Machine Learning with Linear Regression on Iris Dataset: Predicting Petal Length On Species and Three Numerical Variables Including Sepal Length, Sepal Width and Petal Width

by Joshua Nguyen

Abstract Linear Regression is a statistical machine learning algorithm and/or model that analyzes the linear relationship between a response variable and its corresponding dependent variables. This is a machine learning project that uses machine learning to compute a linear regressional model on the Iris dataset completed with R in RStudio. It will predict petal length based on species, petal width, sepal width and sepal length.

Import Libraries

```
> library(corrplot) # Correlation Matrix Plots
> library(corrgram) # Alternative Correlation Matrix Plots
> library(dplyr) # Data Manipulation
> library(ggplot2) # Data Visualization
> library(caTools) # Training and Testing
```

Import Dataset

This project uses the *Iris* dataset provided in R.

```
> 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
> my_data <- iris[,c(1,2,3,4)]
> head(my_data)
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1          5.1         3.5          1.4         0.2
2          4.9         3.0          1.4         0.2
3          4.7         3.2          1.3         0.2
4          4.6         3.1          1.5         0.2
5          5.0         3.6          1.4         0.2
6          5.4         3.9          1.7         0.4
```

Describing the Data

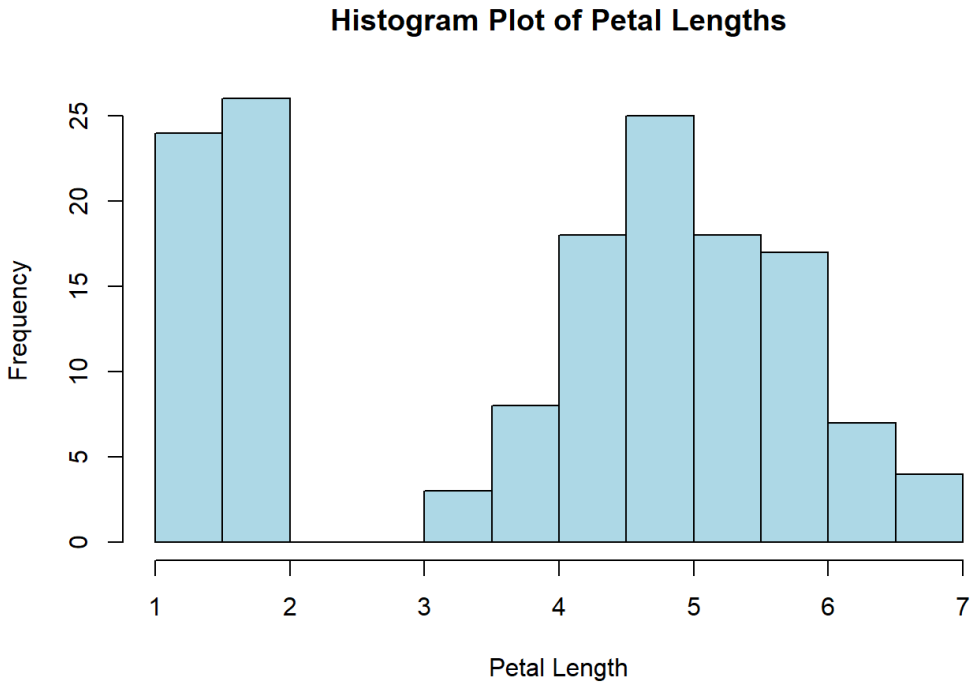
We begin our EDA by checking for null values:

```
> any(is.na(my_data))
[1] FALSE
```

