Interactive Visualization

Sejal Kankriya

2024-02-20

```
library(data.table)
library(dplyr)
library(tidyr)
library(plotly)
library(lubridate)
```

Iris Dataset

> "The Iris flower data set or Fisher's Iris data set is a multivariate data set introduced by the British statistician and biologist Ronald Fisher in his 1936 paper The use of multiple measurements in taxonomic problems as an example of linear discriminant analysis" https://en.wikipedia.org/wiki/Iris_flower_data_set (https://en.wikipedia.org/wiki/Iris_flower_data_set)

```
# Read the iris.csv file

iris_df <- fread("iris.csv", stringsAsFactors = TRUE)
```

```
head(iris_df)
```

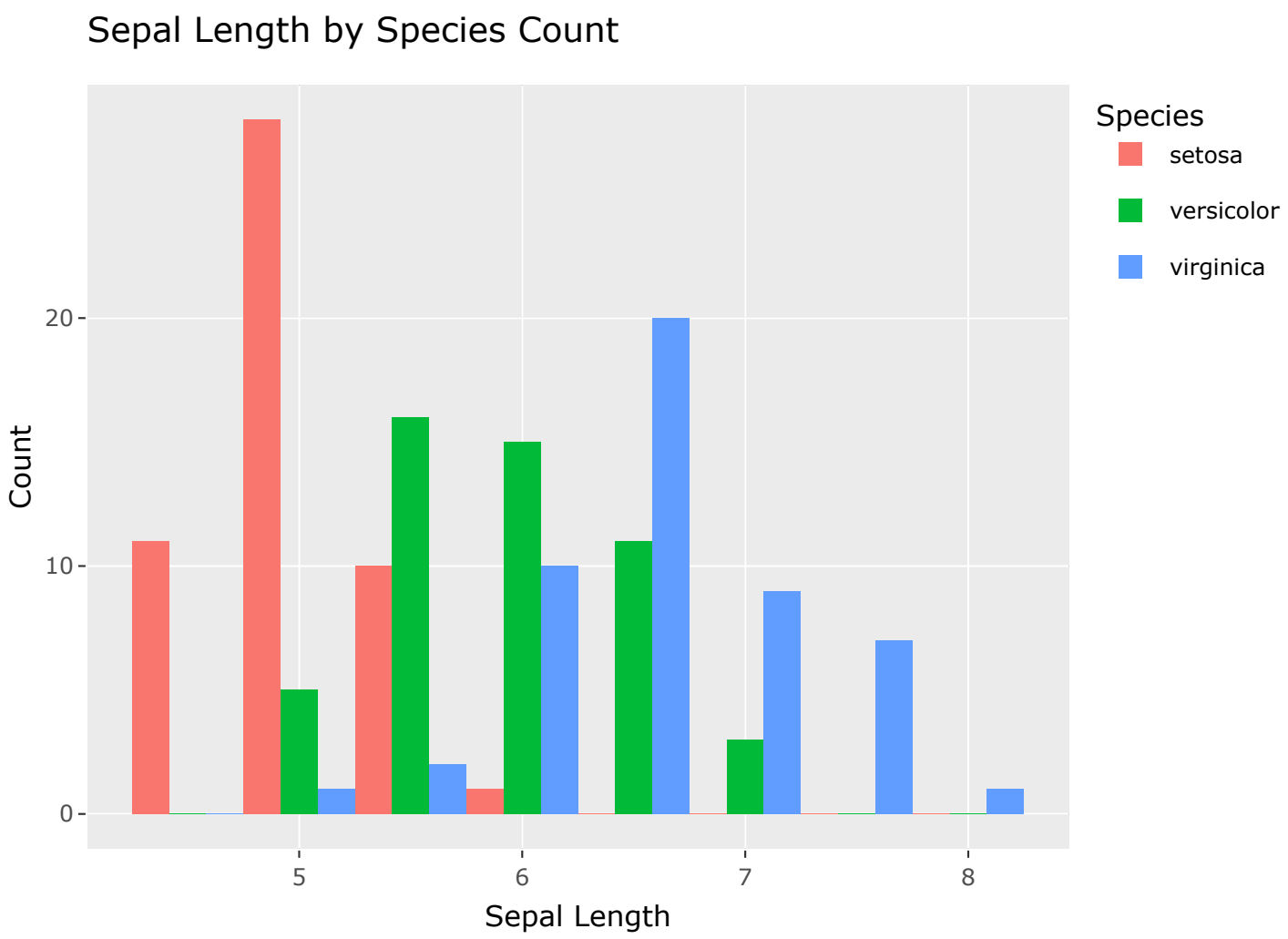
Sepal.Length <dbl>	Sepal.Width <dbl>	Petal.Length <dbl>	Petal.Width <dbl>	Species <fct>
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa

