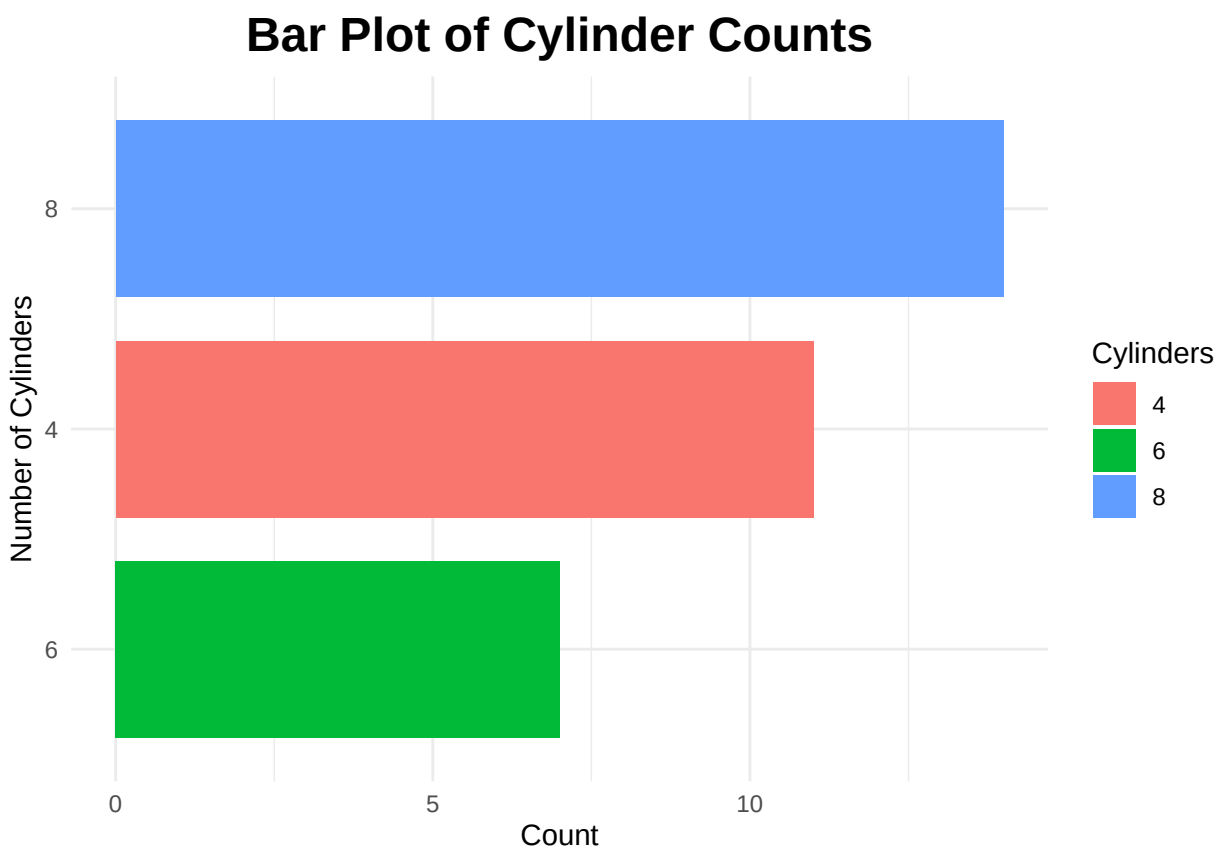
- In ggplot2, the geom_bar() function has a default statistical transformation (stat = "count"), which automatically counts occurrences of categories when no y variable is specified.
- However, if we already have pre-computed values (such as exact counts or aggregated data) and want to use them directly, we set stat = "identity" to ensure the bars represent the actual y values provided.

```r
# Basic barplot
df$cyl = factor(df$cyl) # changing to factor is important here, otherwise R will consider the values ar
# Create the bar plot
ggp <- ggplot(data = df, aes(x = cyl, y = count, fill = cyl)) +
  geom_bar(stat = "identity", width = 0.7) +  # Identity means use precomputed counts
  scale_fill_discrete(name = "Cylinders") +  # Update legend title
  ggtitle("Bar Plot of Cylinder Counts") +
  xlab("Number of Cylinders") +
  ylab("Count") +
  theme_minimal() +  # White background
  theme(
    plot.title = element_text(hjust = 0.5, size = 18, face = "bold")  # Centered bold title
  )

# Display the plot
print(ggp)
```



2.3.4 Horizontal Barplot

```r
# for the summary x variable need to be a factor variable
ggp <- ggplot(data = df,  aes(y = reorder(cyl,count), x = count,  fill = cyl)) +
  geom_bar(stat="identity", width=0.8) +
  scale_fill_discrete(name = "Cylinders") +  # Update legend title
  ggtitle("Bar Plot of Cylinder Counts") +
  ylab("Number of Cylinders") +
  xlab("Count") +
  theme_minimal() +  # White background
  theme(
    plot.title = element_text(hjust = 0.5, size = 18, face = "bold")  # Centered bold title
  )

ggp
```



2.3.5. Application to the Boston-311 Service Data

```r
Boston311_2023_data =
read.csv("https://data.boston.gov/dataset/8048697b-ad64-4bfc-b090-ee00169f2323/resource/e6013a93-1321-4
```

```r
library(stringr)
library(dplyr)
Boston311_2023_data$Parking_Enforcement_status <- str_detect(Boston311_2023_data$case_title,
                                                             regex("\\bParking Enforcement\\b"))
```

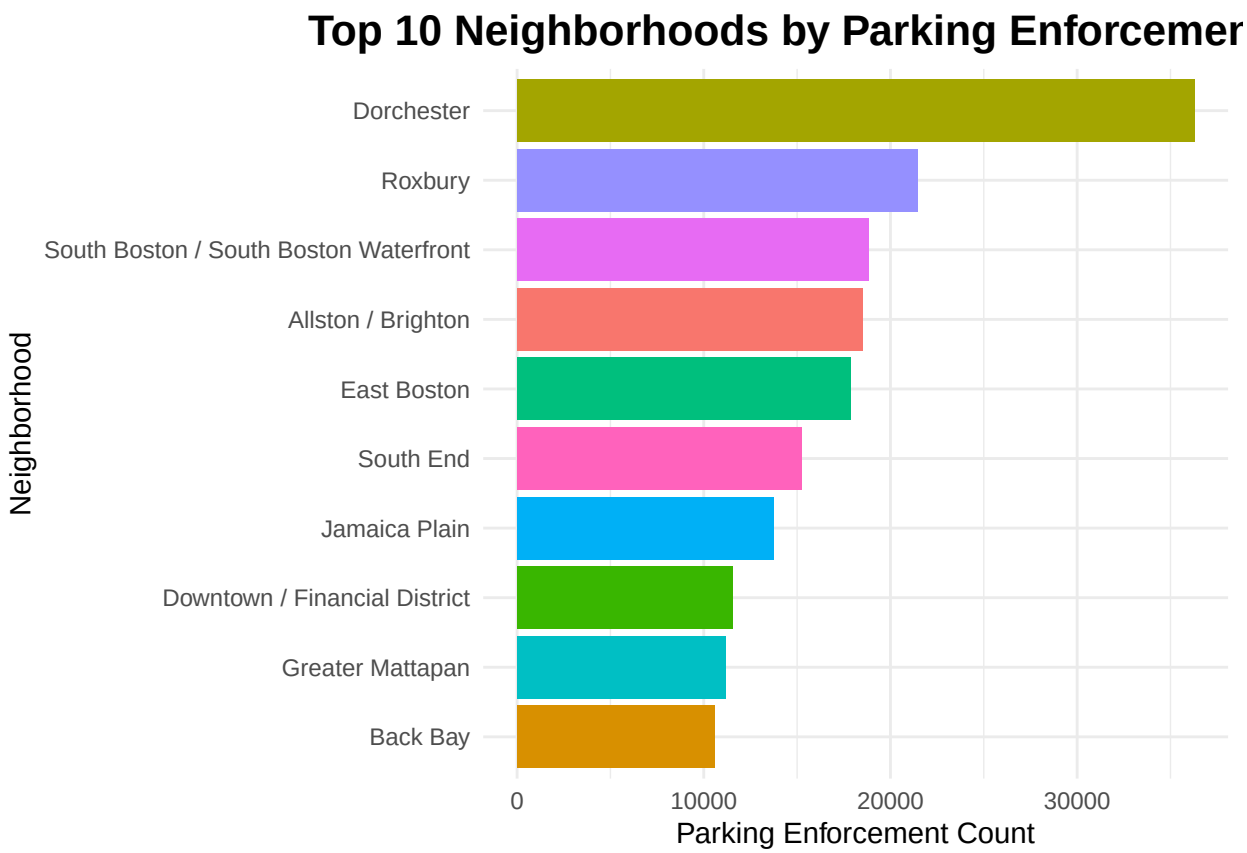
```r
Parking_Enforcement_by_nbd <- Boston311_2023_data %>%
  group_by(neighborhood) %>%
  summarise(nbd_count_Parking_Enforcement = n()) %>%
  arrange(desc(nbd_count_Parking_Enforcement))
head(Parking_Enforcement_by_nbd, 10)
```

```
## # A tibble: 10 x 2
##    neighborhood                          nbd_count_Parking_Enforcement
##    <chr>                                                         <int>
##  1 Dorchester                                                    36272
##  2 Roxbury                                                       21426
##  3 South Boston / South Boston Waterfront                        18835
##  4 Allston / Brighton                                            18490
##  5 East Boston                                                   17862
##  6 South End                                                     15265
##  7 Jamaica Plain                                                 13728
##  8 Downtown / Financial District                                 11526
##  9 Greater Mattapan                                              11191
## 10 Back Bay                                                      10559
```

```r
top_10_nbd = Parking_Enforcement_by_nbd[1:10, ]
top_10_nbd
```

```
## # A tibble: 10 x 2
##    neighborhood                          nbd_count_Parking_Enforcement
##    <chr>                                                         <int>
##  1 Dorchester                                                    36272
##  2 Roxbury                                                       21426
##  3 South Boston / South Boston Waterfront                        18835
##  4 Allston / Brighton                                            18490
##  5 East Boston                                                   17862
##  6 South End                                                     15265
##  7 Jamaica Plain                                                 13728
##  8 Downtown / Financial District                                 11526
##  9 Greater Mattapan                                              11191
## 10 Back Bay                                                      10559
```

```r
ggp_311 <- ggplot(top_10_nbd, aes(y = reorder(neighborhood, nbd_count_Parking_Enforcement),
                                  x = nbd_count_Parking_Enforcement,
                                  fill = neighborhood)) +
  geom_bar(stat = "identity") +
  scale_fill_discrete(name = "Neighborhood") +  # Corrected legend title
  ggtitle("Top 10 Neighborhoods by Parking Enforcement Count") +  # Added title
  xlab("Parking Enforcement Count") +
  ylab("Neighborhood") +
  theme_minimal() +  # Clean background instead of theme_void()
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),  # Centered, bold title
    legend.position = "none"  # Removes the legend
  )

ggp_311
```



2.3.5. Saving Plots

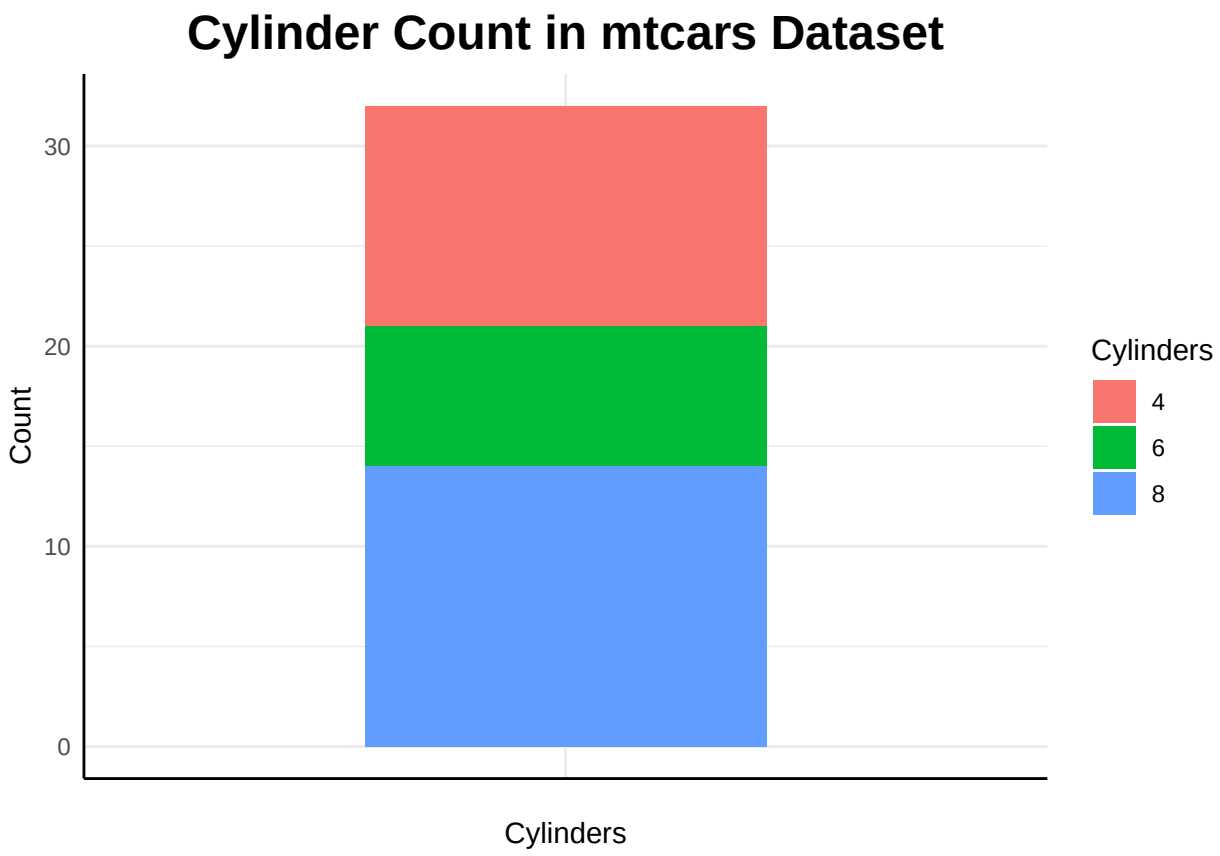
```
ggsave("~/Desktop/NBD_complaint.png") # save in desktop with file name "NBD_complaint"
```

```
## Saving 6.5 x 4.5 in image
```