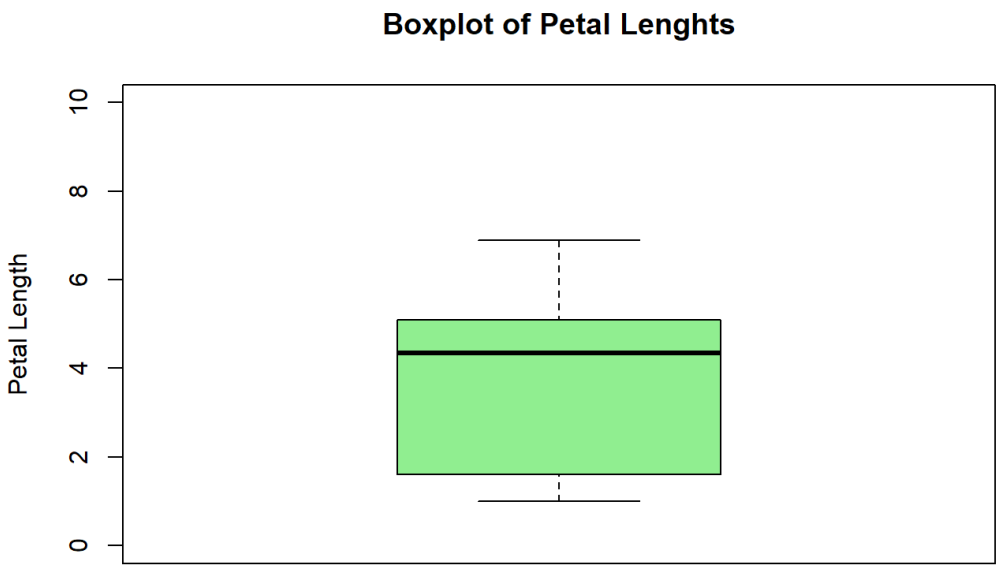
The **False** result indicates that there are no Null or missing values in our dataset - great! Next, lets get some summary statistics.

```
> summary(iris)
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width          Species
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100   setosa     :50
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300   versicolor:50
 Median :5.800   Median :3.000   Median :4.350   Median :1.300   virginica :50
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
```

```
# Histogram Plot of Petal Lengths
> hist(my_data$Petal.Length, xlab = 'Petal Length', main = 'Histogram Plot of
↪  Petal Lengths', right = F, breaks = 10, col = 'light blue')
```



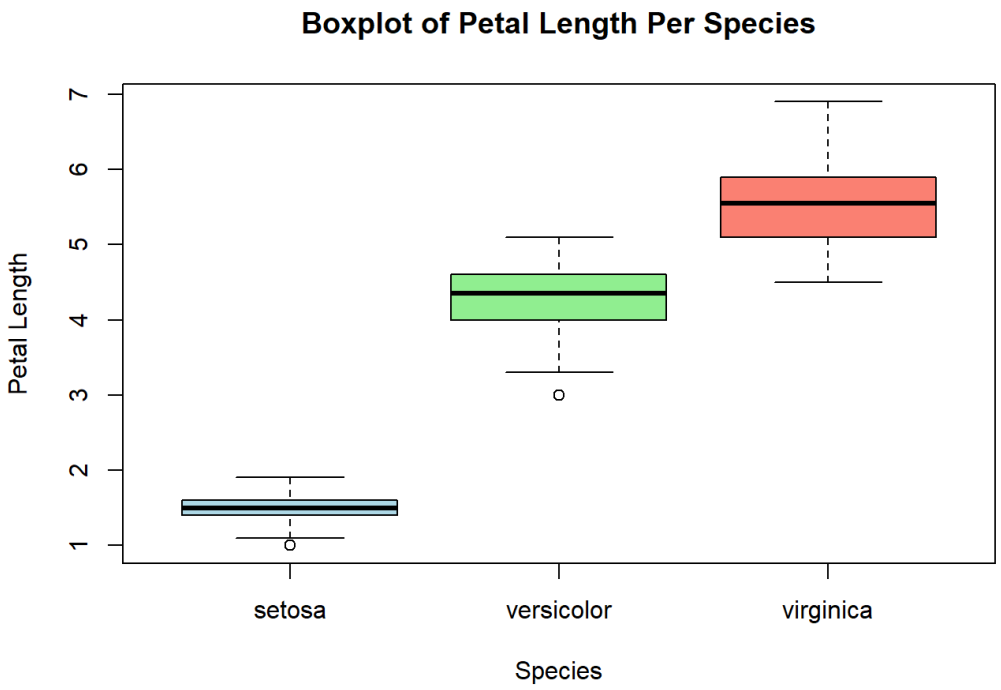
```
# Boxplot Plot of Petal Lengths
> boxplot(my_data$Petal.Length, ylab = 'Petal Length', main = 'Boxplot of Petal
↪  Lenghts', col = ' light green', ylim = c(0,10))
```




```
# Boxplot Plot of Petal Lengths By Species
> boxplot(my_data$Petal.Length ~ iris$Species, xlab = 'Species', ylab = 'Petal
↪  Length', main = 'Boxplot of Petal Length Per Species', col = c('light blue',
↪  'light green', 'salmon'))
```

