# Blog 1: Data Visualizations in R

Eric Lehmphul

5/22/2022

## Why are visualizations important?

Being able to develop meaningful data visualizations is a key component for any data scientist or data analyst. Data Visualizations allow for businesses to gain insights quickly using vast amounts of data. Audiences also respond much better to visualizations than data tables. Visualizations can uncover underlying properties not previously foreseen in the data.

## Types of Visualizations

Therea are many types of plots that are used to visualize data. Typically, graphs are divided into 3 groups: univariate and bivariate, and multivariate.

- Univariate plots:
  - Histogram
  - Density Plots
  - Boxplot
  - Bar Plot
- Bivariate
  - Scatterplot
  - Line graph
  - Grouped Bar Plot
  - Grouped Boxplot
- Multivariate
  - Correlation Matrix
  - Heatmap
  - Grouping
  - Faceted Graphs

This blog will focus on creating scatterplots, correlation matrix, histograms, and a pairs plot to combine scatterplots, correlation matrix, histograms, and boxplots

## How to create plots in R

The R programming language has many useful packages to enhance its functionality. The ggplot2 package uses layers to add components to the graph making the procedure extremely streamlined. GGally provides an improved version of the base R pairs function. Corrplot makes creating correlation matrices extremely simple. I will be using the iris dataset to explore the functionality of r graphing packages.

```
library(tidyverse)
library(GGally)
library(corrplot)
```