2.4. Stacked bar plot

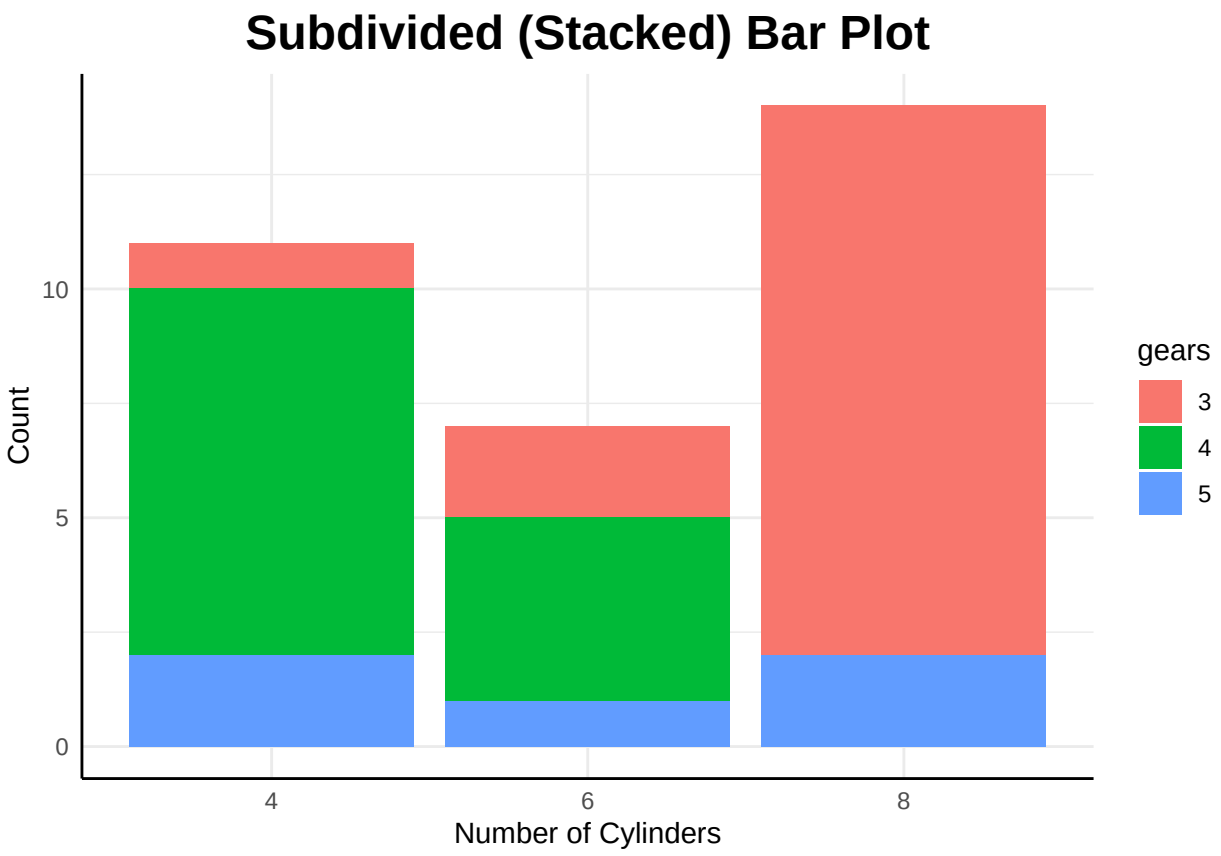
a. Creating Single Stacked bar plot is exactly similar as bar-graph, but we leave x as x = " " (empty).

```
# Create bar plot
ggp <- ggplot(data = mtcars, aes(x = '',  fill = factor(cyl))) +
    geom_bar(stat = "count", width = 0.5) +
    scale_fill_discrete(name = "Cylinders") +  # Corrected legend title
    ggtitle("Cylinder Count in mtcars Dataset") +
    xlab("Cylinders") +
    ylab("Count") +
    theme_minimal()+
    theme(
    axis.line = element_line(colour = "black", size = 0.5),  # Make xy-axes black, size is thickness
    plot.title = element_text(hjust = 0.5, size = 18, face = "bold")  # Centered bold title
)
ggp
```



b. Multiple stacked bar plot based on cylinder and gears

```r
library(ggplot2)

ggplot(mtcars, aes(x = factor(cyl), fill = factor(gear))) +
  geom_bar() +
  scale_fill_discrete(name = "gears") +  # Corrected legend title
  ggtitle("Subdivided (Stacked) Bar Plot") +
  xlab("Number of Cylinders") +
  ylab("Count") +
  theme_minimal()+
    theme(
    axis.line = element_line(colour = "black", size = 0.5),  # Make xy-axes black, size is thickness
    plot.title = element_text(hjust = 0.5, size = 18, face = "bold")  # Centered bold title
)
```



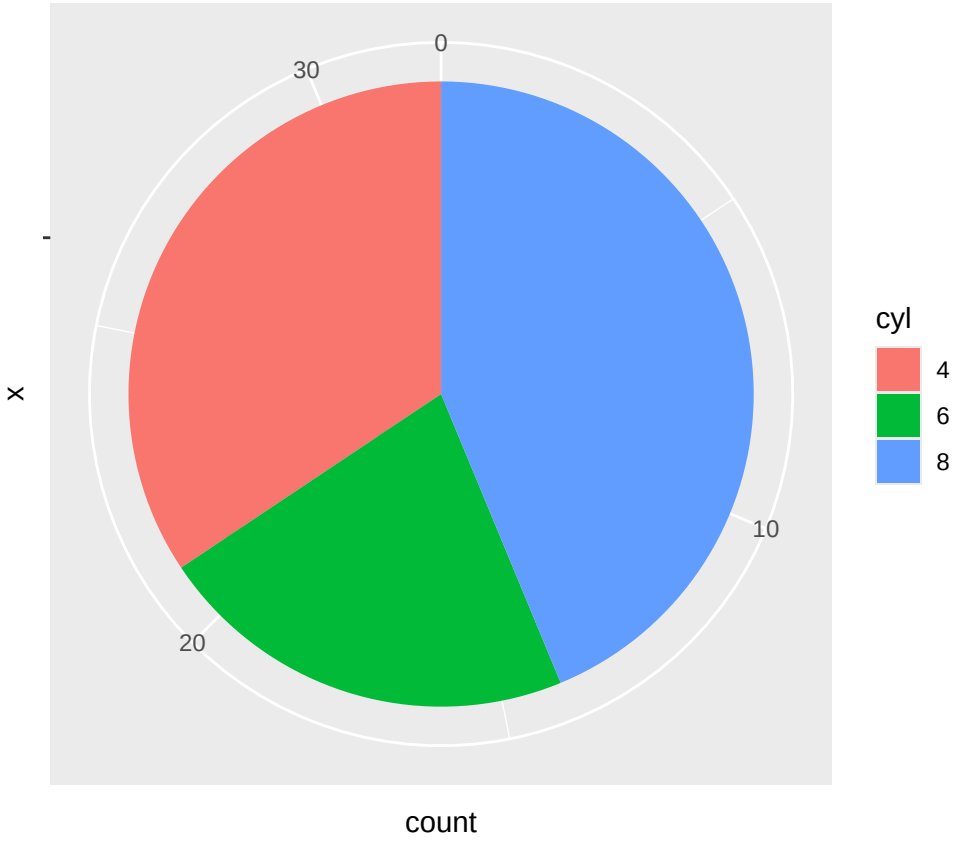
2.5. Pie chart

Creating a Pie Chart in ggplot2 is similar to a subdivided bar chart (stacked bar chart), but we need to add an additional layer using coord_polar(theta = "y") to transform it into a circular shape.

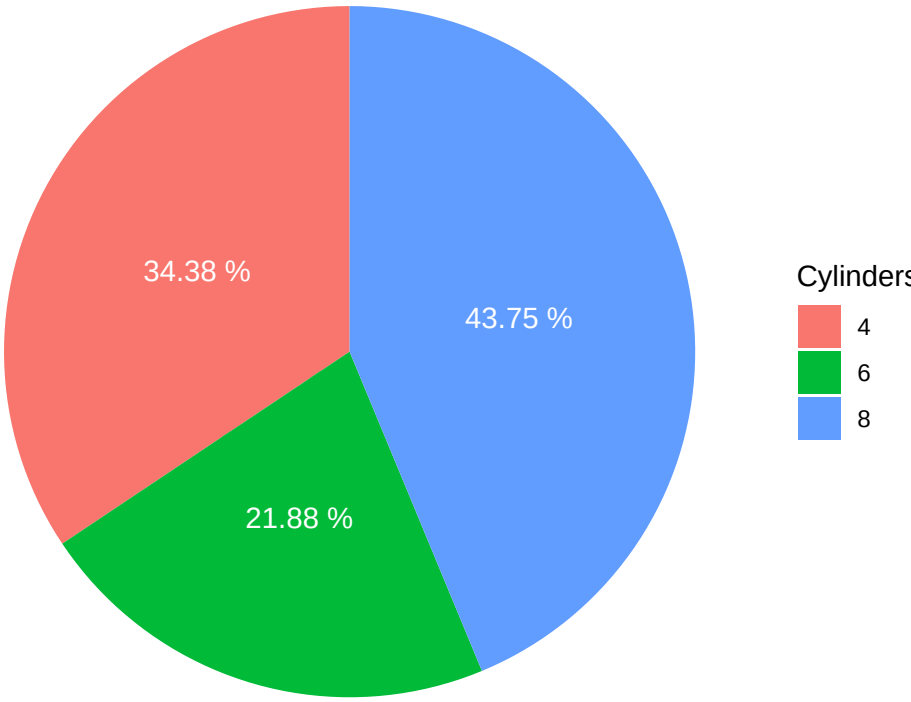
```
df <- mtcars %>%
    group_by(cyl) %>%
    summarize(count = n()) %>% # count the number of cars basedd on cylinders
    mutate(cyl = factor(cyl)) # convert to factor varaible
df
```

```
## # A tibble: 3 x 2
##   cyl   count
##   <fct> <int>
## 1 4        11
## 2 6         7
## 3 8        14
```

```
ggp <- ggplot(data=df, aes(x = '', y = count, fill = cyl)) +
  geom_bar(stat="identity", width=0.7) +
  coord_polar("y", start=0)
ggp
```




```
count_percent <- df$count/sum(df$count) *100
df$perc = round(count_percent, 2)

ggp <- ggplot(data=df, aes(x = '', y = perc, fill = cyl)) +
  geom_bar(stat="identity", width=0.7) +
  coord_polar("y", start=0)+
  scale_fill_discrete(name = "Cylinders") +  # Corrected legend title
  geom_text(aes(label = paste(perc, '%')), color = rep("white", 3),
            position = position_stack(vjust = 0.5)) +
  theme_void()
ggp
```



