Iris Species

Iris Flower Species: Setosa, Versicolor or Virginica?

In this project, I work with the 'Iris' dataset, available from UCI Machine Learning repository. This dataset contains information about properties that differentiate species of Iris flower. There is information about length and width of its Petals and Sepals.

Let's see what each property of Iris flower is. Petals are modified leaves that surround the reproductive parts of flowers. Sepals are the usually-green part that function as protection for the flower in bud. You can read more about Petals and Sepals from the linked pages. It is always good to know more about your dataset as you progress with making prediction on it.

Understand the Dataset

```
library(ggplot2)
library(corrplot)
library(gridExtra)
library(class)
library(gmodels)
```

```
# Load the data
iris <- read.csv(url("http://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"), header

# Add column names
names(iris) <- c("Sepal.Length", "Sepal.Width", "Petal.Length", "Petal.Width", "Species")

# Print first lines
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width     Species
## 1          5.1         3.5          1.4         0.2 Iris-setosa
## 2          4.9         3.0          1.4         0.2 Iris-setosa
## 3          4.7         3.2          1.3         0.2 Iris-setosa
## 4          4.6         3.1          1.5         0.2 Iris-setosa
## 5          5.0         3.6          1.4         0.2 Iris-setosa
## 6          5.4         3.9          1.7         0.4 Iris-setosa
```

Let's see how the structure of our dataset is:

```
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 "Iris-setosa",..: 1 1 1 1 1 1 1 1 1 1 ...
```

All properties regarding Petal and Sepal are numeric, and the Species has remained a factor.

Moving on, I'd like to get a summary on the database to see how far values are from each other. As part of Summary it gives me the minimum, mean, median, and maximum numbers in each variable, making it easier to see how far value are.

```
summary(iris)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300
##  Mean   :5.843   Mean   :3.054   Mean   :3.759   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
##             Species
##  Iris-setosa    :50
##  Iris-versicolor:50
##  Iris-virginica :50
##
##
##
```

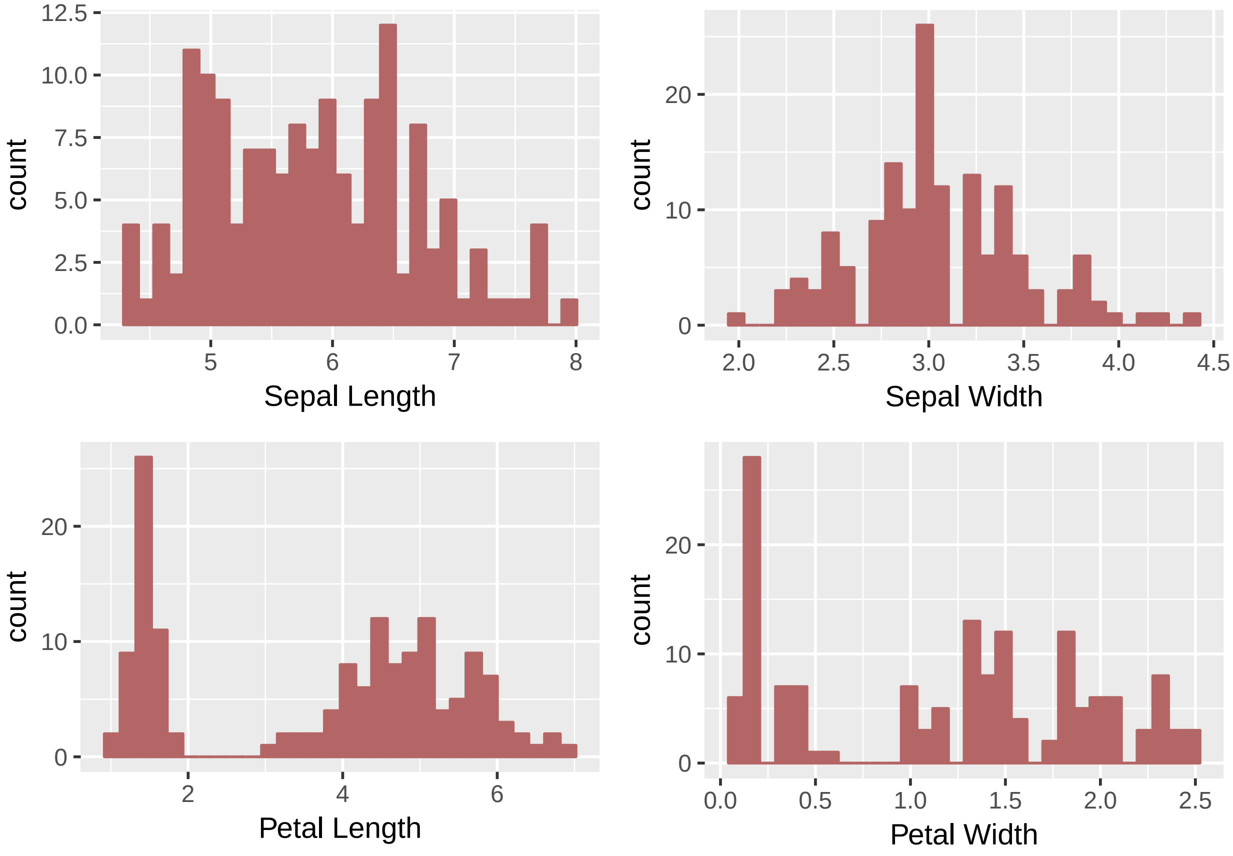
Sepal length ranges from 4.3-7.9. Its width ranges from 2.0-4.0. Petal length ranges from 1.0-6.9 and its width ranges from 0.1-2.5. The number seem to be in a quite nice range from each other. At this point I do not see the need for normalization, but later in my analysis, I might normalize these number to make more accurate predictions.