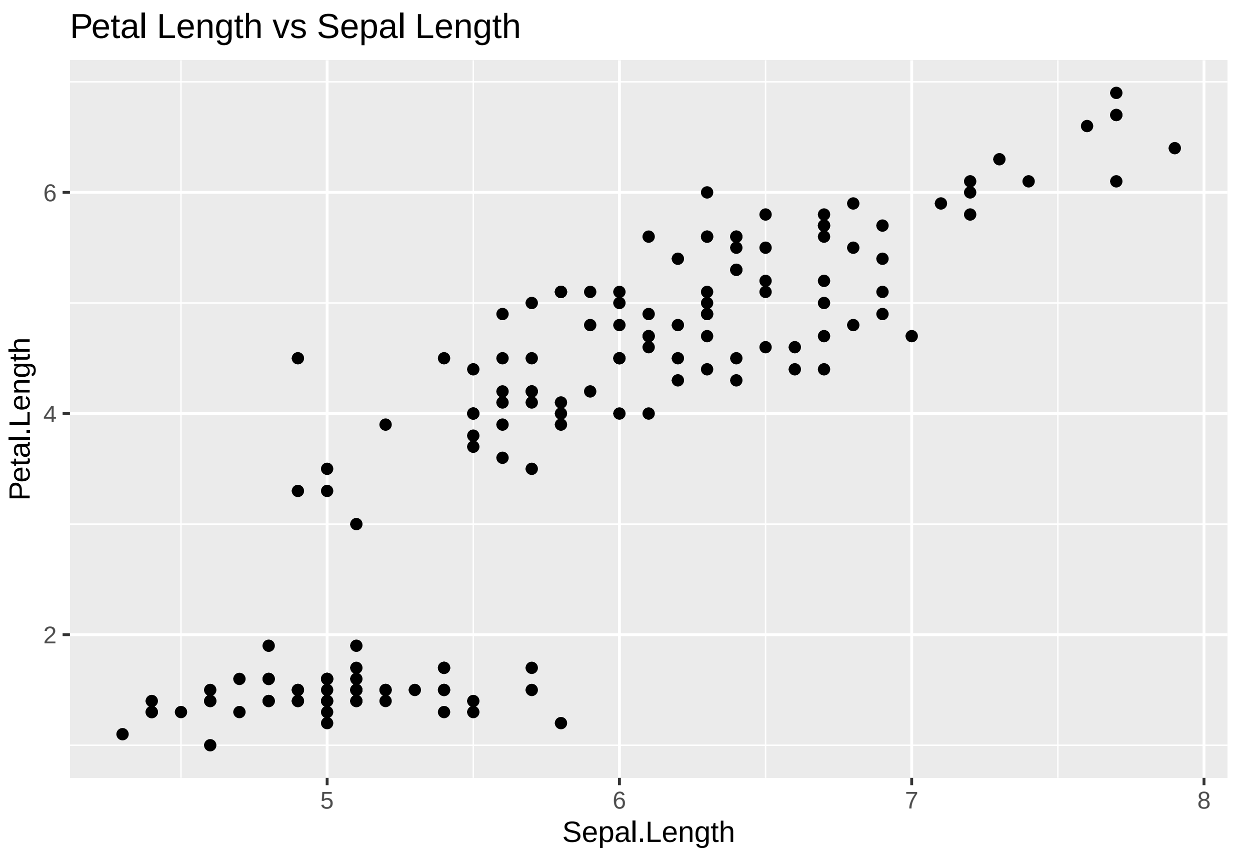
```
data("iris")
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          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          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

## Scatterplot

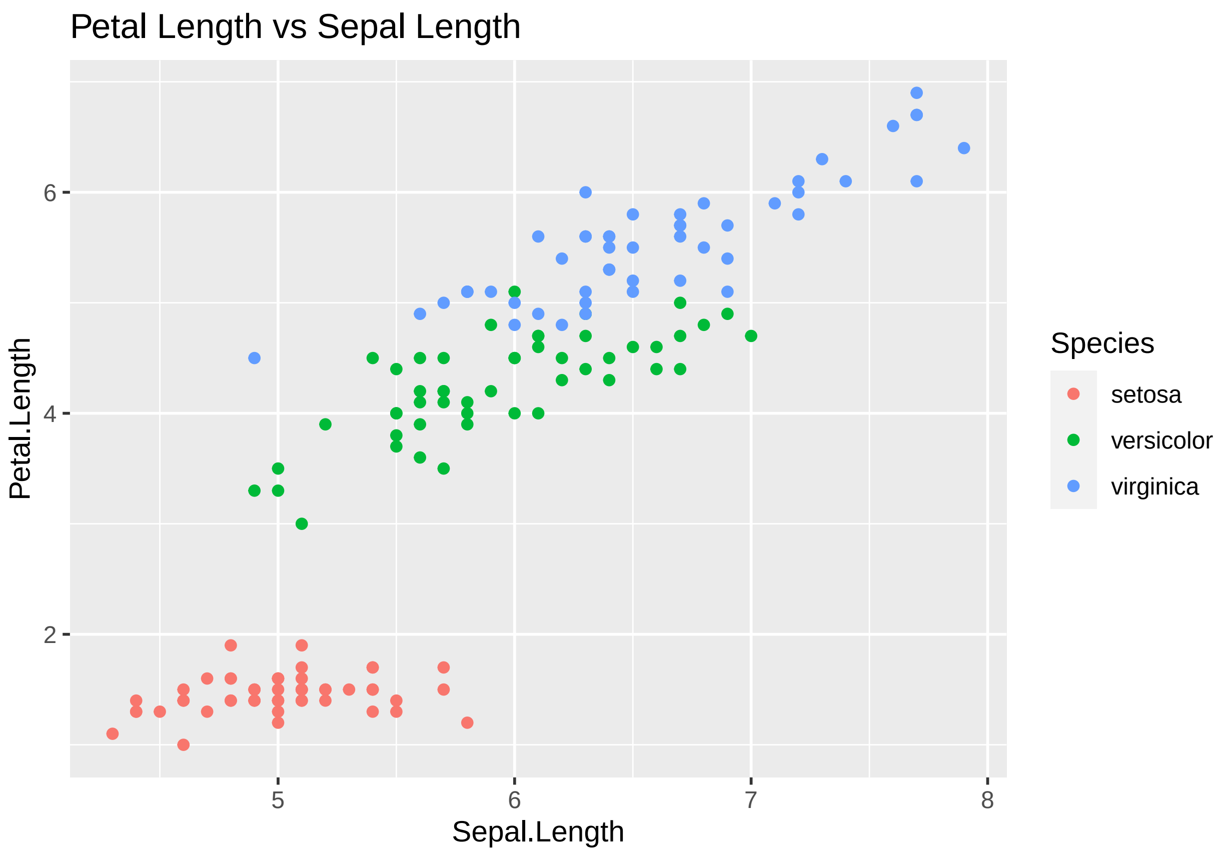
Scatterplots are helpful in identifying relationships between two numeric variables. Scatterplots are easy to produce in ggplot. The only things that need to be specified are the x and y values + geom_point.