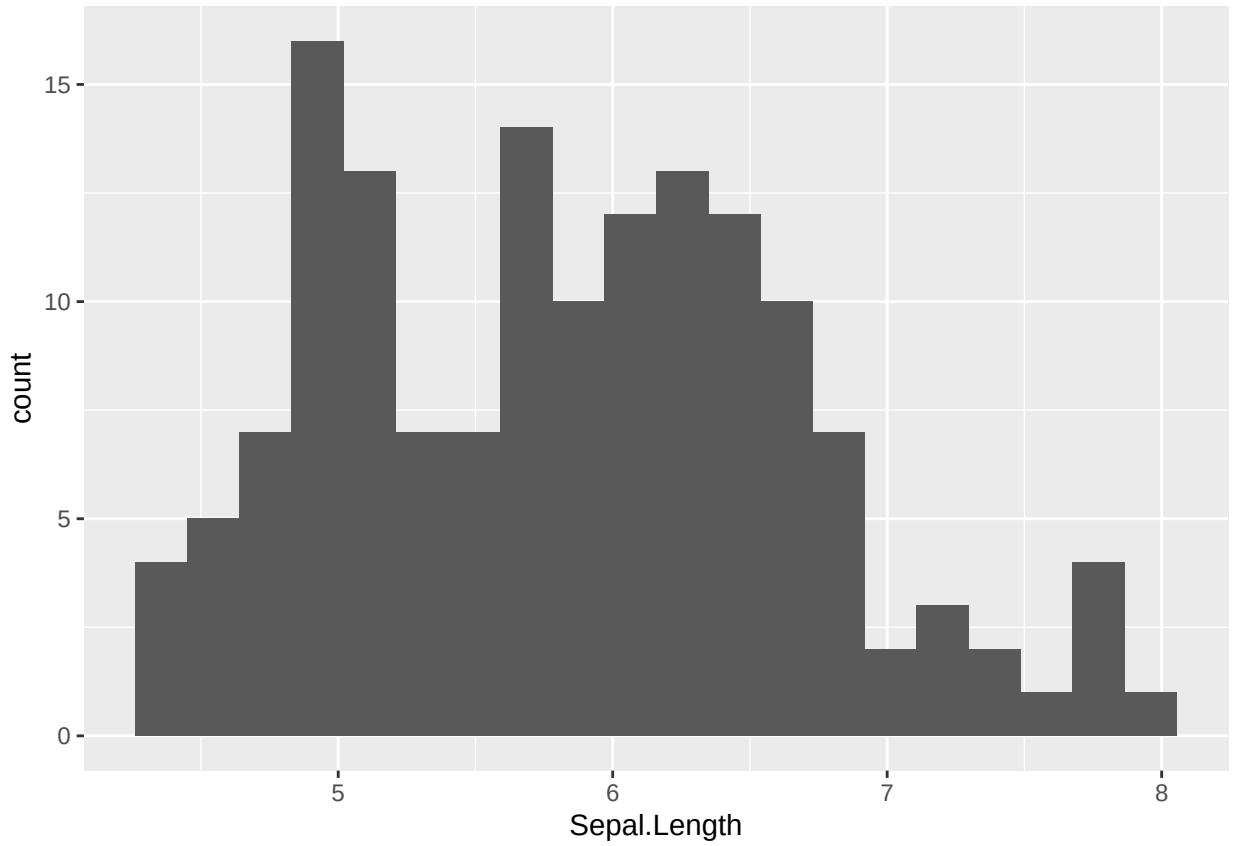
Let's add the percentage in each sector

2.6. Histogram in R

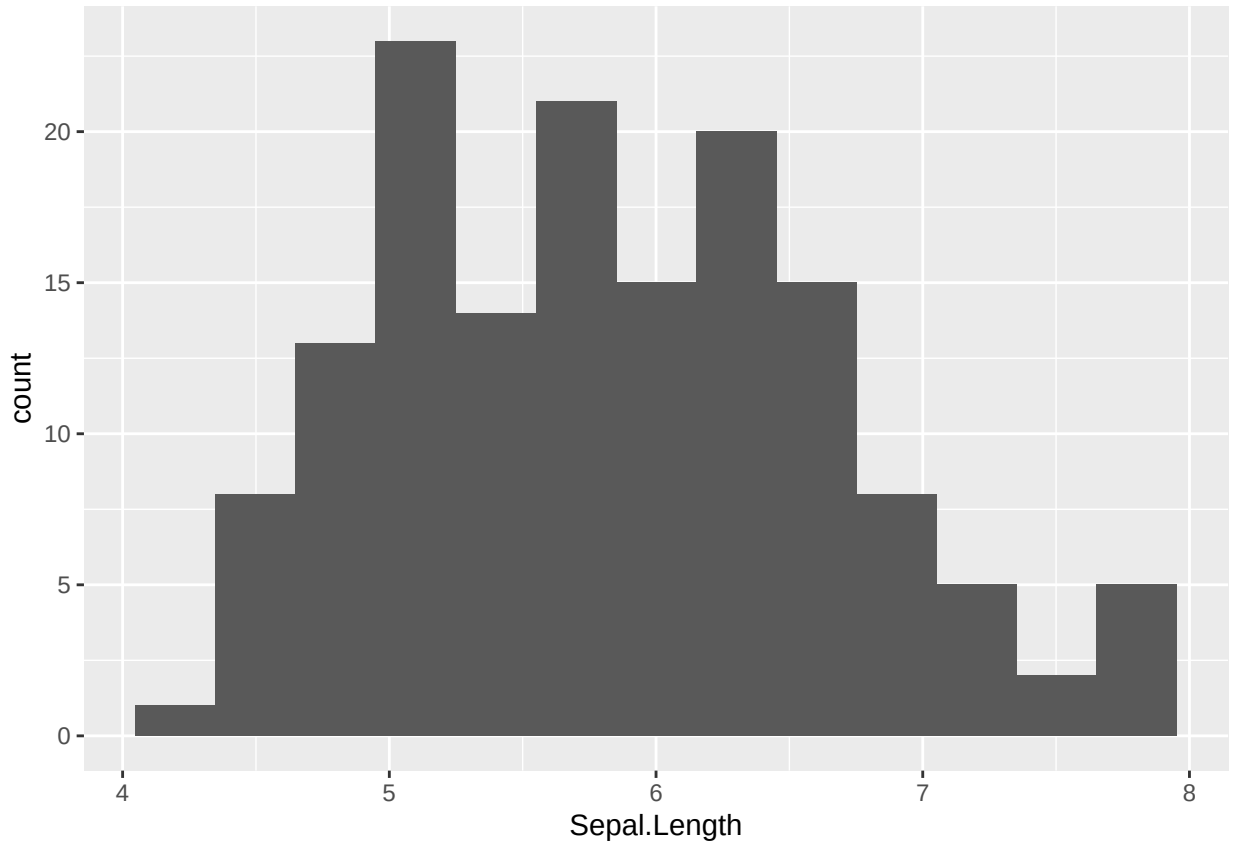
```
#?geom_histogram()
```

2.6.1. Basic histogram

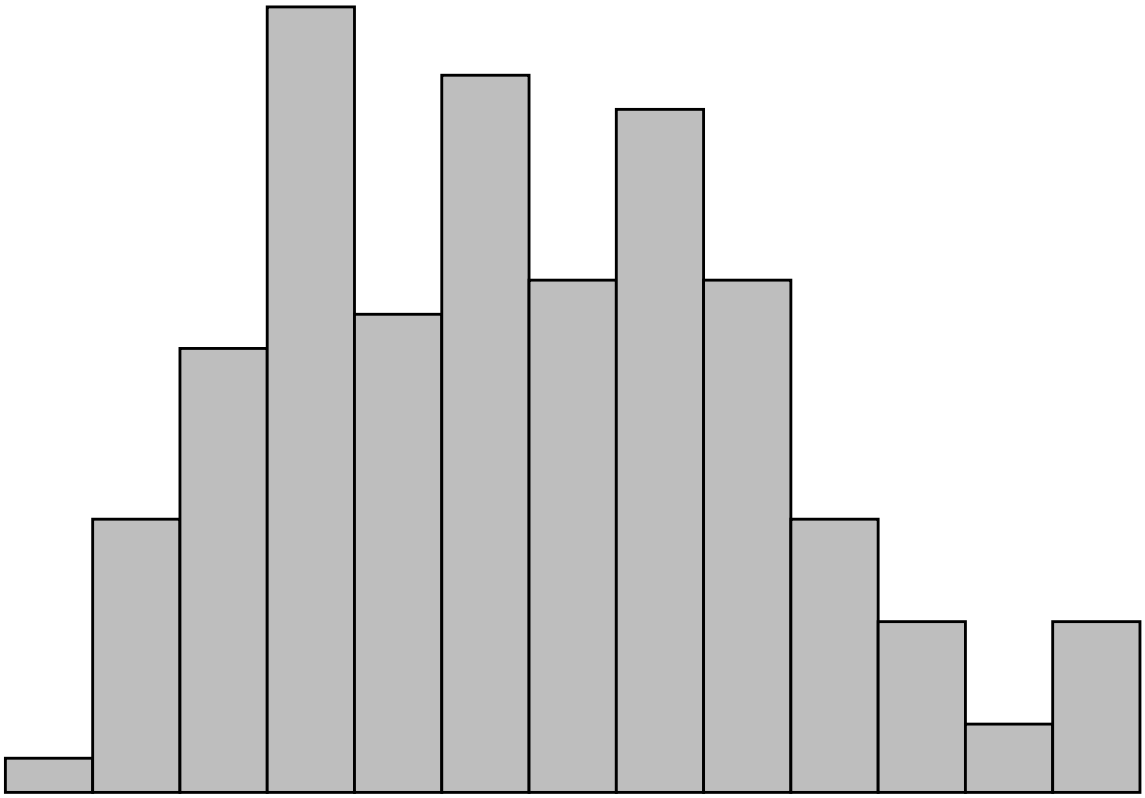
```
ggplot(iris, aes(x=Sepal.Length)) +
  geom_histogram(bins = 20)  # number of bins 20, you can change this
```




```
# Change the width of bins
ggplot(iris, aes(x=Sepal.Length)) +
  geom_histogram(binwidth=0.3,bins = 20)
```



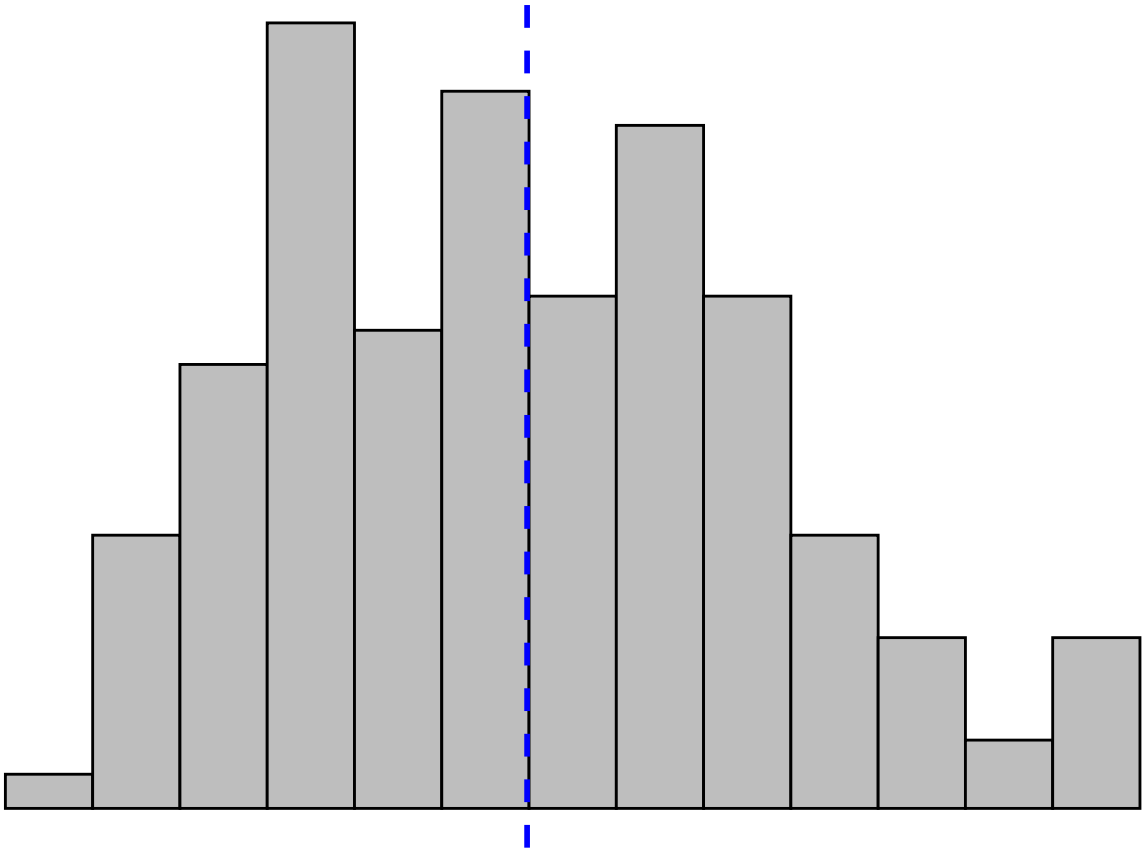

```
# Change colors
#` color means boundary color
#` fill means color fill inside the geometrical object
p <-ggplot(iris, aes(x=Sepal.Length)) +
  geom_histogram(binwidth=0.3,bins = 20, color="black", fill="gray")+
  theme_void()
p
```