Exploratory Analysis

Before starting to create a prediction model, I need to get to know my data and how it's distributed. I first start with simple histograms that can show the distribution of each variable (except Species) in the dataset. Based on the histograms, I will see if I need to tone my values or not. Later I will continue to draw visualizations for Sepals and Petals lengths and widths to see how they are correlated with each other. I make these visualizations for understandable by adding the type of species. I finish my visualization process by a corrplot that can show correlation between each pair of variables.

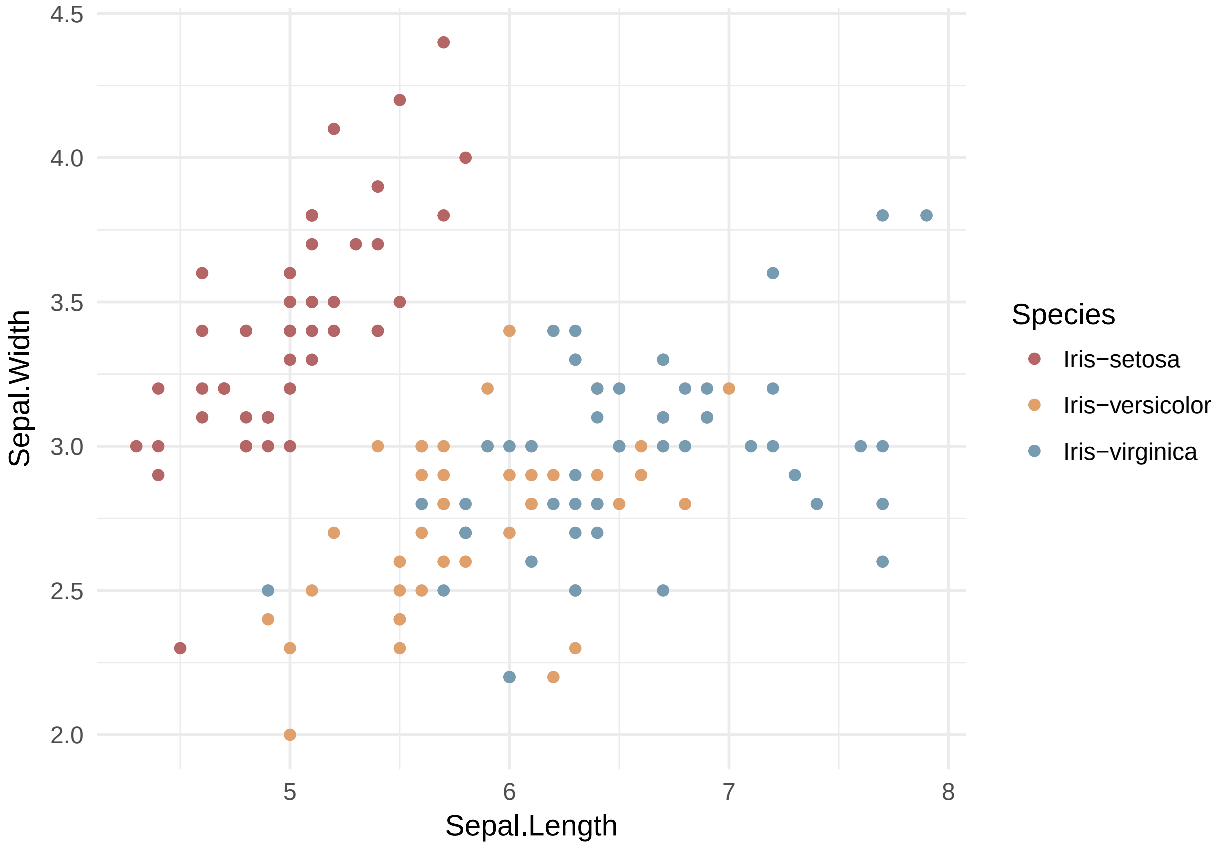
```
#create a function to get a histogram of all the variables in the dataset
get_histogram <- function(var, xlabel) {
  return (qplot(x = var, data = iris, xlab = xlabel,
                color= I('#b46565'), fill = I('#b46565')))
}