6 rows

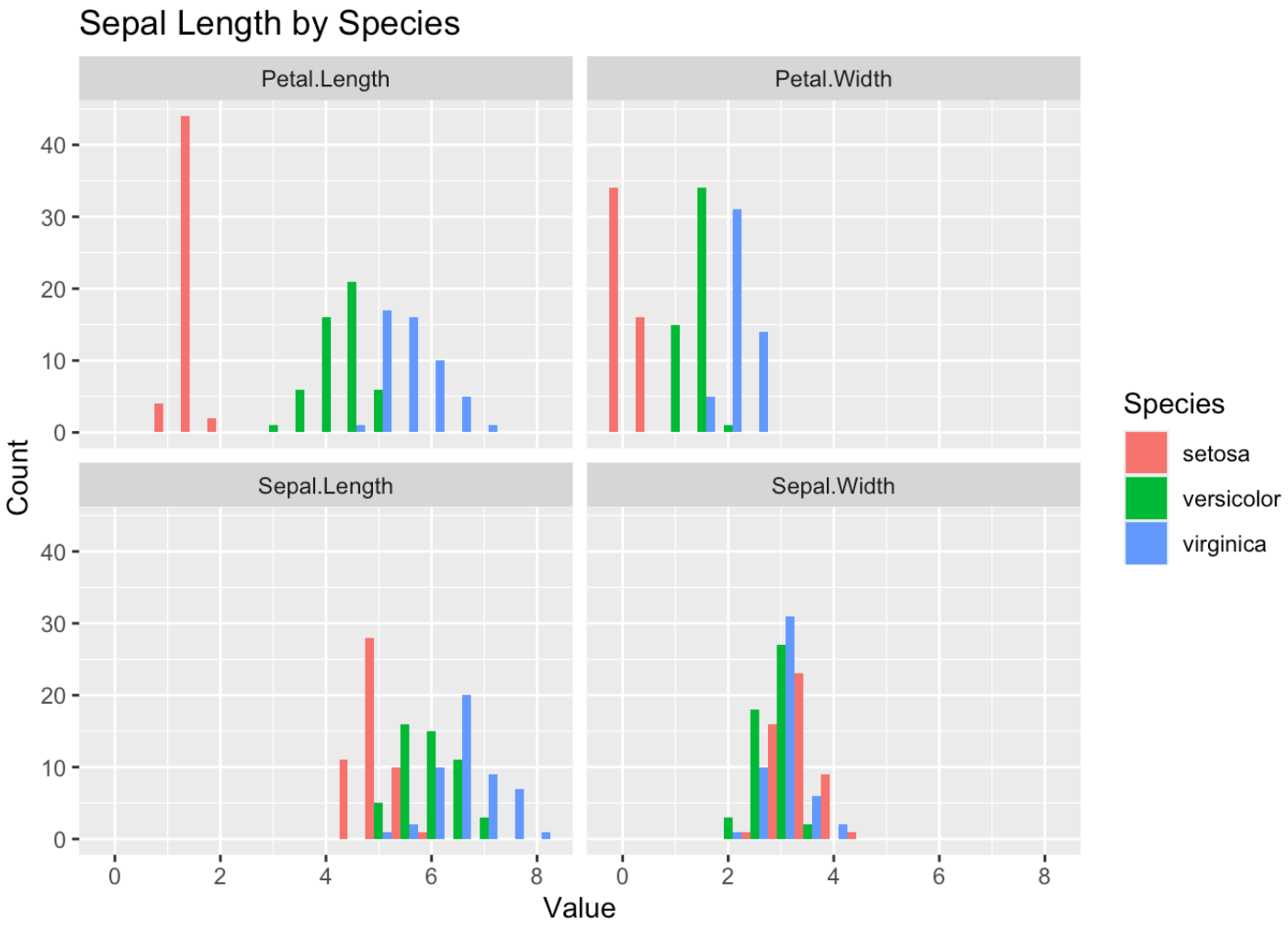
```
# Build histogram plot for Sepal.Length variable for each species using plot_ly

plot_ly(iris_df, x = ~Sepal.Length, color = ~Species, type = "histogram") %>%
  layout(title = "Sepal Length by Species",
    xaxis = list(title = "Sepal Length",
                 tickvals = c(4, 5, 6, 7, 8)),
    yaxis = list(title = "Count"))
```

```
# Repeat previous plot with ggplot2 and convert it to plotly with ggplotly

# Plotting the histogram with gglpot2
p <- ggplot(iris_df, aes(x = Sepal.Length, fill = Species)) +
  geom_histogram(binwidth = 0.5, position="dodge") +
  labs(title = "Sepal Length by Species Count",
       x = "Sepal Length",
       y = "Count")


# converting it to plotly with ggplotly
toWebGL(ggplotly(p))
```

```
## Warning in verify_webgl(p): The following traces don't have a WebGL equivalent:
## 1, 2, 3
```



```
# Create facet 2 by 2 plot with histograms similar to previous but for each metric

long_conv <- iris_df %>%
  gather(key = "metric", value = "value",-Species)


ggplot(long_conv, aes(x = value, fill = Species)) +
  geom_histogram(bins=30, binwidth = 0.5, position="dodge") +
  facet_wrap(~metric, nrow = 2) +
  labs(title = "Sepal Length by Species",
       x = "Value",
       y = "Count")
```