Add mean line and density plot on the histogram

```
# Add mean line
p + geom_vline(aes(xintercept=mean(Sepal.Length)),
            color="blue", linetype="dashed", size=1)
```

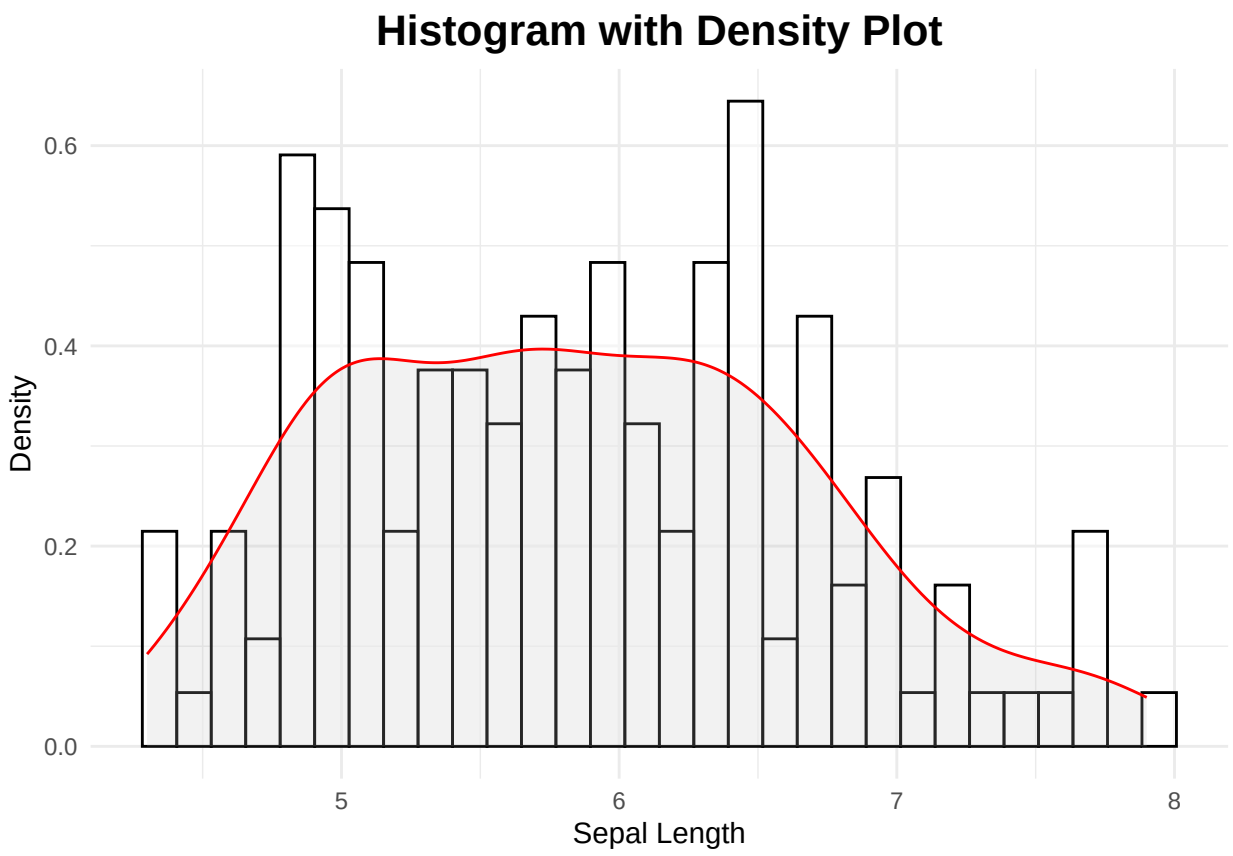
```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



2.6.2. Histogram with density plot

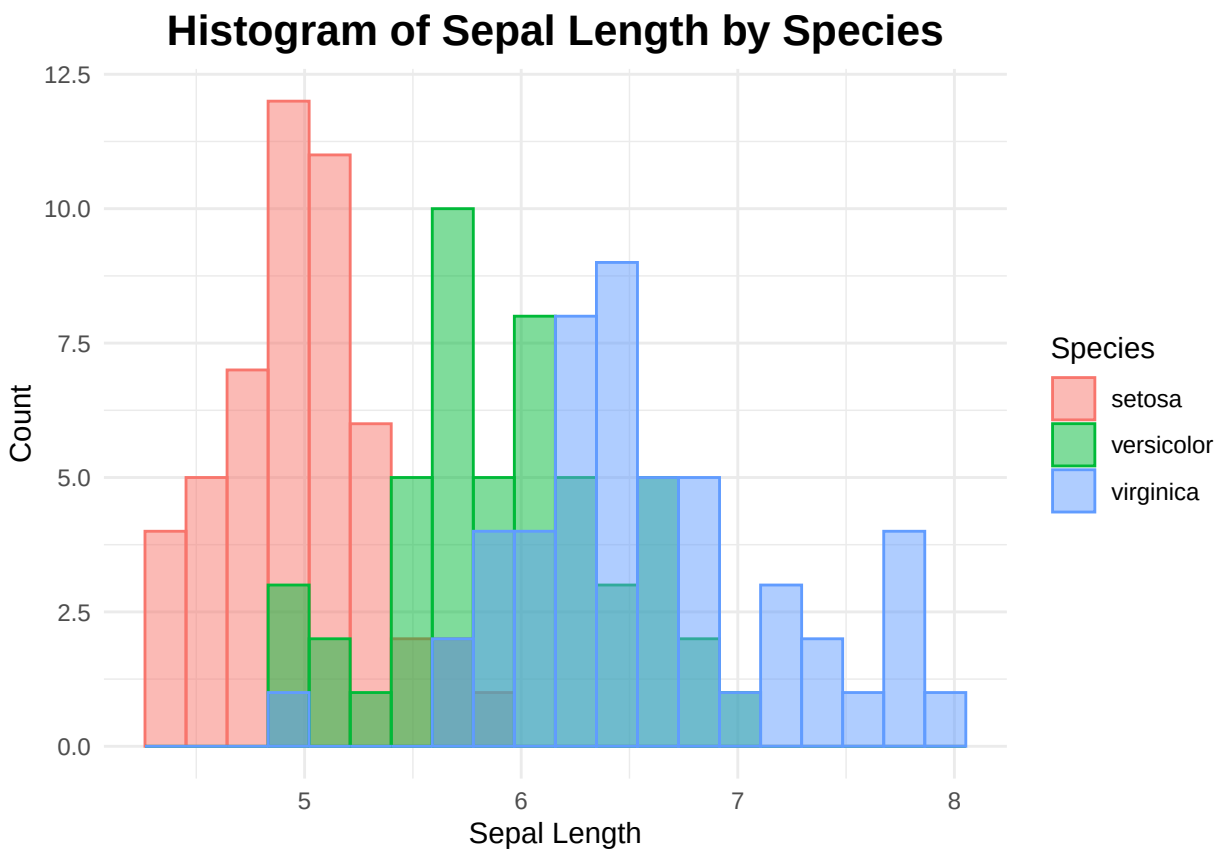
```
ggplot(iris, aes(Sepal.Length)) +
  geom_histogram(aes(y = ..density..),
                 color = "black", fill = "white", alpha = 0.5, bins = 30) +  # Adjust transparency and
  geom_density(color = "red", fill = "grey", alpha = 0.2) +  # Density curve with transparency
  ggtitle("Histogram with Density Plot") +
  xlab("Sepal Length") +
  ylab("Density") +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold")  # Centered title
  )
```

```
## Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(density)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



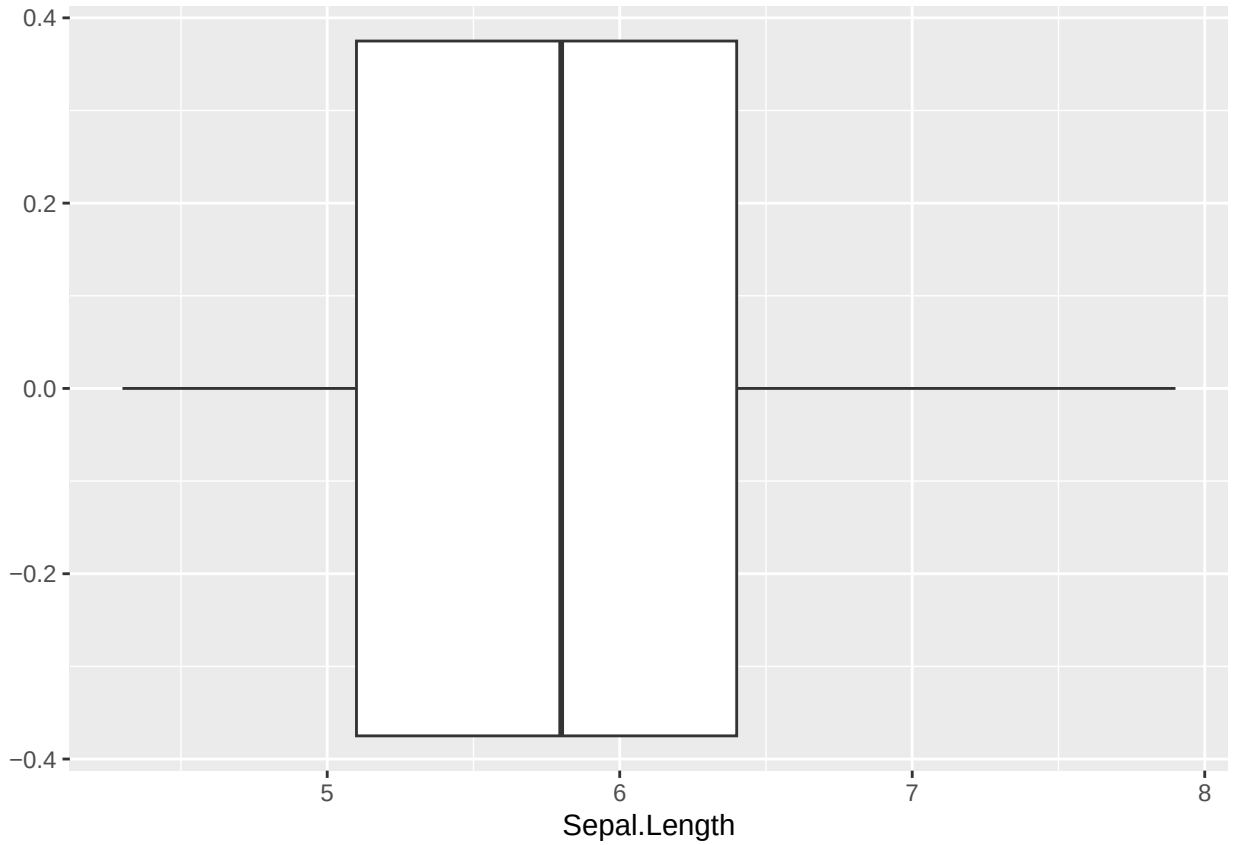
2.6.3. Histogram of a varaible with multiple groups

```r
# Change histogram plot line colors by groups
ggplot(iris, aes(x = Sepal.Length, color = Species, fill = Species)) +
  geom_histogram(alpha = 0.5, position = "identity", bins = 20) +  # Adjust transparency & bins
  ggtitle("Histogram of Sepal Length by Species") +
  xlab("Sepal Length") +
  ylab("Count") +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold")  # Centered title
  )
```

