Classification of iris flowers

The aim is to classify iris flowers among three species (Setosa, Versicolor, or Virginica) from sepals' and petals' length and width measurements.
The iris data set contains fifty instances of each of the three species.
The central goal is to design a model that makes proper classifications for new flowers.

1. Application type

This is a classification project. Indeed, the variable to be predicted is categorical (setosa, versicolor, or virginica).
The goal is to model class membership probabilities conditioned on the flower features.

2. Data set

The first step is to prepare the data set. This is the source of information for the classification problem.
For that, we need to configure the following concepts:

- Data source.
- Variables.
- Instances.

The data source is the file iris.csv. It contains the data for this example in comma-separated values (CSV) format. The number of columns is 5, and the number of rows is 150.
The variables are:

- **SepalLengthCm**: Sepal length, in centimeters, used as input.
- **SepalWidthCm**: Sepal width, in centimeters, used as input.
- **PetalLengthCm**: Petal length, in centimeters, used as input.
- **PetalWidthCm**: Petal width, in centimeters, used as input.
- **Species**: Iris-Setosa, Iris-Versicolor, or Iris-Virginica, used as the target.

```
#loading required packages
library(tidyverse) # visualization/processing
library(lattice)# visualization
library(ggpubr) # for multiple plots
library(GGally) # for pairplots
library(caret)  # machine learning models
library(ggplot2)
library(e1071)
library(dplyr)
library(randomForest)

#Importing the data
dataset=read.csv("C:/Users/Lancy/Desktop/iris.csv")

# View the top rows of the data
head(dataset)
##   Id SepalLengthCm SepalWidthCm PetalLengthCm PetalWidthCm     Species
##    1  1           5.1          3.5           1.4          0.2 Iris-setosa
##   2  2            4.9          3.0           1.4          0.2 Iris-setosa
##   3  3            4.7          3.2           1.3          0.2 Iris-setosa

# Dimensions of the data
dim(dataset)
## [1] 150   6

# Column names of the data
names(dataset)
## [1] "Id"          "SepalLengthCm" "SepalWidthCm"  "PetalLengthCm"
```

```
## [5] "PetalWidthCm"  "Species"

# Structure of the data
str(dataset)
## 'data.frame':    150 obs. of  6 variables:
##  $ Id           : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ SepalLengthCm: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ SepalWidthCm : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ PetalLengthCm: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ PetalWidthCm : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species      : chr  "Iris-setosa" "Iris-setosa" "Iris-setosa" "Iris-setosa" ...


# Unique values per column
lapply(dataset, function(x) length(unique(x)))
## $Id
## [1] 150
##
## $SepalLengthCm
## [1] 35
##
## $SepalWidthCm
## [1] 23
##
## $PetalLengthCm
## [1] 43
##
## $PetalWidthCm
## [1] 22
##
## $Species
## [1] 3

#summary of the data
summary(dataset)
##        Id         SepalLengthCm    SepalWidthCm    PetalLengthCm
##  Min.   :  1.00   Min.   :4.300   Min.   :2.000   Min.   :1.000
##  1st Qu.: 38.25   1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600
##  Median : 75.50   Median :5.800   Median :3.000   Median :4.350
##  Mean   : 75.50   Mean   :5.843   Mean   :3.054   Mean   :3.759
##  3rd Qu.:112.75   3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100
##  Max.   :150.00   Max.   :7.900   Max.   :4.400   Max.   :6.900
##   PetalWidthCm     Species
##  Min.   :0.100   Length:150
##  1st Qu.:0.300   Class :character
##  Median :1.300   Mode  :character
##  Mean   :1.199
##  3rd Qu.:1.800
##  Max.   :2.500
```

Observation:
Checking the scales of features is very important.