Here we see that the Virginica Species has the longest petal length, and the Setosa Species has the shortest petal length on average. Veriscolor Species has intermediate petal length. Lets quantify this:

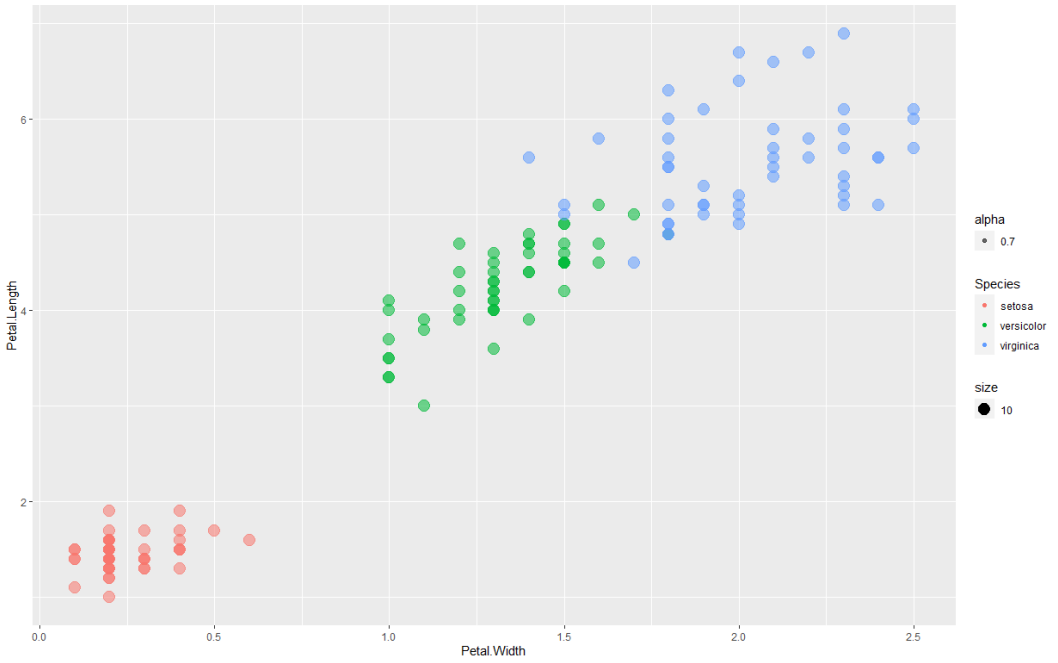
```
# Getting mean and standard deviations
> setosa <- iris[iris$Species == "setosa",]
> versicolor <- iris[iris$Species == "versicolor",]
> virginica <- iris[iris$Species == "virginica",]

# Setosa
> mean(setosa$Petal.Length)
[1] 1.462
> sd(setosa$Petal.Length)
[1] 0.173664

# Veriscolor
> mean(versicolor.Length)
[1] 4.26
> sd(versicolor$Petal.Length)
[1] 0.469911

# Virginica
> mean(virginica.Length)
[1] 5.552
> sd(virginica$Petal.Length)
[1] 0.5518947
```



Visualization with GGPlot

We can make a scatterplot with *ggplot* to see how the response variable (petal length) of our data behaves based on an input variable (petal width) separated by species.

```
# Scatter Plot of Petal Length by Species
> ggplot(data=iris, aes(x=Petal.Width, y=Petal.Length)) +
  geom_point(aes(color=Species, size=10, alpha=0.7))
```



Separating and adding colors to the three species, we can visualize how petal width and petal length are related to each other. There is some positive linear correlation in all three species type. Next, we can verify the **correlation** between the attributes.