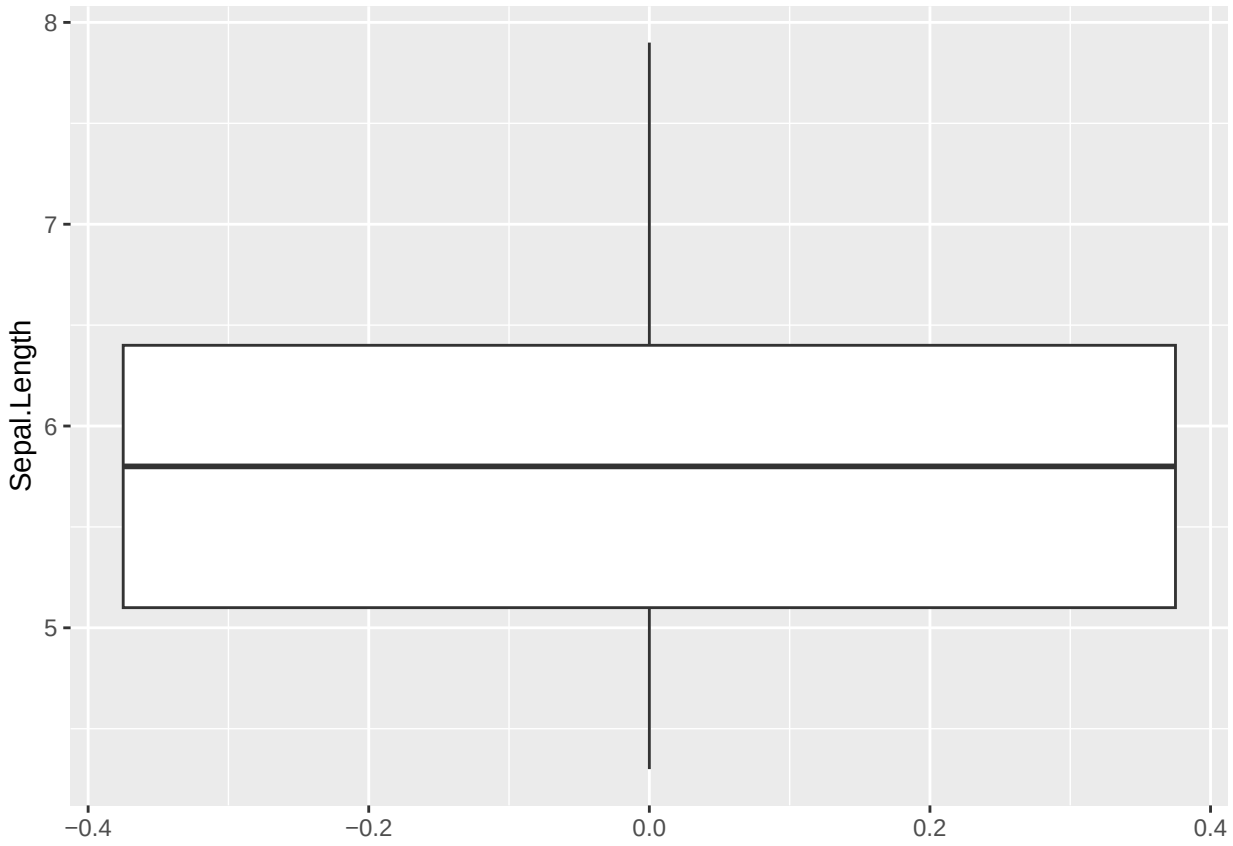


2.7. Box plot

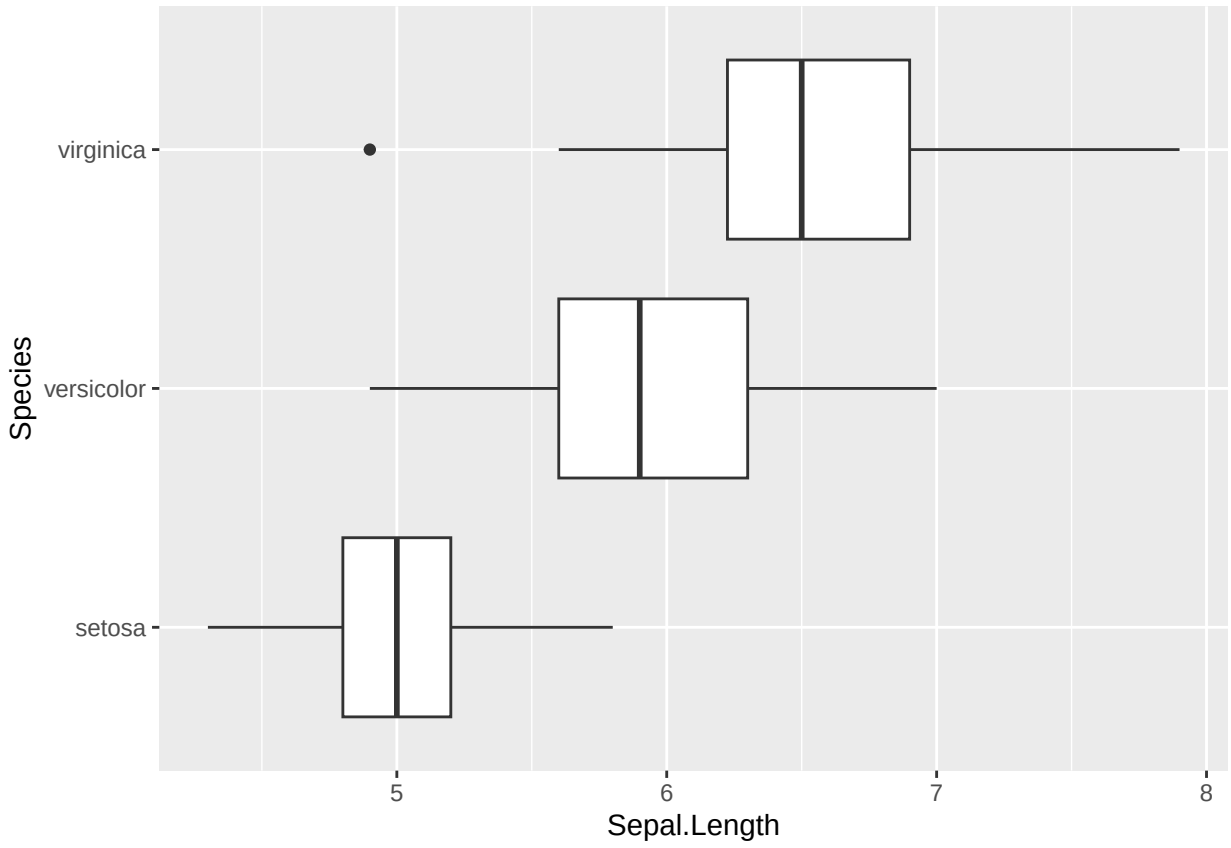
```
# Basic box plot
p <- ggplot(iris, aes(Sepal.Length)) +
  geom_boxplot()
p
```



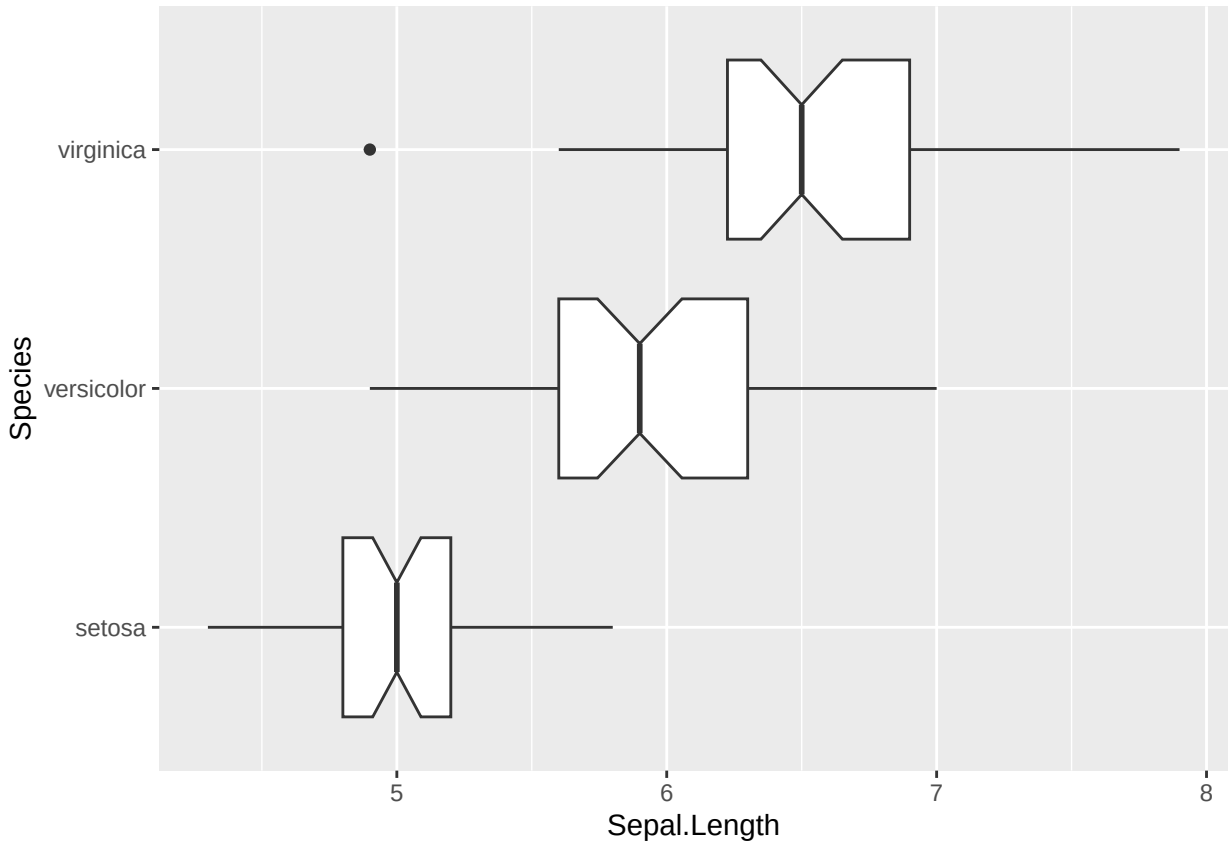

```
# Horizontal box plot
p + coord_flip()
```



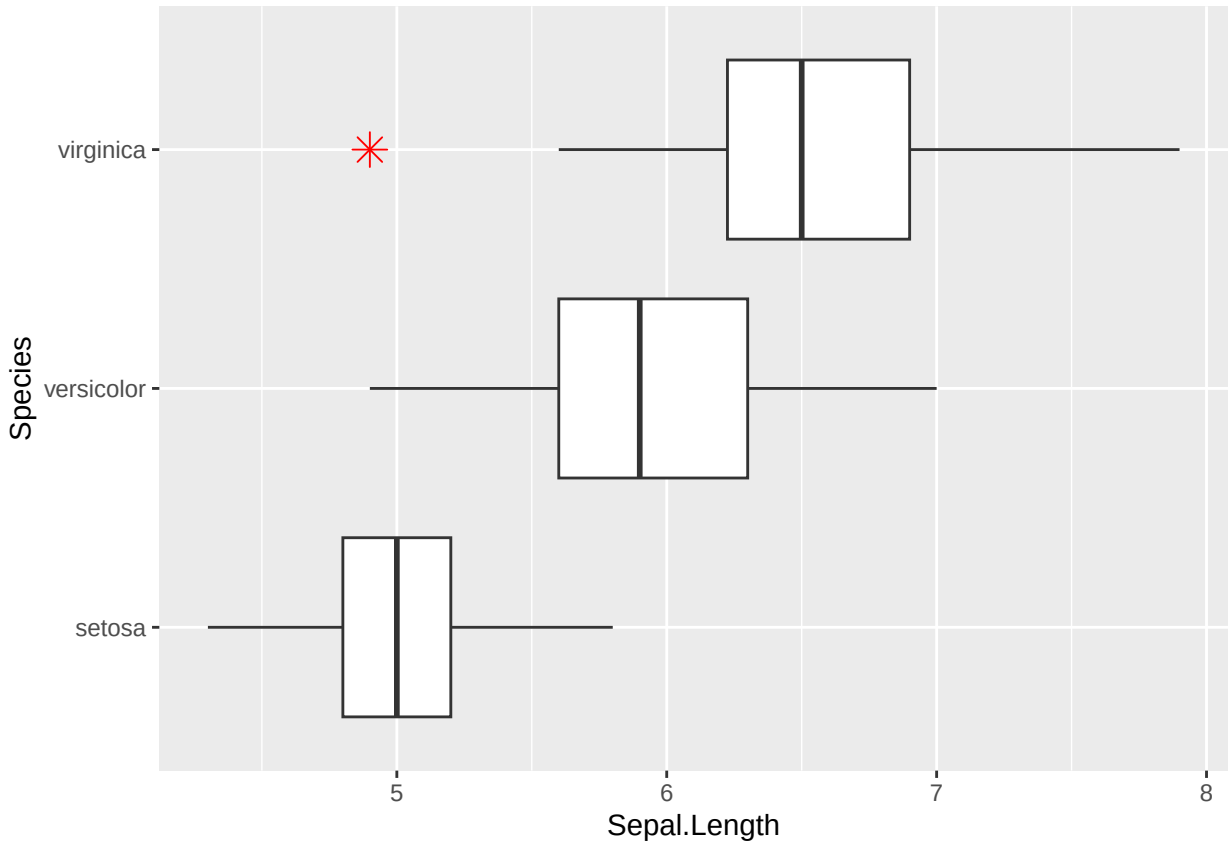

```
#box plot for multiple category
ggplot(iris, aes(x=Sepal.Length, y=Species)) +
geom_boxplot()
```




```
# Notched box plot
ggplot(iris, aes(x=Sepal.Length, y=Species)) +
  geom_boxplot(notch=TRUE)
```