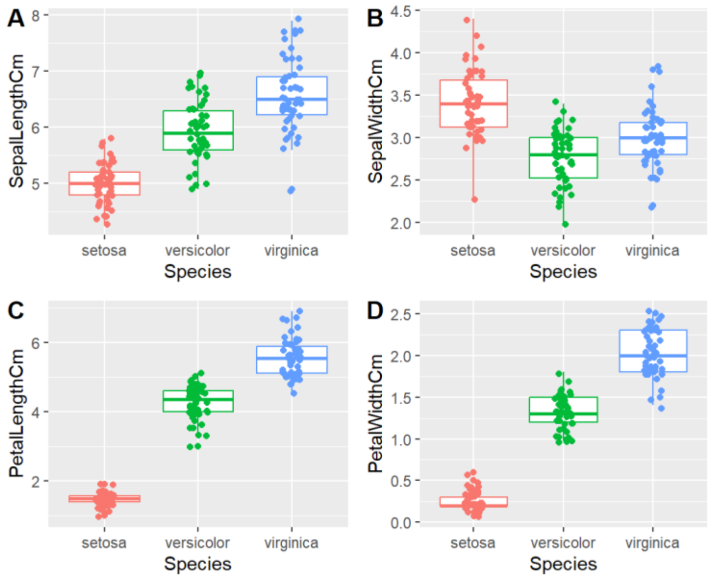
- Sepal length ranges from 4.3-7.9,
- Sepal width range: 2-4.4,
- Petal length range:1-6.9,

- Petal width:0.1-2.5.

The ranges basically are from 0 to 10, so we don't have to do scaling before the building the models.

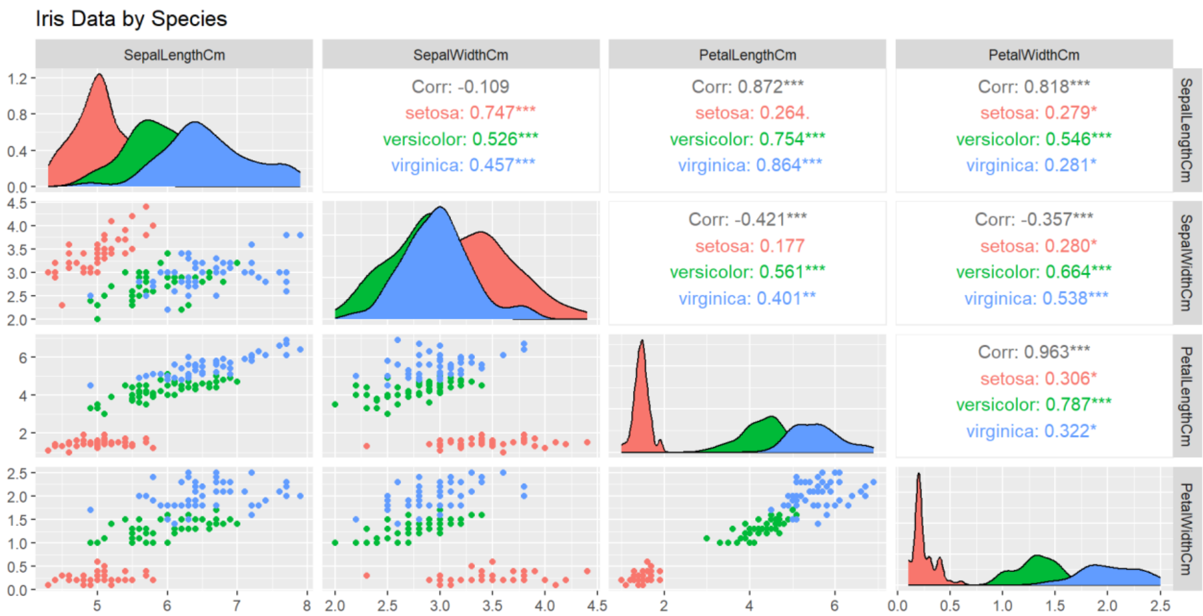
```
#Checking for missing values
sum(is.na(dataset))
## [1] 0

#remove Id for easy processing data
data=dataset[,-1]
head(data)
##   SepalLengthCm SepalWidthCm PetalLengthCm PetalWidthCm     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

#Found Species as character
data$Species=sapply(strsplit(as.character(data$Species),'-'), "[", 2)
str(data)
## 'data.frame':    150 obs. of  5 variables:
##  $ SepalLengthCm: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ SepalWidthCm : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ PetalLengthCm: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ PetalWidthCm : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species      : chr  "setosa" "setosa" "setosa" "setosa" ...


#change Species as factor
data$Species=as.factor(data$Species)
str(data)
## 'data.frame':    150 obs. of  5 variables:
##  $ SepalLengthCm: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ SepalWidthCm : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ PetalLengthCm: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ PetalWidthCm : 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 ...


#using boxplots to understand the distribution of attributes for each Species
p1=ggplot(data, aes(x = Species, y = SepalLengthCm,colour=Species)) +
 geom_boxplot() +
 geom_jitter(shape=16, position=position_jitter(0.1))+
 theme(legend.position="none")

p2=ggplot(data, aes(x = Species, y = SepalWidthCm,colour=Species)) +
 geom_boxplot() +
 geom_jitter(shape=16, position=position_jitter(0.1))+
 theme(legend.position="none")

p3=ggplot(data, aes(x = Species, y = PetalLengthCm,colour=Species)) +
 geom_boxplot() +
 geom_jitter(shape=16, position=position_jitter(0.1))+
 theme(legend.position="none")
```

```
p4=ggplot(data, aes(x = Species, y = PetalWidthCm,colour=Species)) +
 geom_boxplot() +
 geom_jitter(shape=16, position=position_jitter(0.1))+
 theme(legend.position="none")

ggarrange(p1,p2,p3,p4,
     labels = c("A", "B", "C","D"),
     ncol = 2, nrow = 2)
```