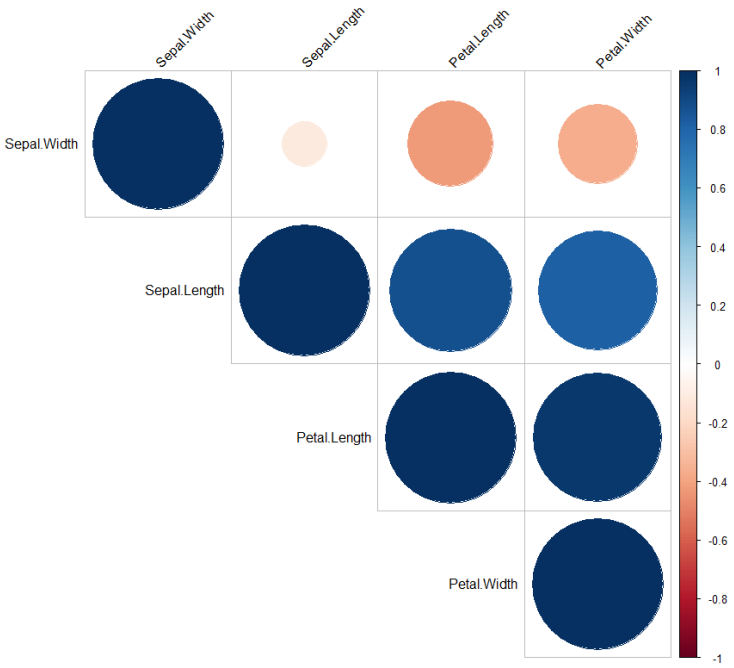
Correlation Matrix

```
# Scatter Plot of Petal Length by Species
> res <- cor(my_data, use = "complete.obs")
> res
             Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length    1.0000000  -0.1175698    0.8717538   0.8179411
Sepal.Width    -0.1175698   1.0000000   -0.4284401  -0.3661259
Petal.Length    0.8717538  -0.4284401    1.0000000   0.9628654
Petal.Width     0.8179411  -0.3661259    0.9628654   1.0000000
```

The *complete.obs* line tells R to omit rows with any Null or *Na* values. From above, we were able to ascertain that our dataset contained zero null values, but I have included it here for good standard of care. Lets visualize our correlation matrix.

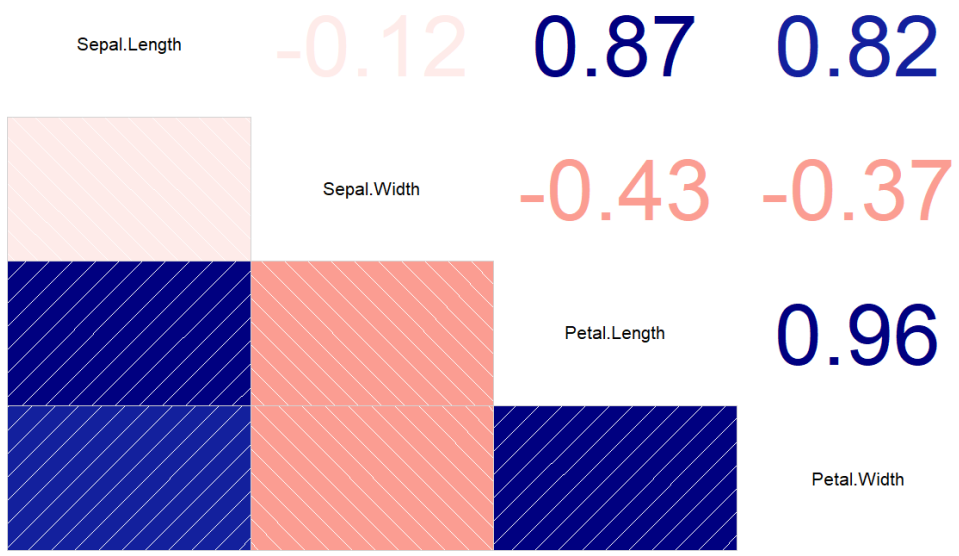
Visualizing Correlation Matrix

```
# Correlation Plot with Corrplot
> corrplot(res, type = "upper", order = "hclust", tl.col = "black", tl.srt = 45)
```