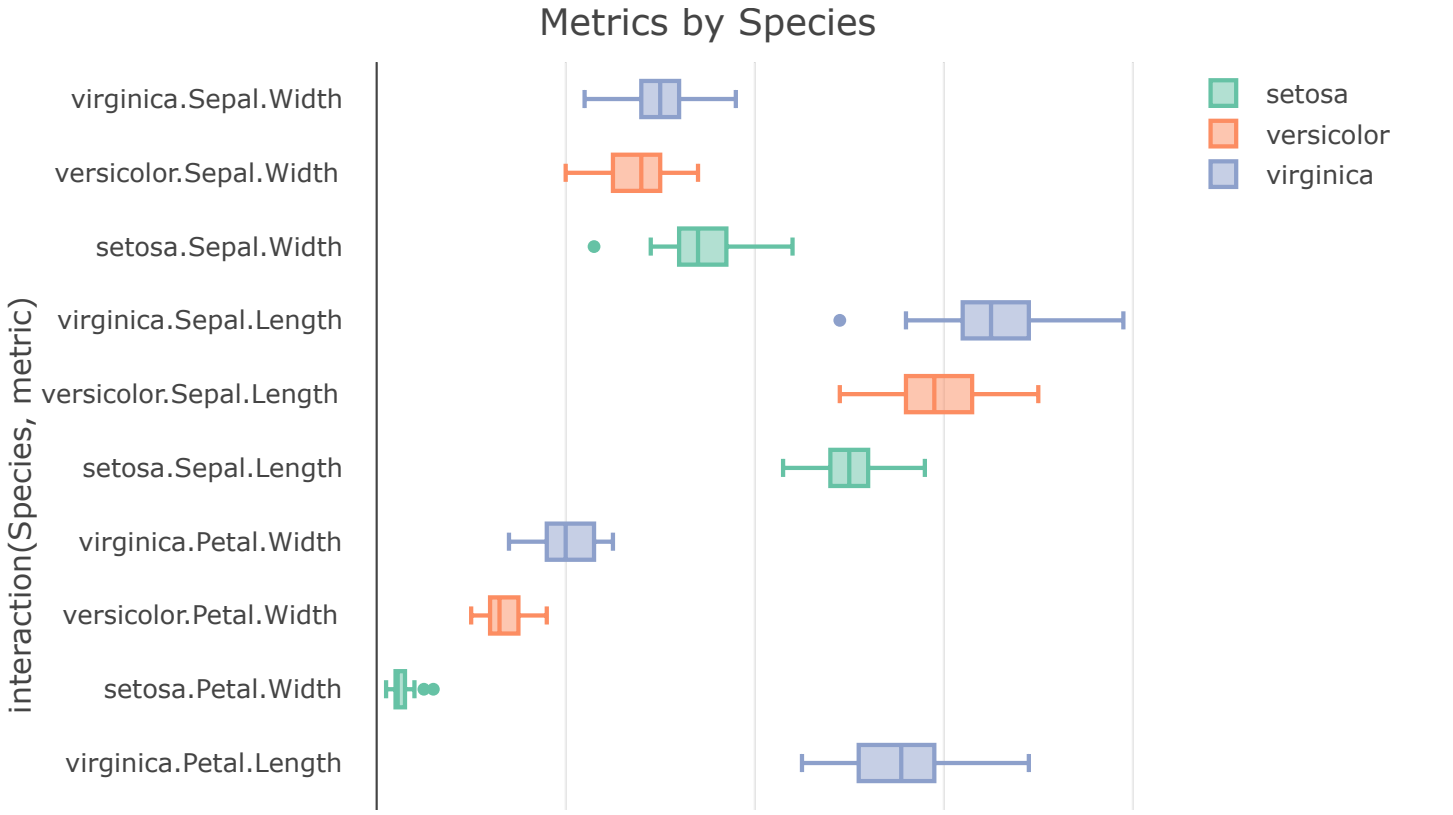


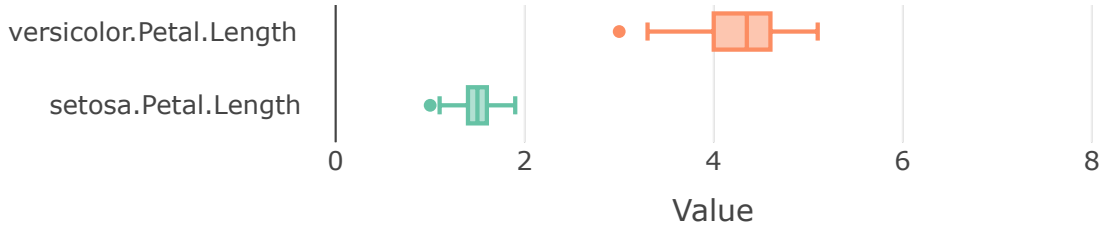
The histograms indicate that the metrics Petal.Length and Petal.Width provide the best species separations among the four metrics in the iris dataset. Petal.Length, in particular, shows a clear distinction between the Setosa species and the other two species, whereas the Versicolor and Virginica species have some overlap. Similarly, Petal.Width shows a clear separation between Setosa and the other two species, with less overlap between the Versicolor and Virginica species than Sepal.Length and Sepal.Width. Sepal.Length and Sepal.Width, on the other hand, show more overlap between the species, making differentiation more difficult.

```
# Repeat above plot but using box plot

plot_ly(long_conv, x = ~ value, y = ~ interaction(Species, metric), color = ~ Species) %>%
  add_boxplot()%>%
  layout(title = "Metrics by Species",
         xaxis = list(title = "Value"),
         yaxis = list(title = "interaction(Species, metric)"),
         facet_col = ~Species + metric)
```

```
## Warning in Ops.factor(Species, metric): '+' not meaningful for factors
```

```
## Warning: 'layout' objects don't have these attributes: 'facet_col'
## Valid attributes include:
## '_deprecated', 'activeshape', 'annotations', 'autosize', 'autotypenumbers', 'calendar', 'clickmode', 'coloraxis', 'colorscale', 'colorway', 'computed', 'datarevision', 'dragmode', 'editrevision', 'editType', 'font', 'geo', 'grid', 'height', 'hidesources', 'hoverdistance', 'hoverlabel', 'hovermode', 'images', 'legend', 'mapbox', 'margin', 'meta', 'metasrc', 'modebar', 'newshape', 'paper_bgcolor', 'plot_bgcolor', 'polar', 'scene', 'selectdirection', 'selectionrevision', 'separators', 'shapes', 'showlegend', 'sliders', 'smith', 'spikedistance', 'template', 'ternary', 'title', 'transition', 'uirevision', 'uniformtext', 'updatemenus', 'width', 'xaxis', 'yaxis', 'barmode', 'bargap', 'mapType'
```