From above boxplots, we can see that virginica has a bigger petal and bigger sepal length, however sentosa has a smaller petal, but bigger sepal length.

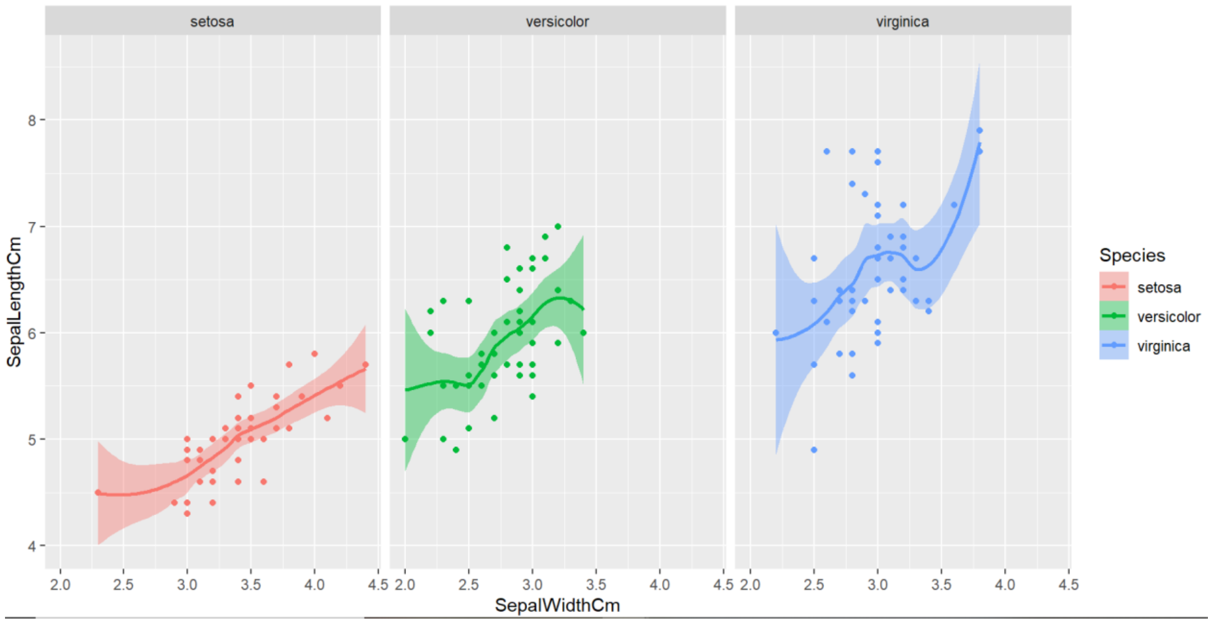
```
#Using Pairplots to understand relationships between attributes
ggpairs(data, columns=1:4, aes(color=Species)) +
 ggtitle("Iris Data by Species")
```