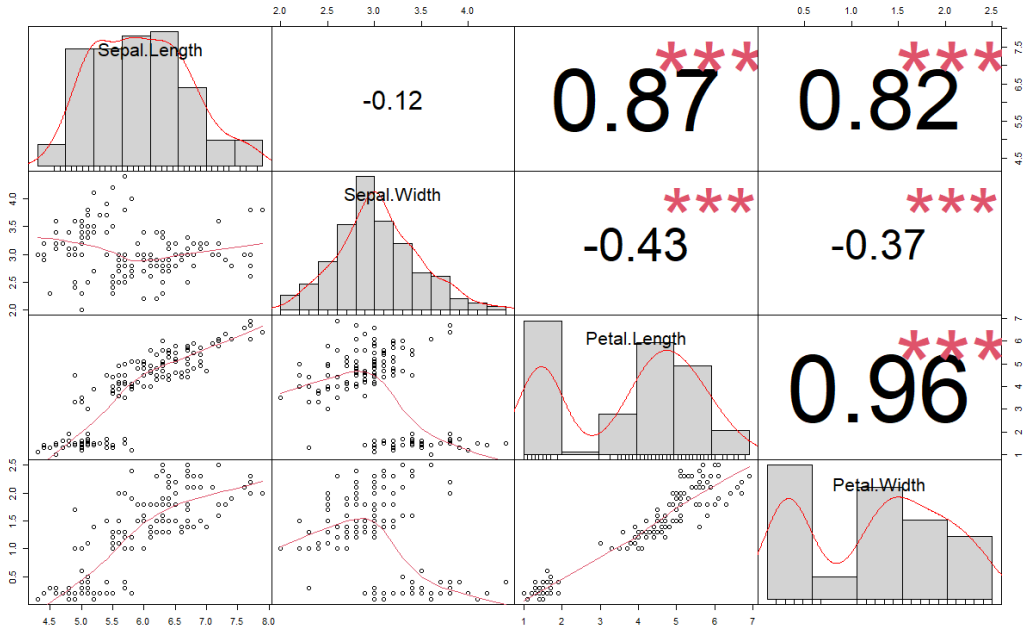
We can use the *corrgram* library to visualize a correlation plot with its significance levels.

```
# Correlation Plot With Corrgram
> corrgram(iris, lower.panel =panel.shade, upper.panel = panel.cor)
```



We can display a chart of the correlation matrix using the *PerformanceAnalytics* dataset. This will also help us visualize the correlation of variables in our dataset.

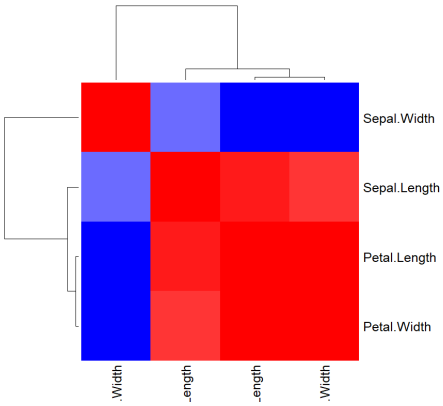
```
# Performance Analytics Chart
> install.packages("PerformanceAnalytics")
> library("PerformanceAnalytics")
> chart.Correlation(my_data, histogram = TRUE, pch = 19)
```



In the above plot, the distribution of each variable in the Iris dataset is shown on the diagonal. The plots left of the diagonal shows the Bivariate Scatter Plots with a fitted line. The plots on top of the diagonal represent the correlation and the significance levels as asterik marks with having more marks representing a p-value incrementally getting smaller as you increase the number of stars.

Lastly, we can use heatmaps to visualize the correlation.

```
# Heatmaps
> col <- colorRampPalette(c('blue', 'white', 'red'))(20)
heatmap(x=res, col = col, symm =  TRUE)
> chart.Correlation(my_data, histogram = TRUE, pch = 19)
```



Model Training and Evaluation

First, we will split our dataset into two parts: train and test. 70 % of our data will be invested into the training phase of our model and the remaining 30 % will be invested into the testing phase for model evaluation.

Before we began splitting the data, we will set a random seed. It is important to know that splitting our data into training and testing sets is a random process. In R, generating random numbers means that we are establishing a *pseudorandom number* which requires a seed in order to initialize. Setting the random seed at a number will help increase the how reproducible the outcome is at the end of the model.

```
# Setting Random Seed
> set.seed(42)
```

```
# Train/Test Splits
> SampleSplit <- sample.split(Y=iris$Petal.Length, SplitRatio = 0.7)
> trainSet <- subset(x = iris, SampleSplit == TRUE)
> testSet <- subset(x = iris, SampleSplit == FALSE)
> model <- lm(Petal.Length ~., data = trainSet)
> summary(model)

Call:
lm(formula = Petal.Length ~ ., data = trainSet)

Residuals:
     Min       1Q   Median       3Q      Max
-0.62314 -0.14947 -0.02203  0.15792  0.58620

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)       -1.08778    0.28857  -3.770 0.000277 ***
Sepal.Length       0.64044    0.05460  11.731  < 2e-16 ***
Sepal.Width       -0.24529    0.08754  -2.802 0.006100 **
Petal.Width        0.72664    0.13317   5.456 3.53e-07 ***
Speciesversicolor  1.25859    0.18995   6.626 1.76e-09 ***
Speciesvirginica   1.69416    0.27053   6.262 9.57e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2425 on 100 degrees of freedom
```