#png(height=1300, width=1500, res=300,
#    filename='iris.png')
grid.arrange(get_histogram(iris$Sepal.Length, 'Sepal Length'),
get_histogram(iris$Sepal.Width, 'Sepal Width'),
get_histogram(iris$Petal.Length, 'Petal Length'),
get_histogram(iris$Petal.Width, 'Petal Width'),
ncol=2)
```



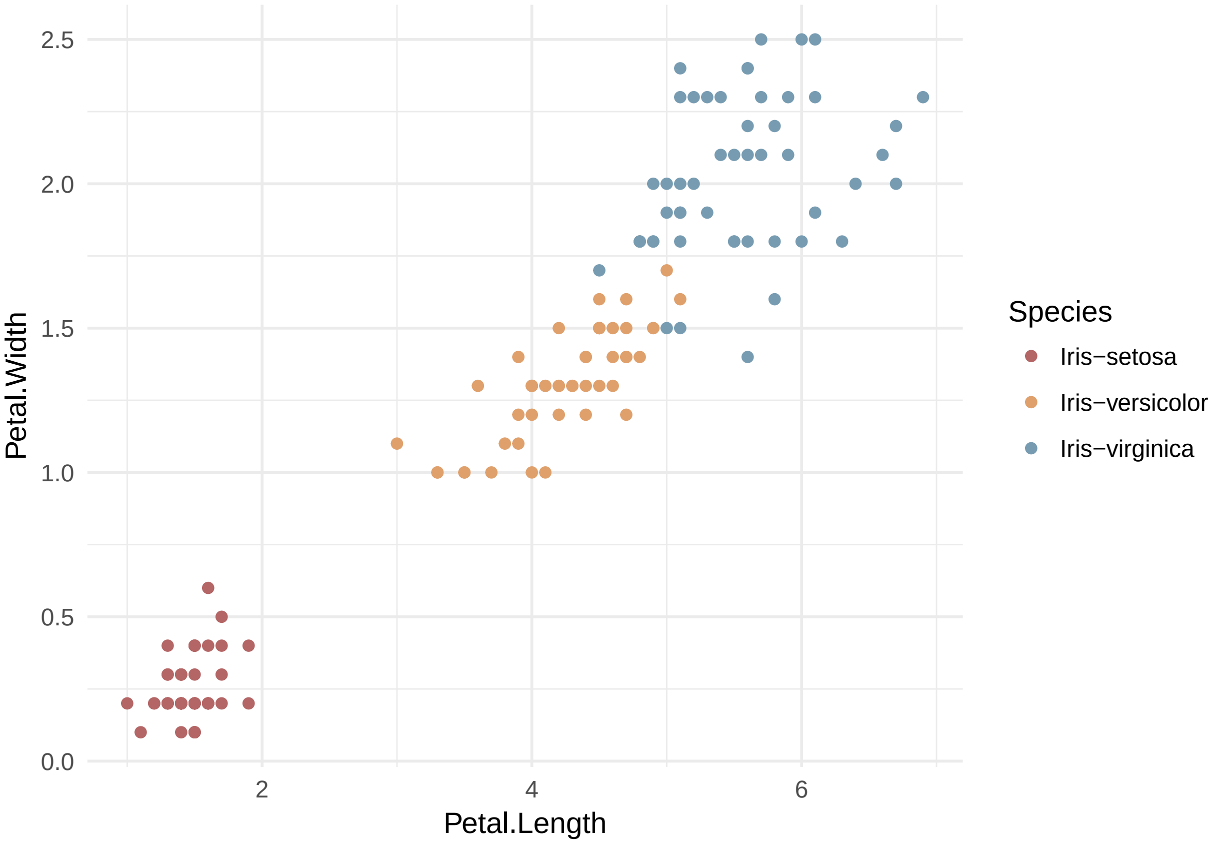
These histograms show that Sepal values looks quite normally distributed. With Petal, however, for its length the data is a bit separated- there are some observations with 0-2cm in length, and the rest are distributed between 3-7. There seems to be no observation with a length between 2-3. For its width, the distribution is not normal either. There is a high peak of petals with a width between 0.0-0.25.

```
ggplot(data=iris, aes(x=Sepal.Length, y=Sepal.Width, col=Species)) +
  geom_point() +
  theme_minimal() +
  scale_color_manual(values = c("#b46565","#dfa06c","#779cb2"))
```