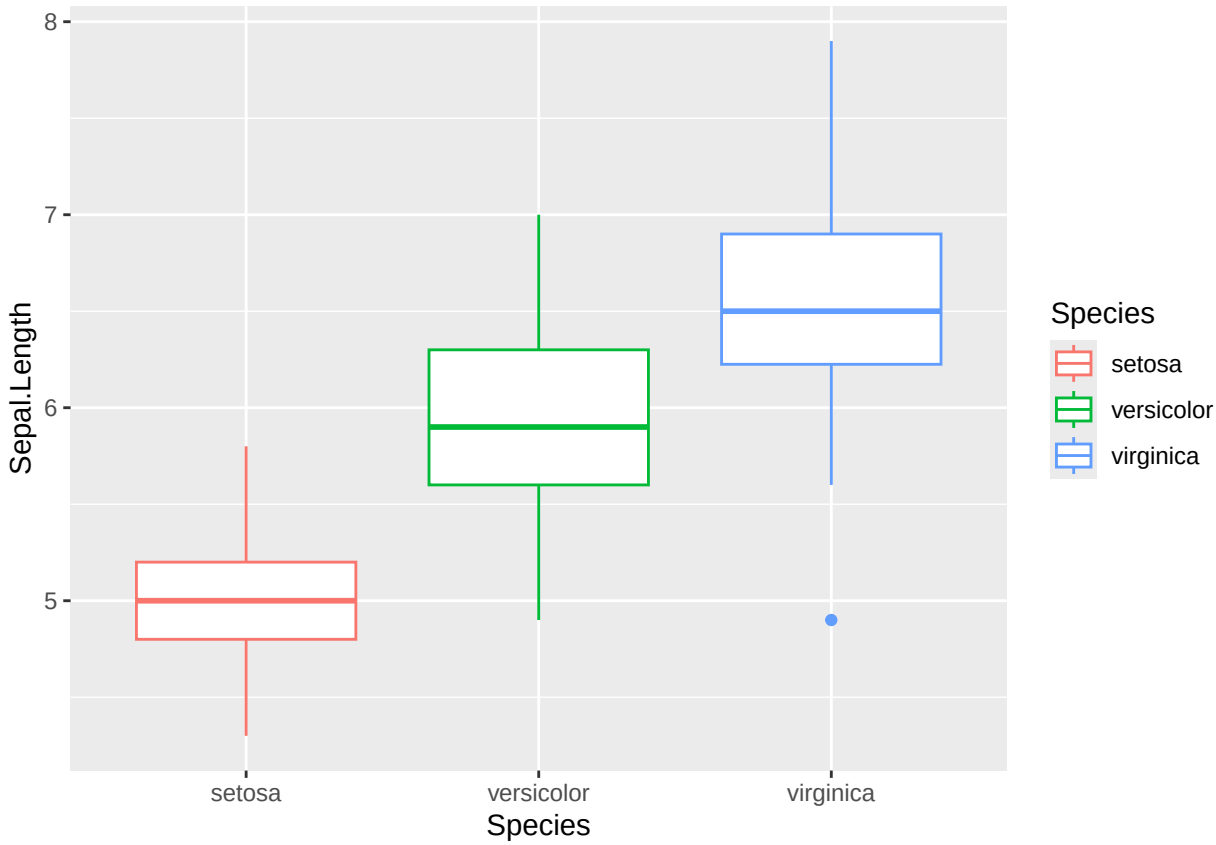

```
# Change outlier, color, shape and size
ggplot(iris, aes(x=Sepal.Length, y=Species)) +
  geom_boxplot(outlier.colour="red", outlier.shape=8,
               outlier.size=4)
```



Change box plot line colors

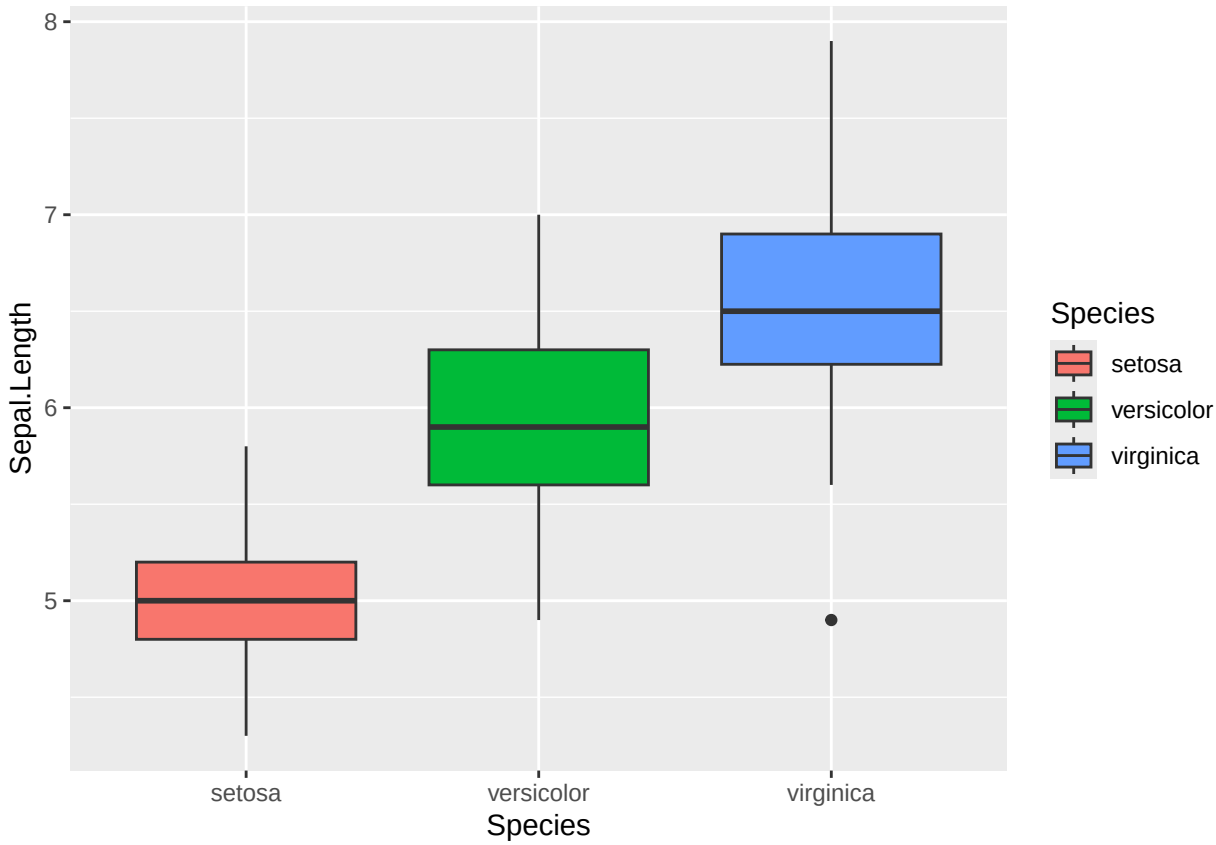
Box plot line colors can be automatically controlled by the level variable :

```
# Change box plot line colors by groups
p<-ggplot(iris, aes(y=Sepal.Length, x=Species, color = Species)) +
  geom_boxplot()
p
```

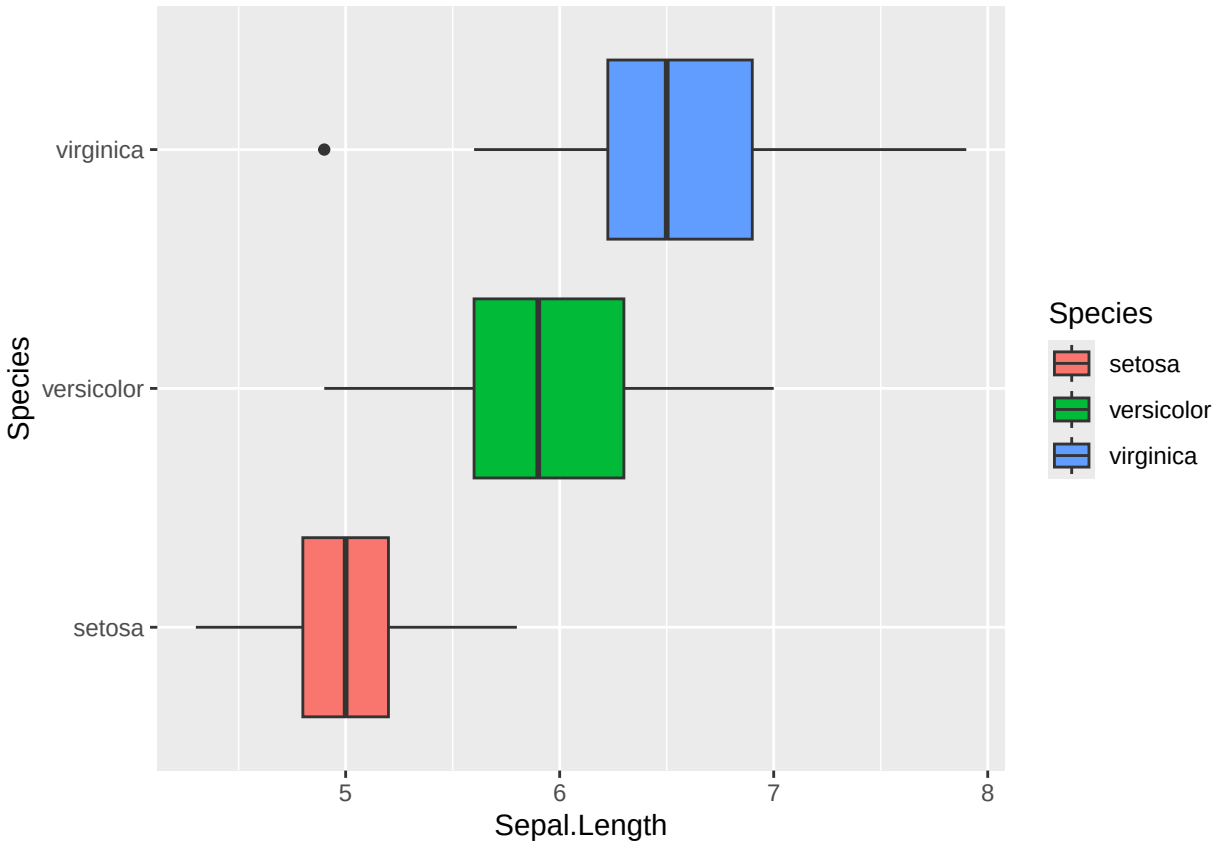


Change box plot fill colors

```
# Change box plot colors by groups
p<- ggplot(iris, aes(y=Sepal.Length, x=Species,  fill= Species)) +
   geom_boxplot()
p
```




```
p + coord_flip() # horizontal box plot
```



2.8. Facet Plots in ggplot2

Faceting in ggplot2 allows for visual comparisons across categories. Below are several insightful facet grid examples, demonstrating different types of visualizations that can benefit from faceting.

- Best for understanding distribution differences across categories.
- Faceted by Species for clearer separation of groups.

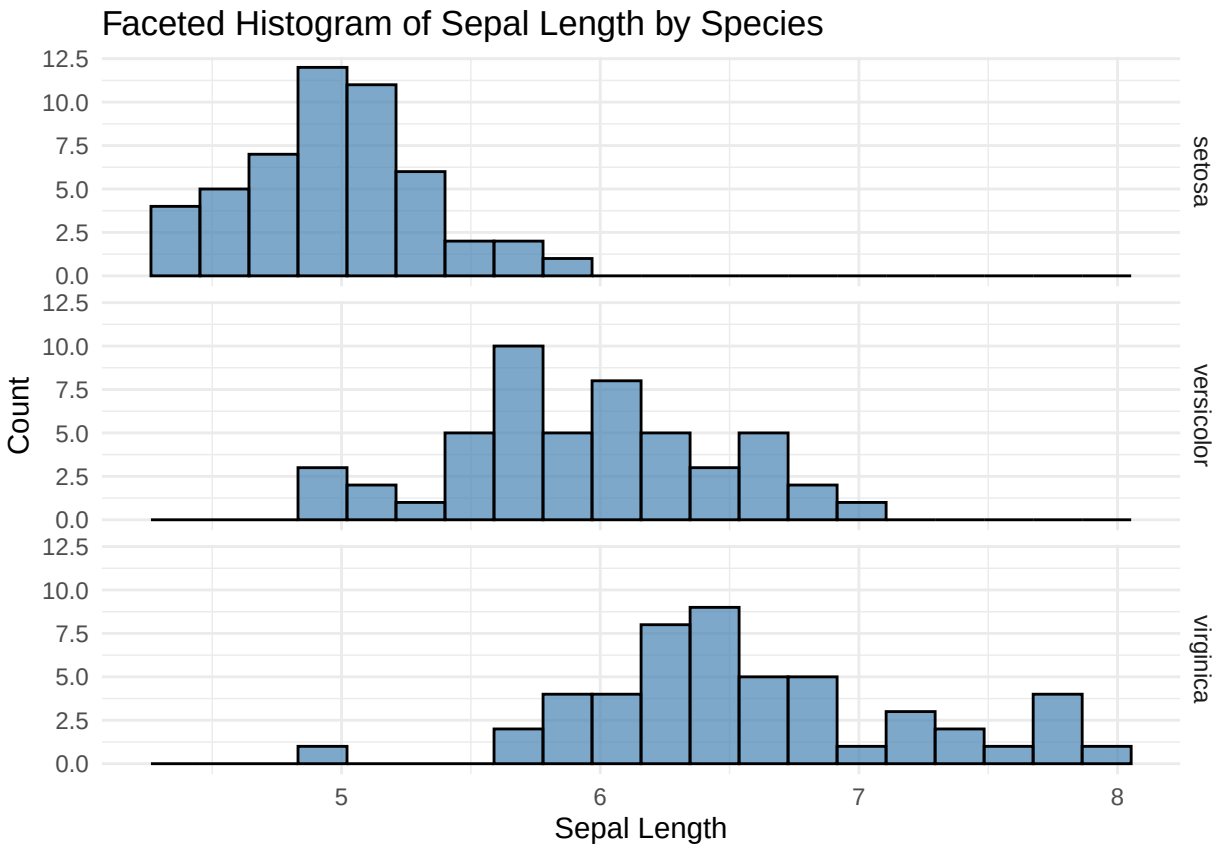
1. A Categorical Variable for Faceting Faceting is used to create multiple subplots based on levels of a categorical variable. The variable should be discrete (categorical) or grouped continuous data.
2. Facet Functions:
   - facet_wrap(~ variable): Creates small multiples in a wrapped format.
   - facet_grid(rows ~ cols): Creates a grid of plots based on row and column factors.
3. Data & Aesthetic Mappings (aes()) A dataset that contains the categorical variable for faceting. The x and y variables should be mapped inside aes().