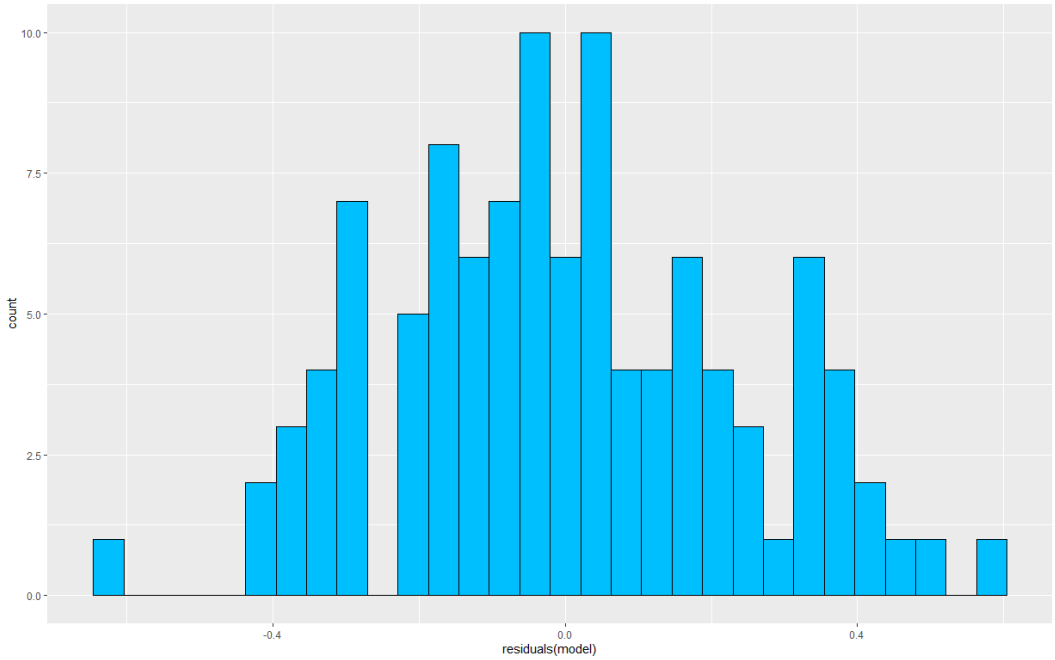
```
Multiple R-squared:  0.982,        Adjusted R-squared:  0.9811
F-statistic:  1088 on 5 and 100 DF,  p-value: < 2.2e-16
```

Notice that we are using all predictors for our model training. That is, we want to predict the petal length attributes as a linear combination of all the other input attributes or variables. Moreover, also note that all of our predictors are statistically significant!

Next, we will check if the variances are normally distributed - we can do this by plotting a histogram of the variances.

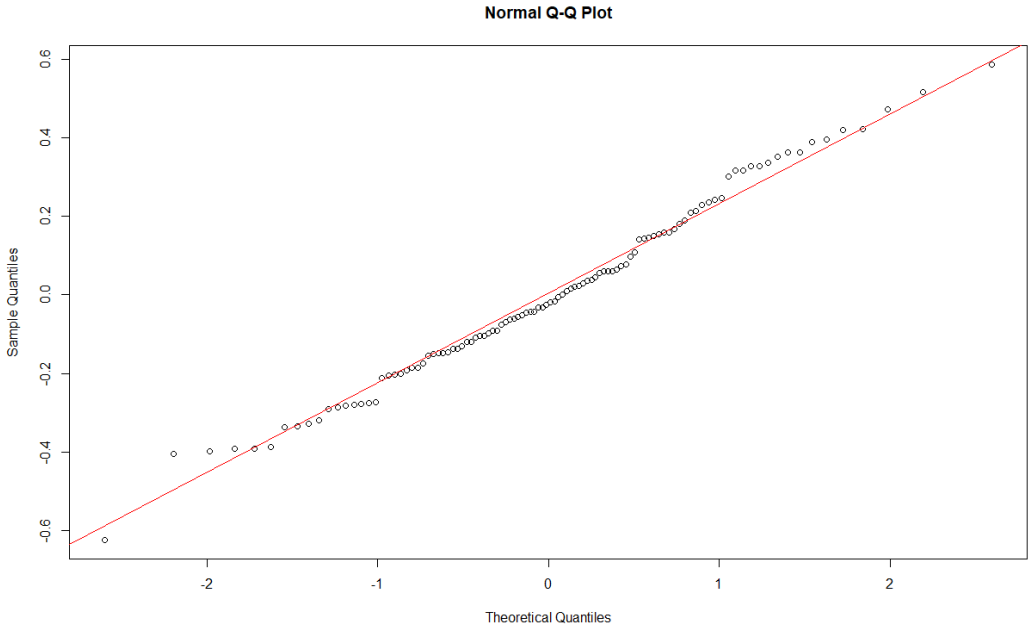
Plotting Model Residuals

```
# Residuals Plot
> modelResiduals <- as.data.frame(residuals(model))
> ggplot(modelResiduals, aes(residuals(model))) +
↪  geom_histogram(fill='deepskyblue', color='black')
```