```
iris %>%
  ggplot(aes(x = Sepal.Length, y = Petal.Length)) +
  geom_point() +
  ggtitle("Petal Length vs Sepal Length")
```



To add color to the scatterplot specify in the aes(), color = Categorical Variable.

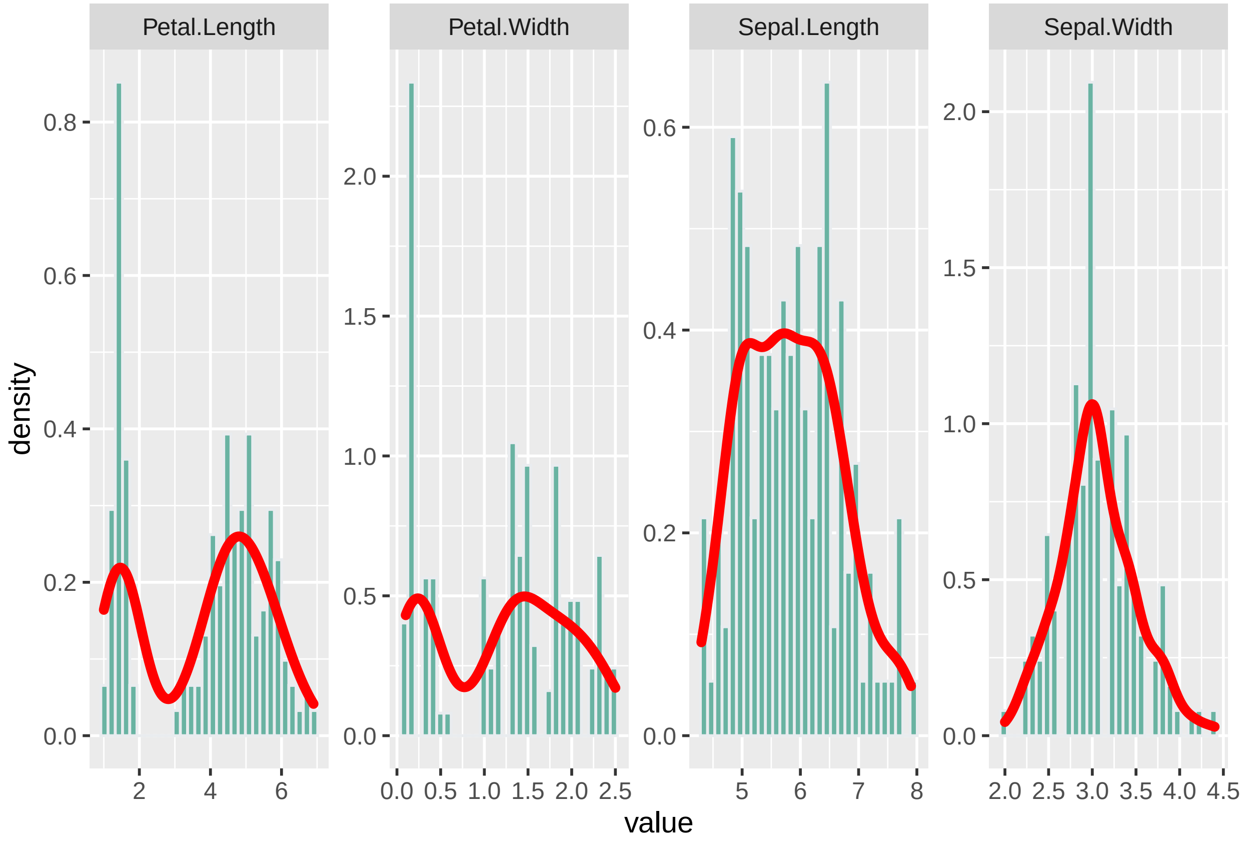
```
iris %>%
  ggplot(aes(x = Sepal.Length, y = Petal.Length, color = Species)) +
  geom_point() +
  ggtitle("Petal Length vs Sepal Length")
```



## Histogram / Density Plot

Histograms and density plots are great for looking at data distributions. The geom_histogram() function can produce histograms and the geom_density() create density plots. Below is functionality of how to combine the two visualization into 1 graph.