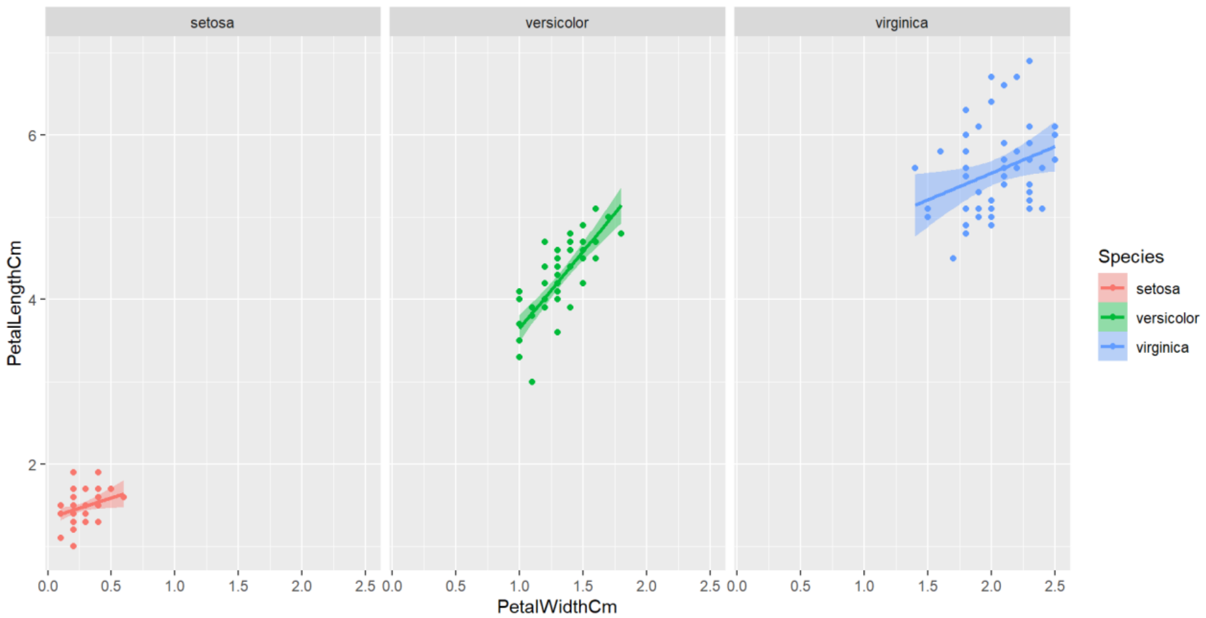
Petal width and Petal length have a strong linear correlation relationship.

```
#Using scatterplot to understand linear relationship
ggplot(data, aes(x =SepalWidthCm , y = SepalLengthCm  , color = Species))+
```

```
geom_point()+
geom_smooth(method="loess", aes(fill= Species, color = Species))+
facet_wrap(~Species, ncol = 3, nrow = 1)
```




```
ggplot(data, aes(x = PetalWidthCm , y =PetalLengthCm  , color = Species))+
 geom_point()+
 geom_smooth(method="lm", aes(fill= Species, color = Species))+
 facet_wrap(~Species, ncol = 3, nrow = 1)
```



Versicolor petal width and length has a strong linear relationship.

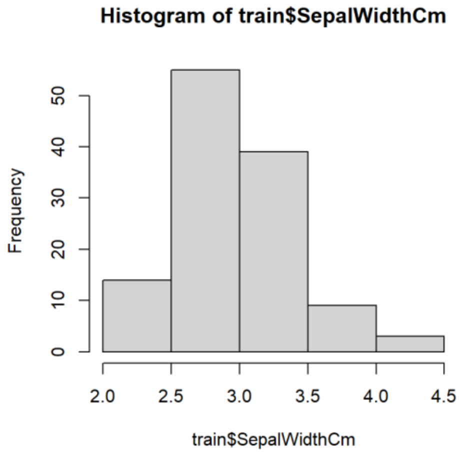
Splitting the data for training and testing

```
set.seed(101)
# We use the dataset to create a partition (80% training 20% testing)
id=createDataPartition(data$Species, p=0.80, list=FALSE)

# select 80% of data to train the models
train=data[id,]
```