We see here that our residuals fit a bell curve really well and can say with strong assumption that our variances are normally distributed. Another way we can confirm normality is by running a QQ Plot of the residuals.

```
# QQ Plot
> qqnorm(model$residuals)
> qqline(model$residuals, col = 'red')
```



Making Predictions

In order to start making and evaluating predictions, we will create a dataframe of actual and predicted values.

```
# Making Predictions
> predictions <- predict(model, testSet)
> modelEval <- cbind(testSet$Petal.Length, predictions)
> colnames(modelEval) <- c('Actual', ' Predicted')
> as.data.frame(modelEval)
    Actual  Predicted
1      1.4   1.465284
2      1.4   1.459840
4      1.5   1.243179
7      1.4   1.242256
10     1.5   1.362647
11     1.5   1.608358
15     1.2   1.790948
21     1.7   1.681945
25     1.9   1.297680
26     1.6   1.523884
28     1.5   1.529328
37     1.3   1.721460
39     1.3   1.139619
47     1.6   1.391697
50     1.4   1.450297
54     4.0   4.073701
73     4.9   4.682324
74     4.7   4.262658
80     3.5   3.910211
83     3.9   4.095054
86     4.5   4.342096
88     4.4   4.586054
89     4.1   3.966043
92     4.6   4.358928
94     3.3   3.535489
95     4.2   4.039630
98     4.3   4.374837
107    4.5   4.366606
109    5.8   5.592064
```

```
112    5.3   5.423538
113    5.5   5.751455
118    6.7   6.204285
125    5.7   5.613825
126    6.0   5.740582
134    5.1   5.044309
135    5.6   4.892615
136    6.1   6.473180
137    5.6   5.551111
139    4.8   5.021111
142    5.1   5.936298
144    5.9   5.847725
146    5.2   5.832739
147    5.0   5.408551
149    5.4   5.414403
```

Errors

To evaluate how good our model fits the data, we will calculate two metrics: **Means Squared Error** and **Root-Mean-Square Error.**

```
# MSE
> modelEval <- as.data.frame(modelEval)
> MSE <- mean((modelEval$Actual - modelEval$` Predicted`)^2)
> MSE
[1] 0.09531731

# RMSE
> RMSE <- sqrt(MSE)
> RMSE
[1] 0.308735
```

Conclusion

Overall, we created the following Linear Regression Models for the Iris dataset that predicts petal length based off of three numerical variables including petal width, sepal length and sepal width.

Setosa Species

Petal.Length = (0.64044)Sepal.Length - (0.2459)Sepal.Width + (0.72664)Petal.Width - 1.08778

Versicolor Species

Petal.Length = (0.64044)Sepal.Length - (0.2459)Sepal.Width + (0.72664)Petal.Width - 1.08778 + 1.25859

Virginica Species

Petal.Length = (0.64044)Sepal.Length - (0.2459)Sepal.Width + (0.72664)Petal.Width - 1.08778 + 1.69416

Notice that the Setosa species has the lowest intercept value and the Virginica species possesses the highest intercept as shown previously. In addition, we see that our root-mean squared deviation value is 0.309, and the mean squared error is 0.0953. Therefore, our model is on average off by 0.308 units of Petal Length - pretty accurate!