```
iris %>%
  gather(-c(Species), key = variable, value = value) %>%
  ggplot(., aes(x = value)) +
  geom_histogram(aes(x=value, y = ..density..), bins = 30, fill="#69b3a2", color="#e9ecef") +
  geom_density(aes(x=value), color='red', lwd = 1.75) +
  facet_wrap(~variable, scales ="free", ncol = 4)
```

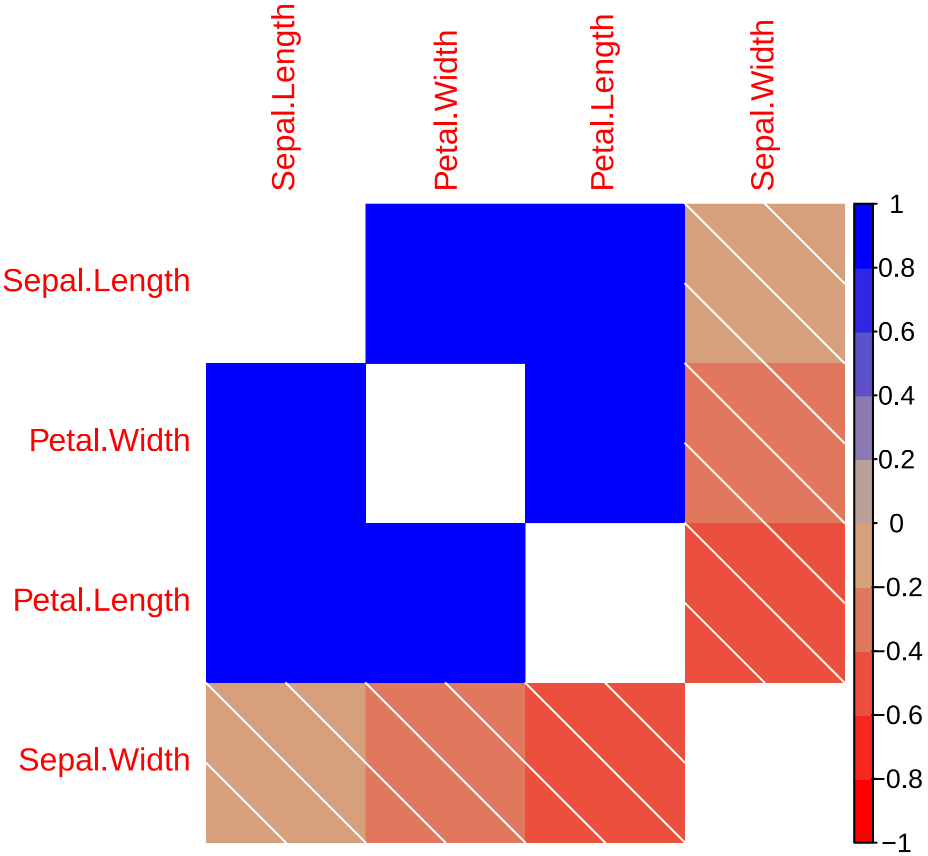


## Correlation Matrix Plot

The correlation matrix library, corrplot, effectively visualizes the correlations between numeric data. This type of plot can be leveraged to find highly correlated features or if the any features are highly correlated with the dependent variable.