Sepal length and width seems to be positively correlated. For Setosa the correlation is stronger than Virginica and Versicolor. As you see for Versicolor and Virginica the points on the scatter plots are more seperated than Setosa.

```r
ggplot(data=iris, aes(x=Petal.Length, y=Petal.Width, col=Species)) +
  geom_point() +
  theme_minimal() +
  scale_color_manual(values = c("#b46565","#dfa06c","#779cb2"))
```



There is also a positive correlation between length and width of petal. The correlation seems to be strong for all flower species. After doing the visualization, let's also print out some number to make sure about these correlations.

```
# Return values of levels of species (Setosa,Versicolor,Virginica)
x=levels(iris$Species)

# Print Setosa correlation matrix
print(x[1])
```

```
## [1] "Iris-setosa"
```

```
cor(iris[iris$Species==x[1],1:4])
```

```
##              Sepal.Length Sepal.Width Petal.Length Petal.Width
## Sepal.Length    1.0000000   0.7467804    0.2638741   0.2790916
## Sepal.Width     0.7467804   1.0000000    0.1766946   0.2799729
## Petal.Length    0.2638741   0.1766946    1.0000000   0.3063082
## Petal.Width     0.2790916   0.2799729    0.3063082   1.0000000
```

```
# Print Versicolor correlation matrix
print(x[2])
```

```
## [1] "Iris-versicolor"
```

```
cor(iris[iris$Species==x[2],1:4])
```

```
##              Sepal.Length Sepal.Width Petal.Length Petal.Width
## Sepal.Length    1.0000000   0.5259107    0.7540490   0.5464611
## Sepal.Width     0.5259107   1.0000000    0.5605221   0.6639987
## Petal.Length    0.7540490   0.5605221    1.0000000   0.7866681
## Petal.Width     0.5464611   0.6639987    0.7866681   1.0000000
```

```r
# Print Virginica correlation matrix
print(x[3])
```