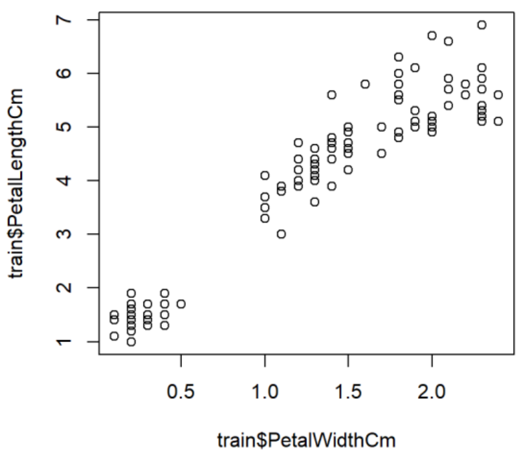
dim(train)
[1] 120   5

# select 20% of the data for testing
test=data[-id,]
dim(test)
[1] 30 5

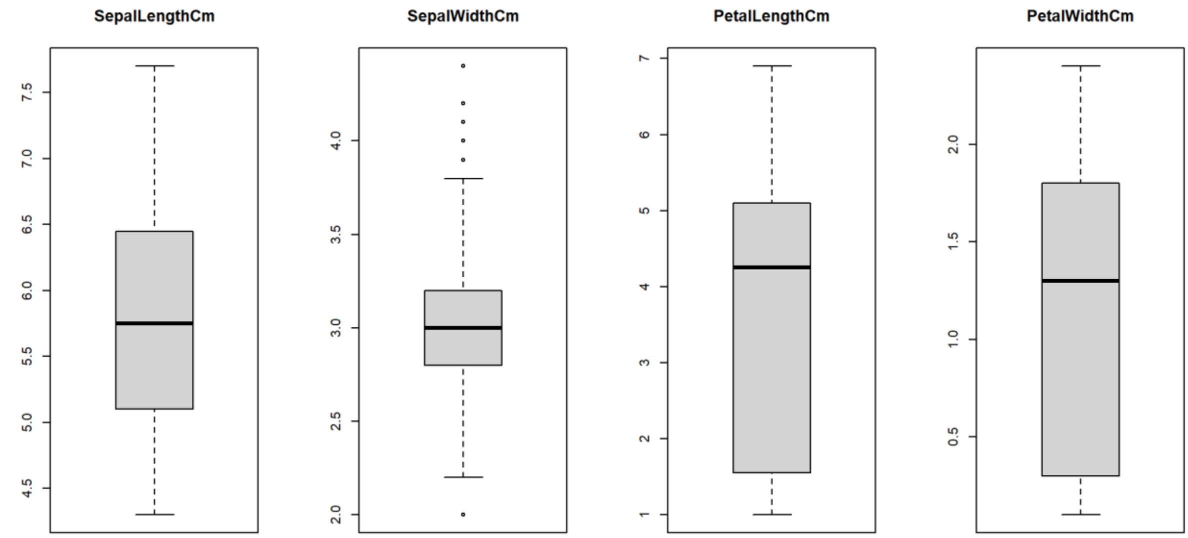
Histogram to understand the distribution and attributes
hist(train$SepalWidthCm)



Scatterplot to understand the distribution and attributes
plot(train$PetalLengthCm ~ train$PetalWidthCm, data=train)



Box plot to understand how the distribution varies by class of flower
par(mfrow=c(1,4))
for(i in 1:4) {
  boxplot(train[,i], main=names(train)[i])
}



#review the train dataset to confirm the Species are randomly selected
lapply(train, function(x) length(unique(x)))
$SepalLengthCm
[1] 34
##
$SepalWidthCm
[1] 23
##
$PetalLengthCm
[1] 42
##
$PetalWidthCm
[1] 20
##
$Species
[1] 3

table(train$Species)
```
##
##     setosa versicolor  virginica
##         40         40         40
```
summary(train)
```
##  SepalLengthCm    SepalWidthCm   PetalLengthCm    PetalWidthCm
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.575   1st Qu.:0.300
##  Median :5.750   Median :3.000   Median :4.250   Median :1.300
##  Mean   :5.836   Mean   :3.041   Mean   :3.734   Mean   :1.187
##  3rd Qu.:6.425   3rd Qu.:3.200   3rd Qu.:5.100   3rd Qu.:1.800
##  Max.   :7.700   Max.   :4.400   Max.   :6.900   Max.   :2.400
##        Species
##  setosa    :40
##  versicolor:40
##  virginica :40
##
##
##
```

str(train)