2.8.1. Faceted Histogram (Comparing Distributions Across Categories)

Question: How does the distribution of Sepal.Length vary across Species?

```
ggplot(iris, aes(x = Sepal.Length)) +
  geom_histogram(color = "black", fill = "steelblue", bins = 20, alpha = 0.7) +
  facet_grid(Species ~ .) +  # Facet by Species
```

```
  ggtitle("Faceted Histogram of Sepal Length by Species") +
  xlab("Sepal Length") +
  ylab("Count") +
  theme_minimal()
```



Plot like this helps us to compare the distribution as follows:

- Shape: Iris setosa exhibits a symmetric distribution, while versicolor and virginica show slight right skewness.
- Measure of Center: Iris setosa has the shortest average sepal length, followed by versicolor, with virginica having the longest.
- Measure of Dispersion: Iris virginica displays the greatest variability in sepal length, whereas setosa shows the least.

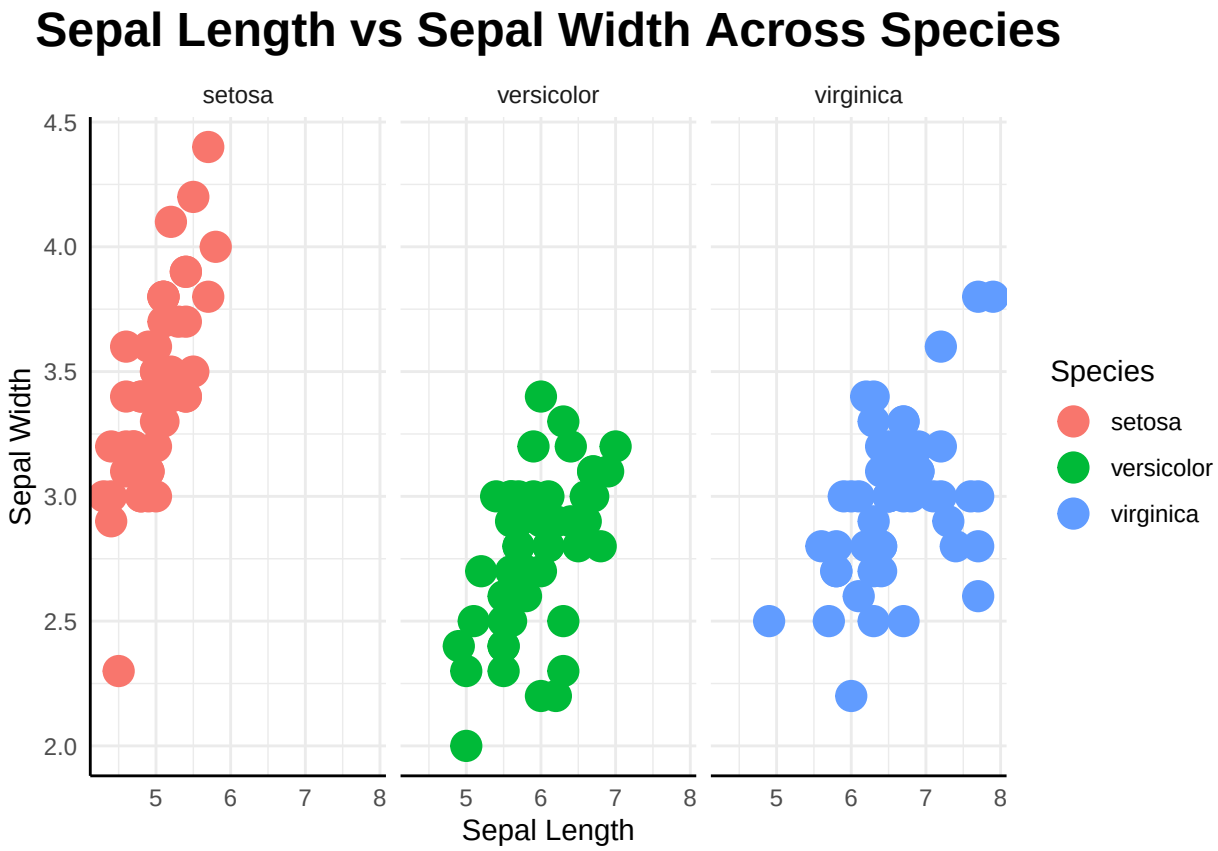
2.8.2. Faceted Scatter Plot (Relationships Across Categories)

- Best for visualizing category-wise correlations.
- Helps detect trends, clusters, or outliers within groups.

Question: How does the relationship between Sepal.Length and Sepal.Width differ across Species?

```
ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species)) +
  geom_point(size = 5) +
  facet_wrap(~ Species) +  # Creates separate scatter plots per Species
```

```
  ggtitle("Sepal Length vs Sepal Width Across Species") +
  xlab("Sepal Length") +
  ylab("Sepal Width") +
  theme_minimal()+
    theme(
    axis.line = element_line(colour = "black", size = 0.5),  # Make xy-axes black, size is thickness
    plot.title = element_text(hjust = 0.5, size = 18, face = "bold")  # Centered bold title
)
```



2.8.3. Faceted Line Chart (Time Series Data Across Groups)

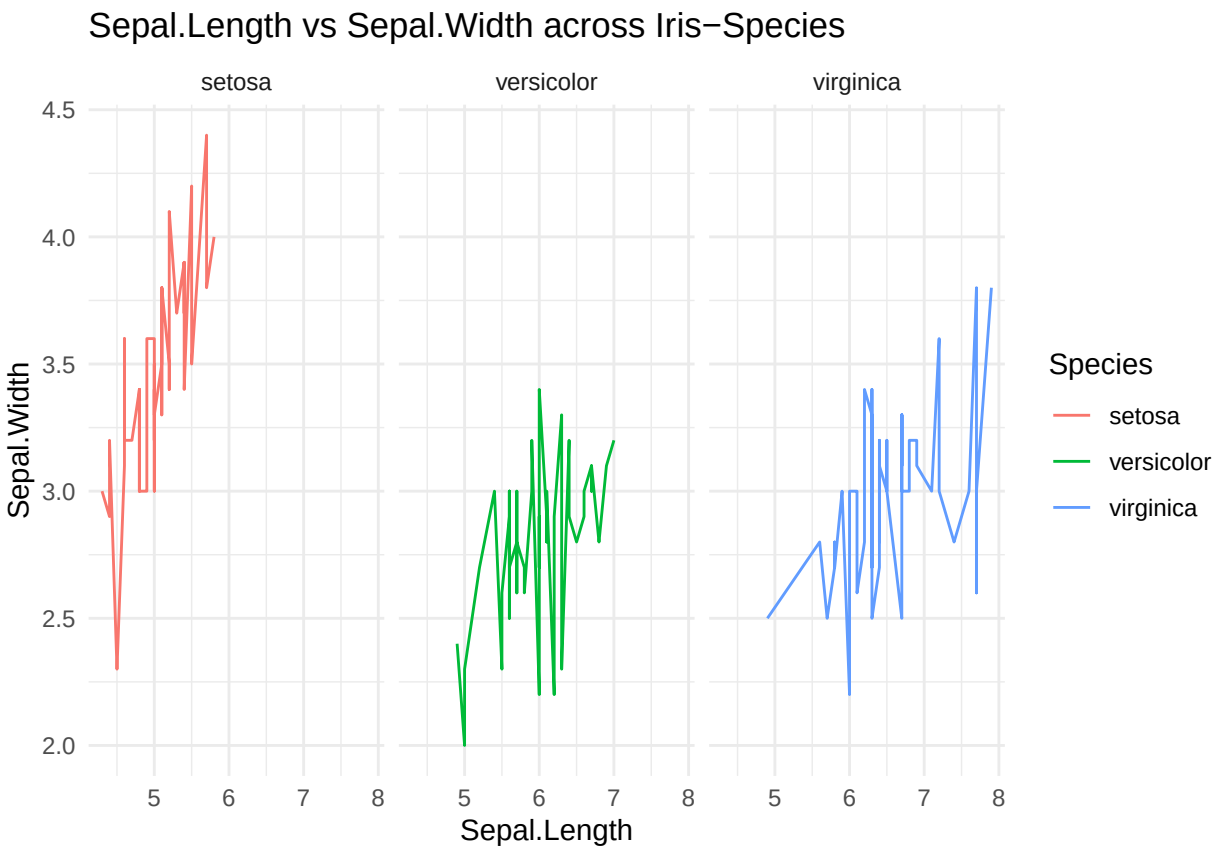
- Best for visualizing trends in different groups over a continuous variable.

Question: How does Sepal.Width trend with Sepal.Length across Iris-Species?

```
str(iris)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```r
ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species)) +
  geom_line() +
  facet_wrap(~ Species) +  # One line chart per Species
  ggtitle("Sepal.Length vs Sepal.Width across Iris-Species") +
  xlab("Sepal.Length") +
  ylab("Sepal.Width") +
  theme_minimal()
```

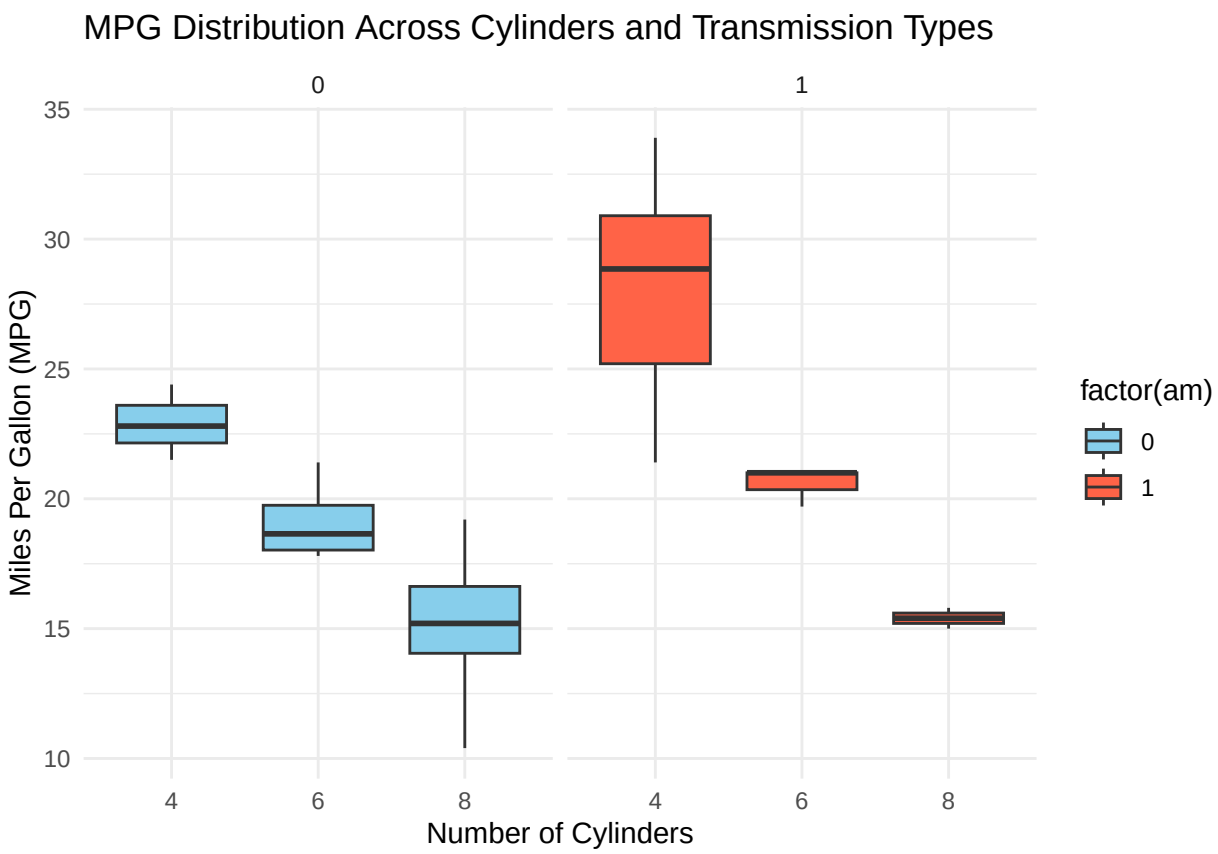


2.8.4. Faceted Box Plot (Comparing Distributions Side-by-Side)

- Best for comparing distributions across two categorical variables.