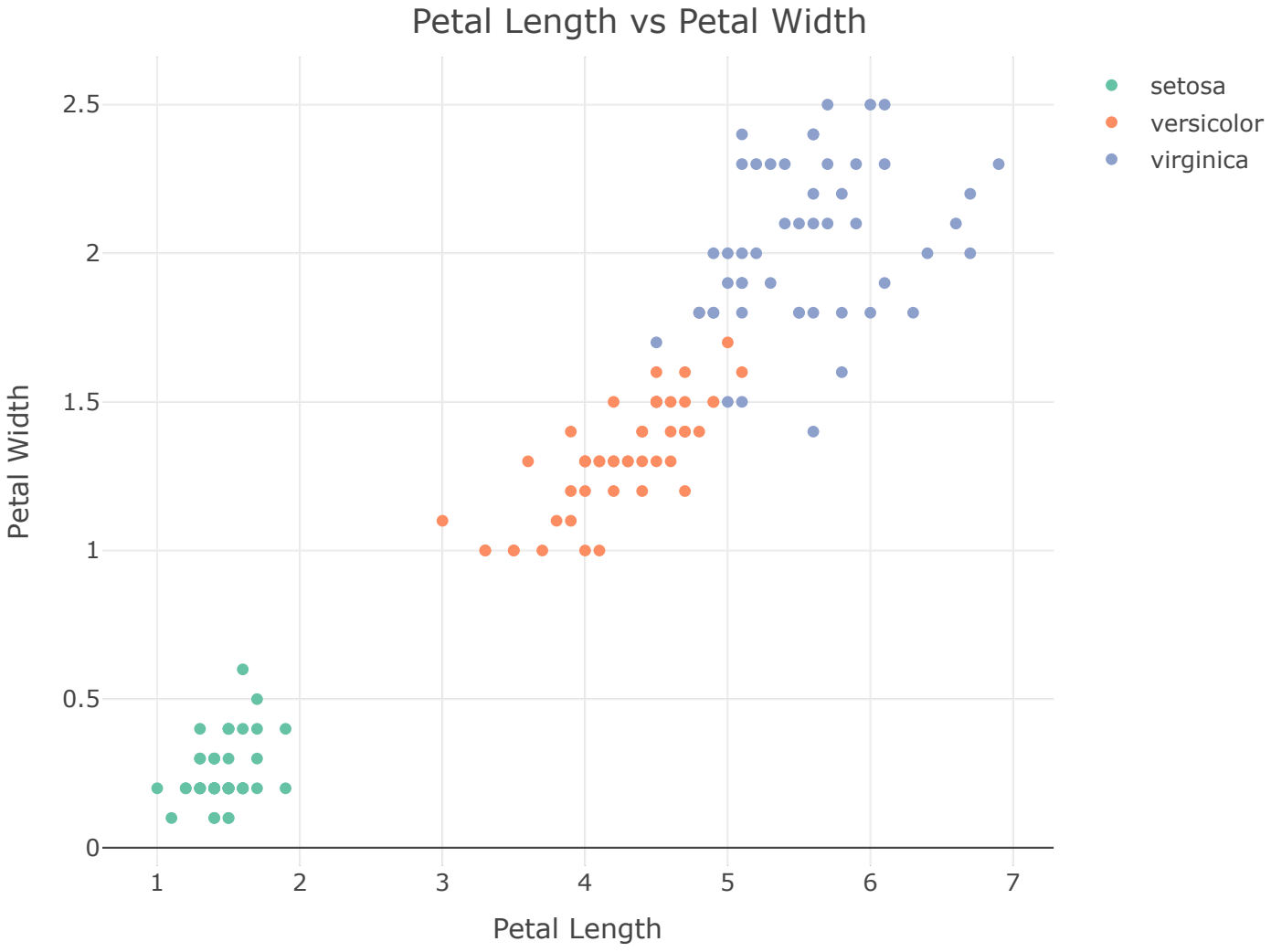




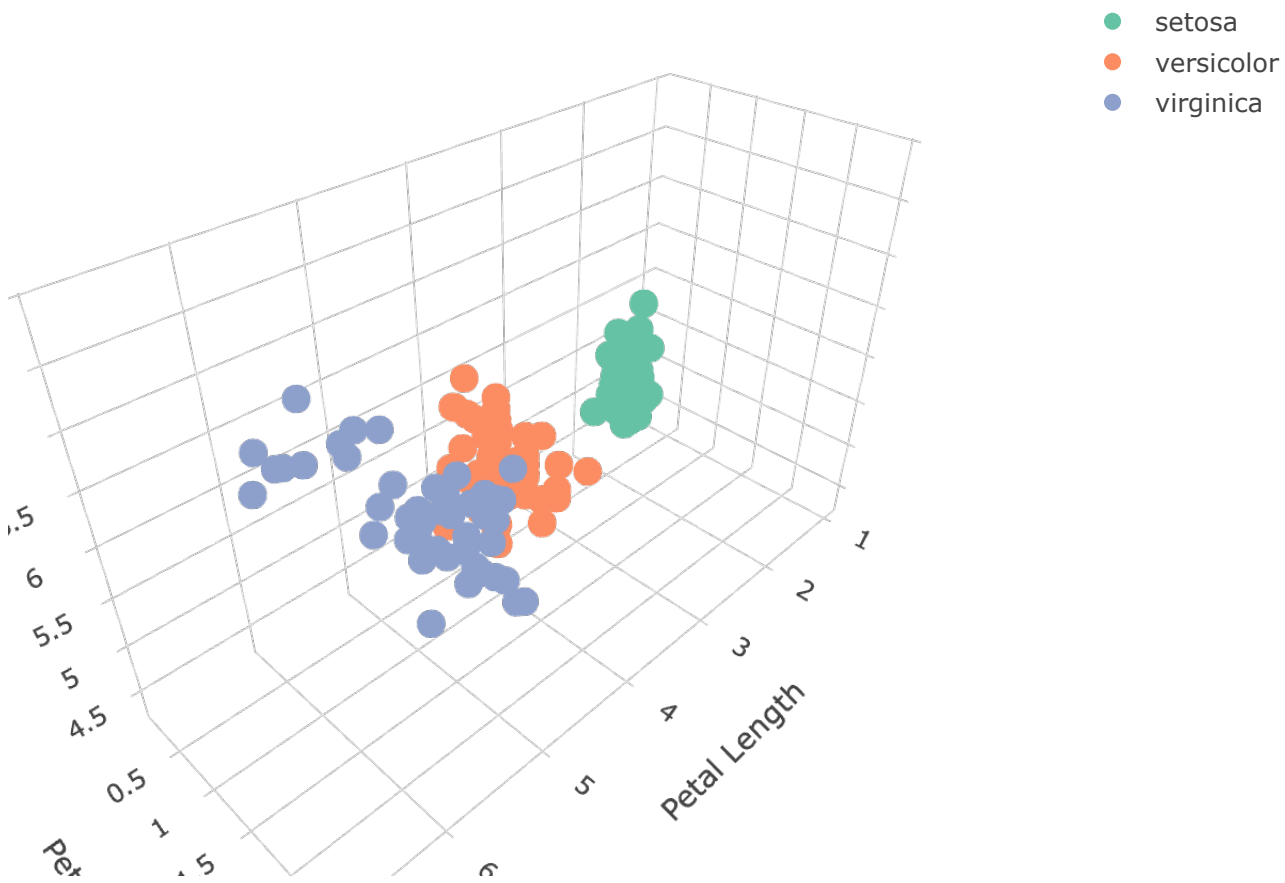

```
# Choose two metrics which separates species the most and use it to make scatter plot
# color points by species
```

Based on the previous boxplot, we can see that Petal Length and Petal Width perform the best seperation of Species. We'll use these two to plot the scatterplot below.

```
plot_ly(iris_df, x = ~Petal.Length, y = ~Petal.Width, color = ~Species,
              type = "scatter", mode = "markers") %>%
  layout(title = "Petal Length vs Petal Width",
         xaxis = list(title = "Petal Length"),
         yaxis = list(title = "Petal Width"))
```



```
# Choose three metrics which separates species the most and use it to make 3d plot
# color points by species

plot_ly(iris_df, x = ~Petal.Length, y = ~Petal.Width, z = ~Sepal.Length,
        color = ~Species, type = "scatter3d", mode = "markers") %>%
  layout(scene = list(xaxis = list(title = "Petal Length"),
                      yaxis = list(title = "Petal Width"),
                      zaxis = list(title = "Sepal Length")))
```



According to the visualizations, the metrics “Petal Length,” “Petal Width,” and “Sepal Length” appear to show the best separation between the three iris species. The histograms and boxplots show that the distribution of these metrics differs significantly between the three species. Furthermore, the scatterplot demonstrates clear clustering of the three species based on these metrics. Overall, these findings suggest that these metrics can be used to differentiate between the three iris species.