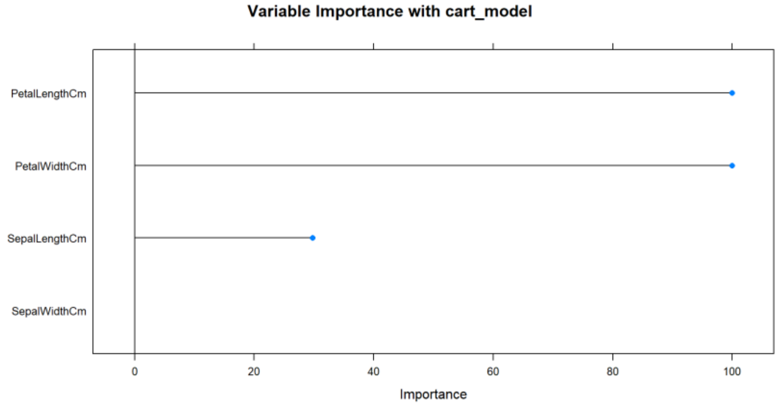
'data.frame':   120 obs. of 5 variables:
$ SepalLengthCm: num 4.7 5.4 4.6 5 4.4 4.9 5.4 4.8 4.8 4.3 ...
$ SepalWidthCm : num 3.2 3.9 3.4 3.4 2.9 3.1 3.7 3.4 3 3 ...
$ PetalLengthCm: num 1.3 1.7 1.4 1.5 1.4 1.5 1.5 1.6 1.4 1.1 ...
$ PetalWidthCm : num 0.2 0.4 0.3 0.2 0.2 0.1 0.2 0.2 0.1 0.1 ...
$ Species   : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
Observation- 1.The train dataset has 120 observations while test dataset has 30. 2.Each class has the same number of instances (40).

Model building
Model 1: Decision tree

```
set.seed(101)

cart_model <- train(train[,1:4], train[, 5], method='rpart2')
## note: only 2 possible values of the max tree depth from the initial fit.
##  Truncating the grid to 2 .

# Predict the labels of the test set
Predictions=predict(cart_model,test[,1:4])

# Evaluate the predictions
table(predictions)
## predictions
##     setosa versicolor  virginica
##         10          9         11

# Confusion matrix
confusionMatrix(predictions,test[,5])
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         10          0         0
##   versicolor      0          8         1
##   virginica       0          2         9
##
## Overall Statistics
##
##                Accuracy : 0.9
##                  95% CI : (0.7347, 0.9789)
##     No Information Rate : 0.3333
##     P-Value [Acc > NIR] : 1.665e-10
##
##                   Kappa : 0.85
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                 1.0000            0.8000           0.9000
## Specificity                 1.0000            0.9500           0.9000
## Pos Pred Value              1.0000            0.8889           0.8182
## Neg Pred Value              1.0000            0.9048           0.9474
## Prevalence                  0.3333            0.3333           0.3333
```

Detection Rate 0.3333 0.2667 0.3000
Detection Prevalence 0.3333 0.3000 0.3667
Balanced Accuracy 1.0000 0.8750 0.9000

#feature importance
importance_cart <- **varImp**(cart_model)
plot(importance_cart, main="Variable Importance with cart_model")



As suspected, Petal Width is the most used variable, followed by Petal Length and Sepal Length.

Model 2 KNN

 # Train the model with preprocessing
set.seed(101)
knn_model <- **train**(train[, 1:4], train[, 5], method='knn',
preProcess=**c**("center", "scale"))