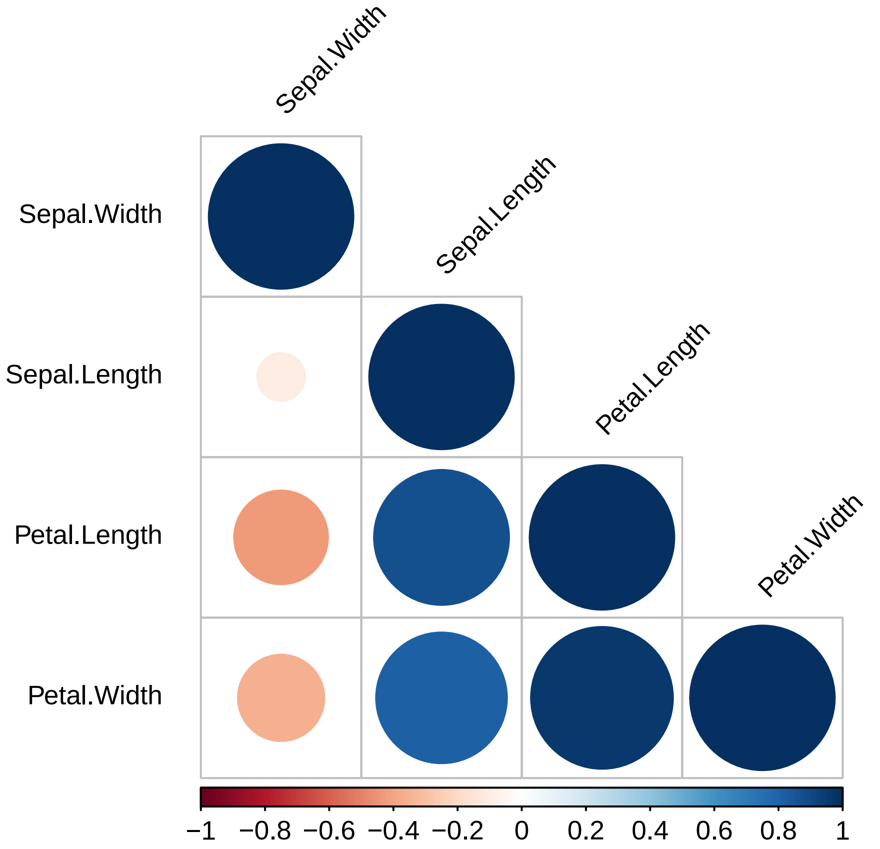
```
## [1] "Iris-virginica"
```

```r
cor(iris[iris$Species==x[3],1:4])
```

```
##              Sepal.Length Sepal.Width Petal.Length Petal.Width
## Sepal.Length    1.0000000   0.4572278    0.8642247   0.2811077
## Sepal.Width     0.4572278   1.0000000    0.4010446   0.5377280
## Petal.Length    0.8642247   0.4010446    1.0000000   0.3221082
## Petal.Width     0.2811077   0.5377280    0.3221082   1.0000000
```

```r
M <- cor(iris[c(1:4)])

#using corplot to create a correlation plot from all the variables
corrplot(M, method = 'circle', type="lower", order ="hclust",
         tl.col="black", tl.cex = 0.8, tl.offset = 1, tl.srt = 45)
```



The correlations shown both with numbers and a corrplot confirm the conclusions above:

- Petal width and length has an overall positive correlation with each other. The correlation is quite strong (the circle blue color is towards the darker shade)
- Sepal width and length seem to have a negative correlation with each other, and it is not strong either
- Petal length and width seem to have a quite strong negative correlation with sepal width
- Petal length and width seem to have a strong positive correlation with sepal length

Prediction Model

After understanding the dataset and getting some proper visualization and correlations, it's time to put it into performing the predictions.

However, before that I like to form my data into a more normalized shape. I decided to scale all my values between [0,1]. I do this because I do not like any of the variables(aka. features) to have an unwanted weight on the way my prediction model work. Although the range of the values do not differ by a high degree, my personal preference was to normalize my data.

Normalization

To perform this process, I write a function that takes a value, substract it from the minimum value and divide it by the difference of minimum and maximum values.