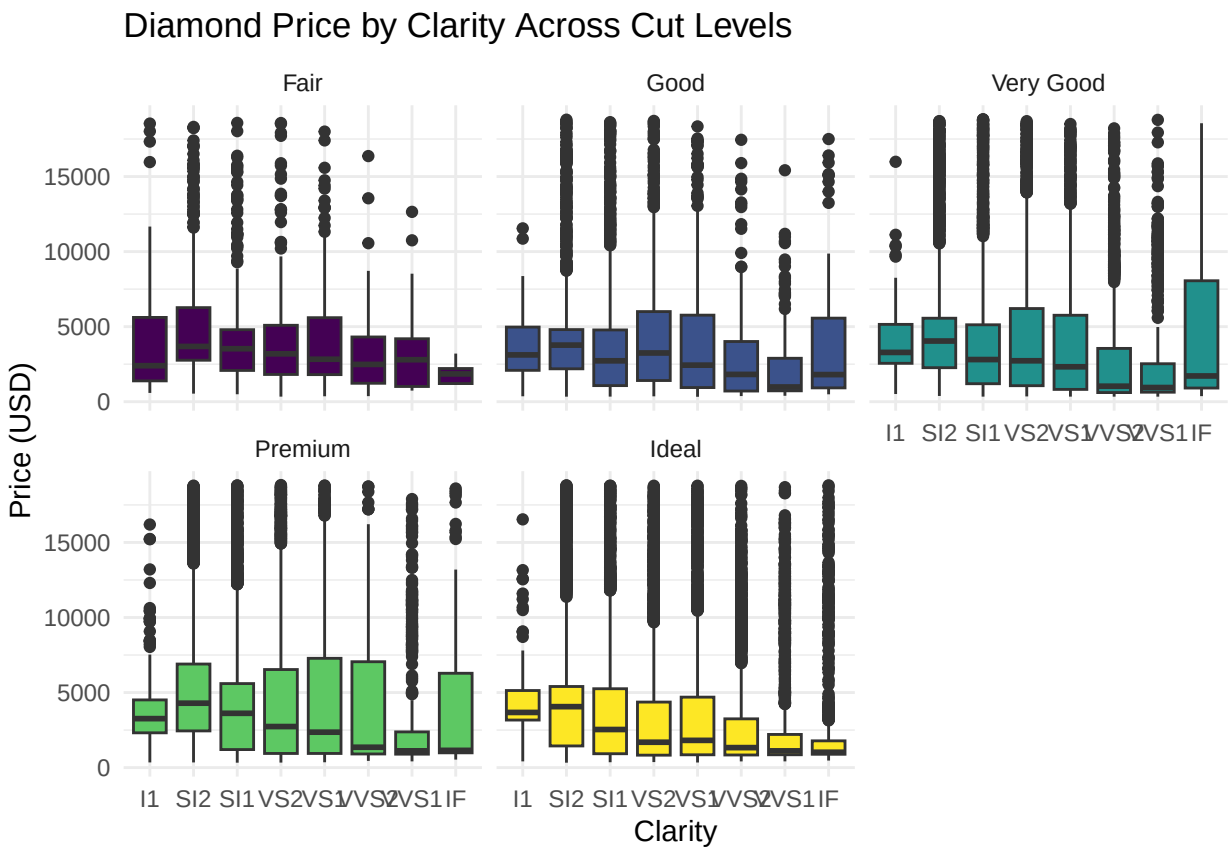
Question: How does mpg vary across transmission types and cylinders?

```r
ggplot(mtcars, aes(x = factor(cyl), y = mpg, fill = factor(am))) +
  geom_boxplot() +
  facet_grid(. ~ am) +  # Faceted by transmission (am: 0 = Automatic, 1 = Manual)
  scale_fill_manual(values = c("skyblue", "tomato")) +  # Custom colors
  ggtitle("MPG Distribution Across Cylinders and Transmission Types") +
  xlab("Number of Cylinders") +
  ylab("Miles Per Gallon (MPG)") +
  theme_minimal()
```




```
data("diamonds")
```

```
ggplot(diamonds, aes(x = clarity, y = price, fill = cut)) +
  geom_boxplot() +
  facet_wrap(~ cut) +
  ggtitle("Diamond Price by Clarity Across Cut Levels") +
  xlab("Clarity") +
  ylab("Price (USD)") +
  theme_minimal() +
  theme(legend.position = "none")
```

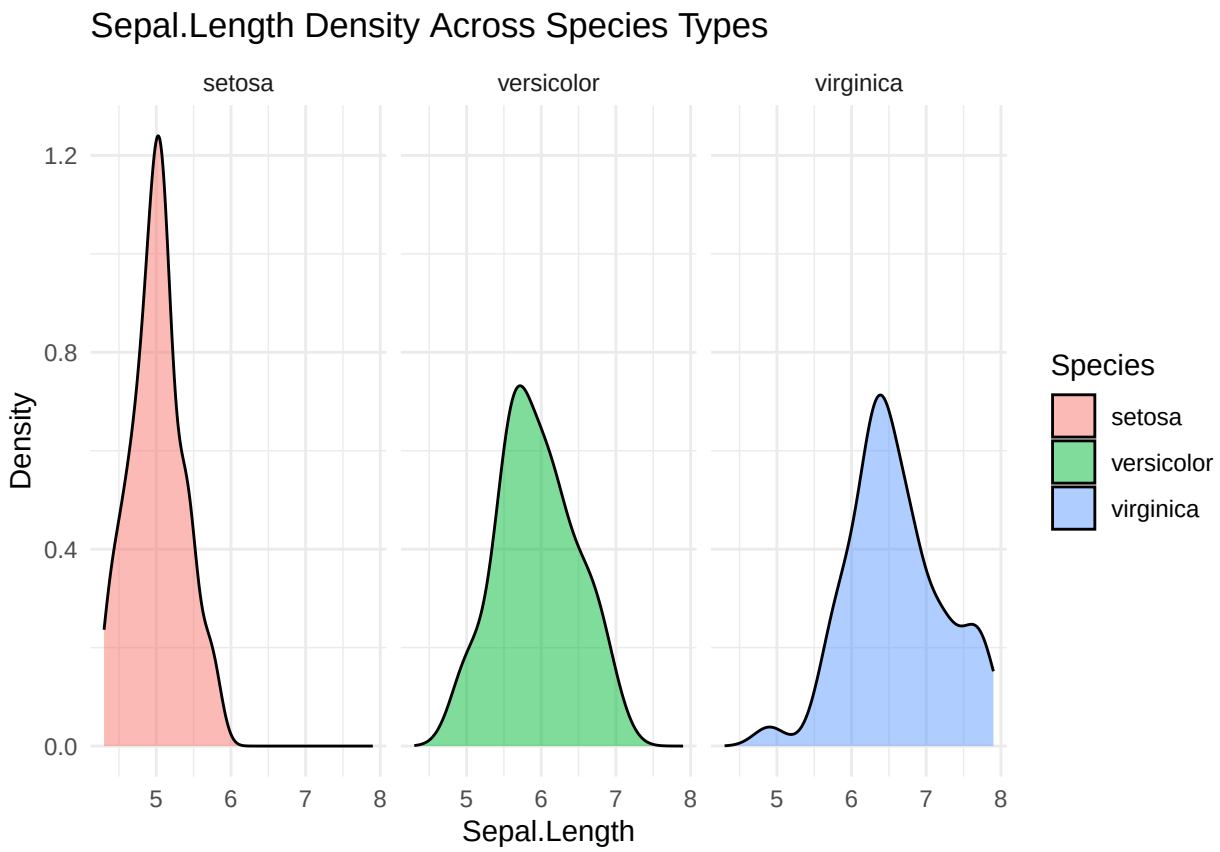


2.8.5. Faceted Density Plot (Comparing Density Distributions)

- Best for comparing smooth distributions instead of histograms.

Question: How does the density of Sepal.Length vary across different Species?

```
ggplot(iris, aes(x = Sepal.Length, fill = Species)) +
  geom_density(alpha = 0.5) +
  facet_wrap(~ Species) +  # Separate density plots for each transmission type
  ggtitle("Sepal.Length Density Across Species Types") +
  xlab("Sepal.Length") +
  ylab("Density") +
  theme_minimal()
```

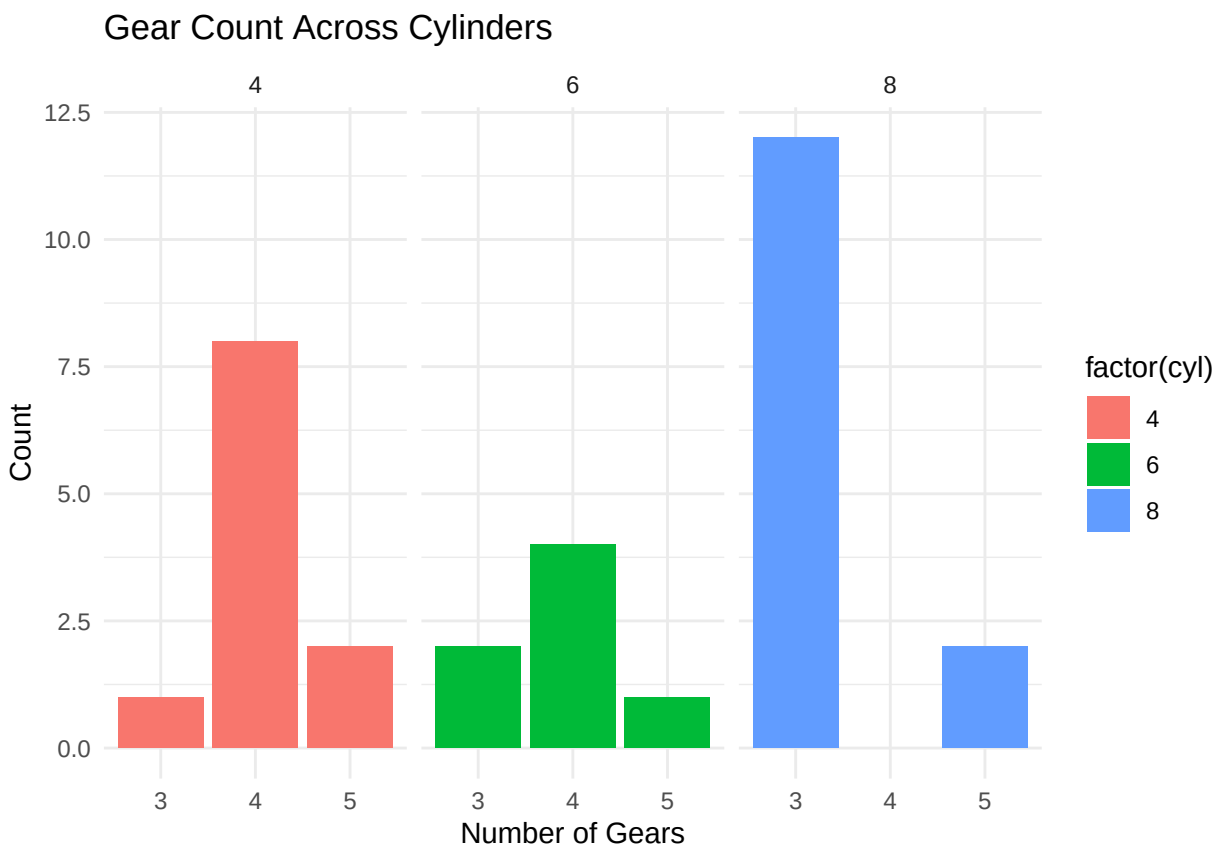


2.8.6. Faceted Bar Chart (Comparing Categorical Counts)

- Best for visualizing frequency differences across categories.
- Faceted by cyl to separate comparisons.

Question: How do gear counts vary across different cylinder types?

```
ggplot(mtcars, aes(x = factor(gear), fill = factor(cyl))) +
  geom_bar(position = "dodge") +
  facet_wrap(~ cyl) +  # Separate bar charts per cylinder type
  ggtitle("Gear Count Across Cylinders") +
  xlab("Number of Gears") +
  ylab("Count") +
  theme_minimal()
```


Gear Count Across Cylinders
4
6
8
12.5
10.0
7.5
5.0
2.5
0.0
Count
3
4
5
Number of Gears
factor(cyl)
4
6
8