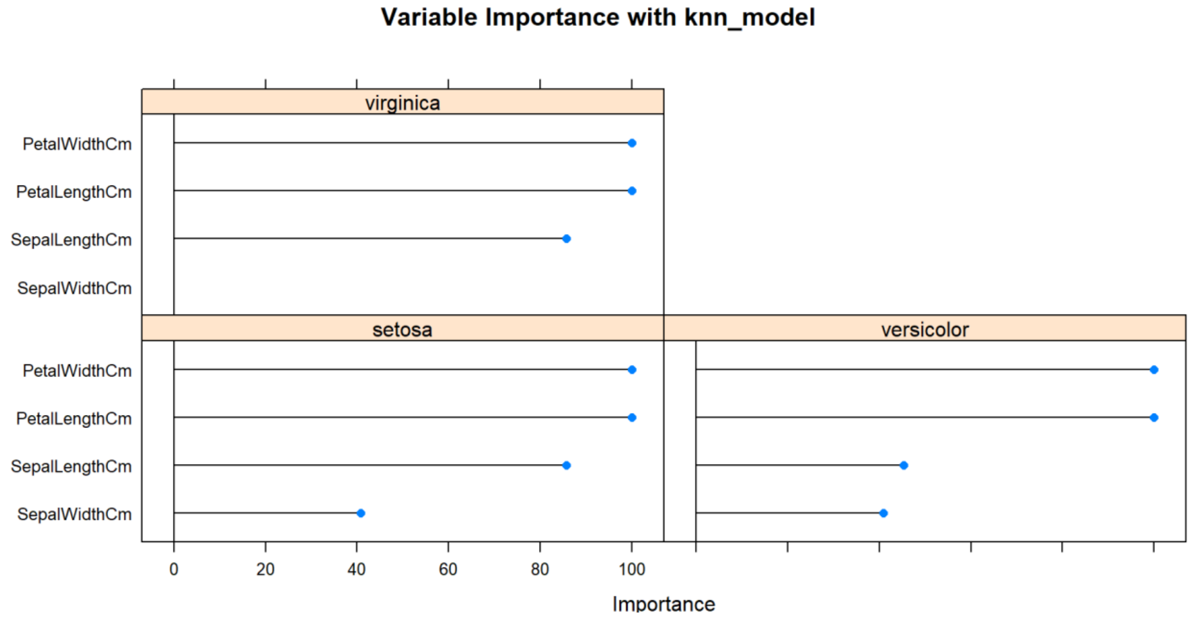
 # Predict values
predictions<-**predict**(knn_model,test[,1:4], type="raw")

 # Confusion matrix
confusionMatrix(predictions,test[,5])

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         10          0         0
##   versicolor      0          7         2
##   virginica       0          3         8
##
## Overall Statistics
##
##                Accuracy : 0.8333
##                  95% CI : (0.6528, 0.9436)
##     No Information Rate : 0.3333
##     P-Value [Acc > NIR] : 2.444e-08
##
##                   Kappa : 0.75
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                 1.0000            0.7000           0.8000
```

```
## Specificity            1.0000     0.9000     0.8500
## Pos Pred Value         1.0000     0.7778     0.7273
## Neg Pred Value         1.0000     0.8571     0.8947
## Prevalence             0.3333     0.3333     0.3333
## Detection Rate         0.3333     0.2333     0.2667
## Detection Prevalence   0.3333     0.3000     0.3667
## Balanced Accuracy      1.0000     0.8000     0.8250
```

```
#feature importance
importance_knn <- varImp(knn_model)
plot(importance_knn, main="Variable Importance with knn_model")
```



Model 3 Neural Network

```
# Train the model with preprocessing
set.seed(101)
nnet_model <- train(train[, 1:4], train[, 5], method='nnet',
            preProcess=c("center", "scale"),
            tuneLength = 2,
            trace = FALSE,
            maxit = 100)

# Predict values
predictions<-predict(nnet_model,test[,1:4], type="raw")

# Confusion matrix
confusionMatrix(predictions,test[,5])
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         10          0         0
##   versicolor      0          7         1
##   virginica       0          3         9
##
## Overall Statistics
##
##                Accuracy : 0.8667
```

```
##                95% CI : (0.6928, 0.9624)
##     No Information Rate : 0.3333
##     P-Value [Acc > NIR] : 2.296e-09
##
##                  Kappa : 0.8
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity               1.0000            0.7000           0.9000
## Specificity               1.0000            0.9500           0.8500
## Pos Pred Value            1.0000            0.8750           0.7500
## Neg Pred Value            1.0000            0.8636           0.9444
## Prevalence                0.3333            0.3333           0.3333
## Detection Rate            0.3333            0.2333           0.3000
## Detection Prevalence      0.3333            0.2667           0.4000
## Balanced Accuracy         1.0000            0.8250           0.8750
```