- value - min(value) / max(value) - min(value)

This puts all my values in the range [0,1], hence, will normalize my distribution.

```
# Create normalization function
normalize <- function(x) {
    num <- x - min(x)
    denom <- max(x) - min(x)
    return (num/denom)
}

# Normalize the iris data (all values will be between [0,1] after this operation)
# Exclude 'Species' as it is a factor value and cannot be included in this operation
iris_norm <- as.data.frame(lapply(iris[1:4], normalize))

# Add the 'Species' column back to the normalized dataframe
iris_norm$Species <- iris$Species

# Print the first 6 observations
head(iris_norm)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width     Species
## 1   0.22222222   0.6250000   0.06779661  0.04166667 Iris-setosa
## 2   0.16666667   0.4166667   0.06779661  0.04166667 Iris-setosa
## 3   0.11111111   0.5000000   0.05084746  0.04166667 Iris-setosa
## 4   0.08333333   0.4583333   0.08474576  0.04166667 Iris-setosa
## 5   0.19444444   0.6666667   0.06779661  0.04166667 Iris-setosa
## 6   0.30555556   0.7916667   0.11864407  0.12500000 Iris-setosa
```

As you see, all the values are now between [0,1]. If I get a summary from the normalized dataframe, you will see that the range of minimum and maximum values are now 0-1 which is much smaller.

```
summary(iris_norm)
```

```
##   Sepal.Length     Sepal.Width      Petal.Length     Petal.Width
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.00000
##  1st Qu.:0.2222   1st Qu.:0.3333   1st Qu.:0.1017   1st Qu.:0.08333
##  Median :0.4167   Median :0.4167   Median :0.5678   Median :0.50000
##  Mean   :0.4287   Mean   :0.4392   Mean   :0.4676   Mean   :0.45778
##  3rd Qu.:0.5833   3rd Qu.:0.5417   3rd Qu.:0.6949   3rd Qu.:0.70833
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.00000
##            Species
##  Iris-setosa    :50
##  Iris-versicolor:50
##  Iris-virginica :50
##
##
##
```

Training And Test Sets

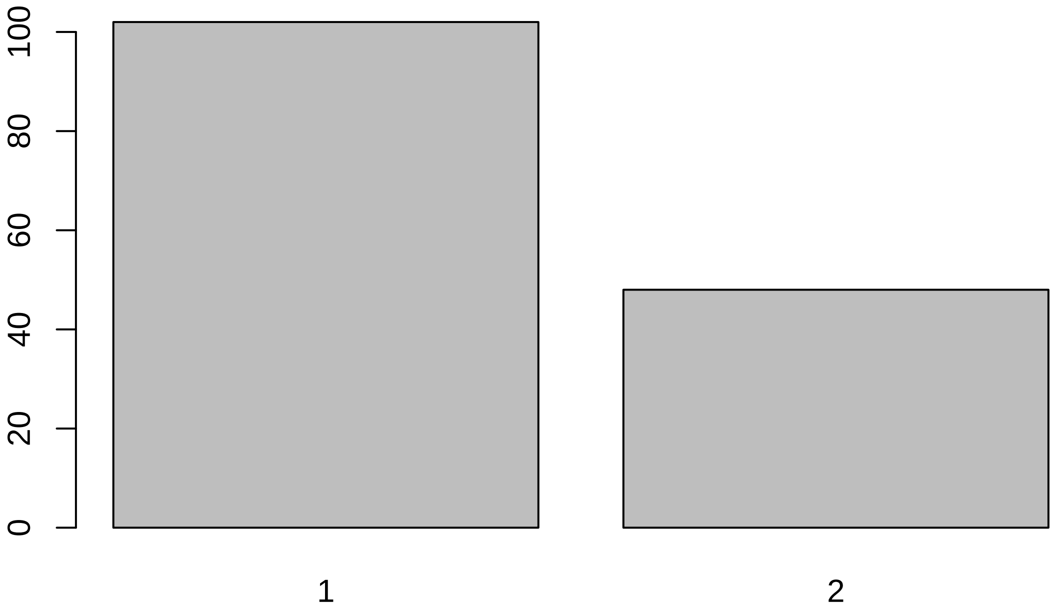
Now that my distribution is in a satisfactory level for me, I start by dividing it up to a training and testing dataset. I use 2/3 of my data as a training set and 1/3 as a testing set.