```
correlation.matrix <- cor(iris[,1:4])

corrplot(correlation.matrix, method = 'shade', order = 'AOE',col= colorRampPalette(c("red","tan", "blue
```

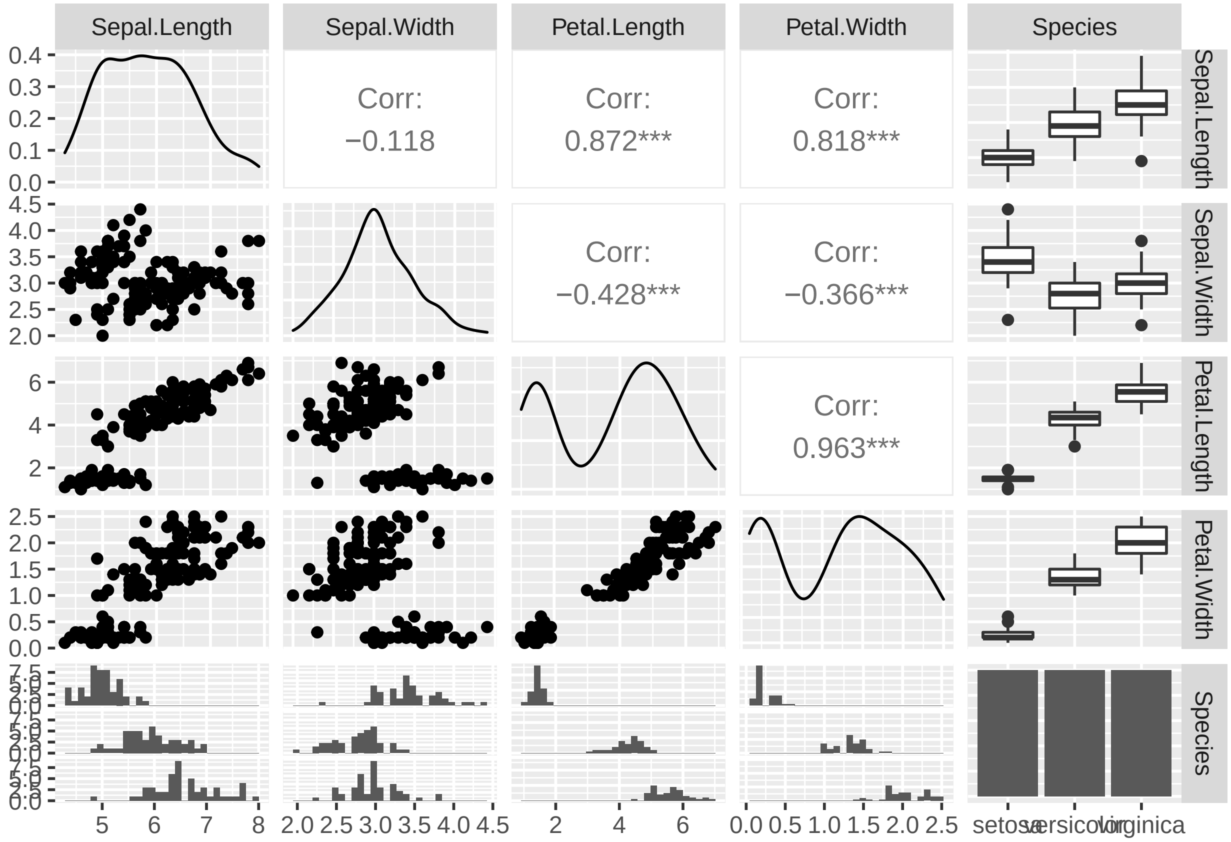


### Scatterplot / Correlation / Boxplot / Density Plot Matrix

The ggpairs function combines many different types of graphs into 1 visualization. This is a great tool for exploratory data analysis. The only critique with this function is that the more variables included in the function, the harder the visualization is to interpret.

```
ggpairs(iris)
```

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



## Data Visualization: Things to keep in mind

The goal of data visualization is to tell a story that can be understood quickly from an audience. It is best to keep visualizations simple and uncluttered, as the clutter can take away from the story in the visualization.