I use the sample function to take a sample with a size that is set as the number of rows of the Iris data set (i.e. 150). I use sample with replacement, meaning that I choose from a vector of 2 elements and assign either 1 or 2 to the 150 rows of the Iris data set. The assignment of the elements is done based on the probability weights of 0.67 and 0.33.

```
# Random number generator
set.seed(1234)

# Take a sample with a size that is set as the number of rows of the Iris data
ind <- sample(2, size=nrow(iris), replace=TRUE, prob=c(0.67, 0.33))

# Show the sample in a bar plot
barplot(table(sample(2, size=nrow(iris), replace=TRUE, prob=c(0.67, 0.33))))
```



```
# Create training set
iris.training <- iris[ind==1, 1:4]

# Create test set
iris.test <- iris[ind==2, 1:4]

# Create training labels
iris.trainLabels <- iris[ind==1,5]
```

```
# Create test labels
iris.testLabels <- iris[ind==2, 5]
```

Build the Model

For the prediction, I decided to use the K-nearest neighbour model. It's a very simple algorithm that use the contribution of the neighbours, so that the nearer ones contribute more to the average than the more distant ones. We set the k parameter ourselves. I give the knn function a training set, a testing set and my training labels. Note that you do not want to give your testing labels as these are the labels you would like the algorithm to predict for you. After running the algorithm, I store the results and create a new dataframe with my testing labels and the predicted labels to confirm how accurately my algorithm predicted the labels for me. I name this dataframe test.pred. After you print the merged dataframe, you see that all predictions are correct except for one versicolor label that the algorithm predicted as virginica.

```
# Build the model
iris_pred <- knn(train = iris.training, test = iris.test, cl = iris.trainLabels, k=3)

# Create a dataframe out of testing labels
irisTestLabels <- data.frame(iris.testLabels)

# Merge the testing label and prediction dataframes
test.pred <- data.frame(iris_pred, iris.testLabels)

# Rename the columns
names(test.pred) <- c("Predicted Species", "Observed Species")

# Print the merged dataframe
# test.pred
```

There is also another way to observe your results, using CrossTable function in R. With this table you can see how your testing labels and predictive algorithm have worked.

In our dataset as you see all predictions were correct except for versicolor which was predicted as virginica. You can find this by looking at '1' between Iris-virginica and Iris-versicolor.

```
CrossTable(x = iris.testLabels, y = iris_pred, prop.chisq=FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  40
##
##
##                 | iris_pred
## iris.testLabels |     Iris-setosa | Iris-versicolor |  Iris-virginica |       Row Total |
## ----------------|-----------------|-----------------|-----------------|-----------------|
##     Iris-setosa |              12 |               0 |               0 |              12 |
##                 |           1.000 |           0.000 |           0.000 |           0.300 |
##                 |           1.000 |           0.000 |           0.000 |                 |
##                 |           0.300 |           0.000 |           0.000 |                 |
## ----------------|-----------------|-----------------|-----------------|-----------------|
## Iris-versicolor |               0 |              12 |               0 |              12 |
##                 |           0.000 |           1.000 |           0.000 |           0.300 |
##                 |           0.000 |           0.923 |           0.000 |                 |
##                 |           0.000 |           0.300 |           0.000 |                 |
## ----------------|-----------------|-----------------|-----------------|-----------------|
##  Iris-virginica |               0 |               1 |              15 |              16 |
##                 |           0.000 |           0.062 |           0.938 |           0.400 |
##                 |           0.000 |           0.077 |           1.000 |                 |
##                 |           0.000 |           0.025 |           0.375 |                 |
## ----------------|-----------------|-----------------|-----------------|-----------------|
##    Column Total |              12 |              13 |              15 |              40 |
##                 |           0.300 |           0.325 |           0.375 |                 |
## ----------------|-----------------|-----------------|-----------------|-----------------|
##
##
```

Our algorithm did a pretty good job with predicting the labels based on petal and sepal lengths and widths, with only 1 mistake in the prediction.