```
#feature importance
importance_nnet <- varImp(nnet_model);importance_nnet
```

```
## nnet variable importance
##
##   variables are sorted by maximum importance across the classes
##               Overall setosa versicolor virginica
## PetalLengthCm  100.00 100.00     100.00    100.00
## PetalWidthCm    97.48  97.48      97.48     97.48
## SepalWidthCm    39.49  39.49      39.49     39.49
## SepalLengthCm    0.00   0.00       0.00      0.00
```

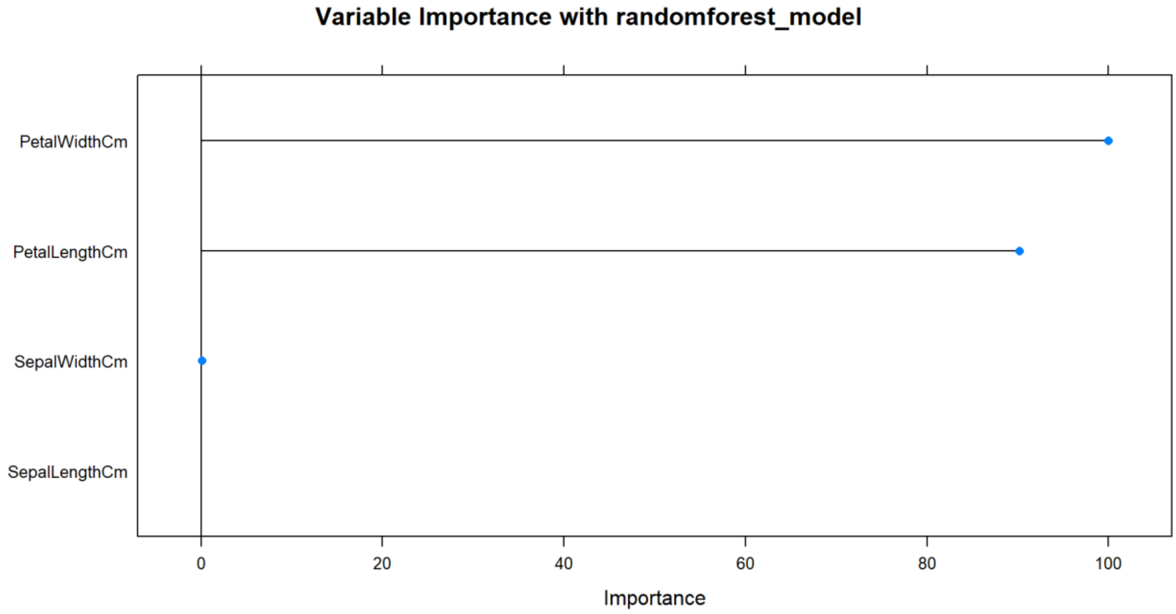
Model 4 Randomforest

```
# Train the model with preprocessing
set.seed(101)
randomforest_model <- train(train[, 1:4], train[, 5], method='rf')

# Predict values
predictions<-predict(randomforest_model,test[,1:4], type="raw")

# Confusion matrix
confusionMatrix(predictions,test[,5])
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         10          0         0
##   versicolor      0          9         0
##   virginica       0          1        10
##
## Overall Statistics
##
##                Accuracy : 0.9667
##                  95% CI : (0.8278, 0.9992)
##     No Information Rate : 0.3333
```

P-Value [Acc > NIR] : 2.963e-13
##
Kappa : 0.95
##
Mcnemar's Test P-Value : NA
##
Statistics by Class:
##

```
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                1.0000            0.9000           1.0000
## Specificity                1.0000            1.0000           0.9500
## Pos Pred Value             1.0000            1.0000           0.9091
## Neg Pred Value             1.0000            0.9524           1.0000
## Prevalence                 0.3333            0.3333           0.3333
## Detection Rate             0.3333            0.3000           0.3333
## Detection Prevalence       0.3333            0.3000           0.3667
## Balanced Accuracy          1.0000            0.9500           0.9750
```



Compare model performances
We have tried a few models on the Iris dataset which hopefully gives a broad overview of the variety of algorithms and models possible in R. As a final step we can summarize the results of our analysis by presenting the training set results for the models we employed

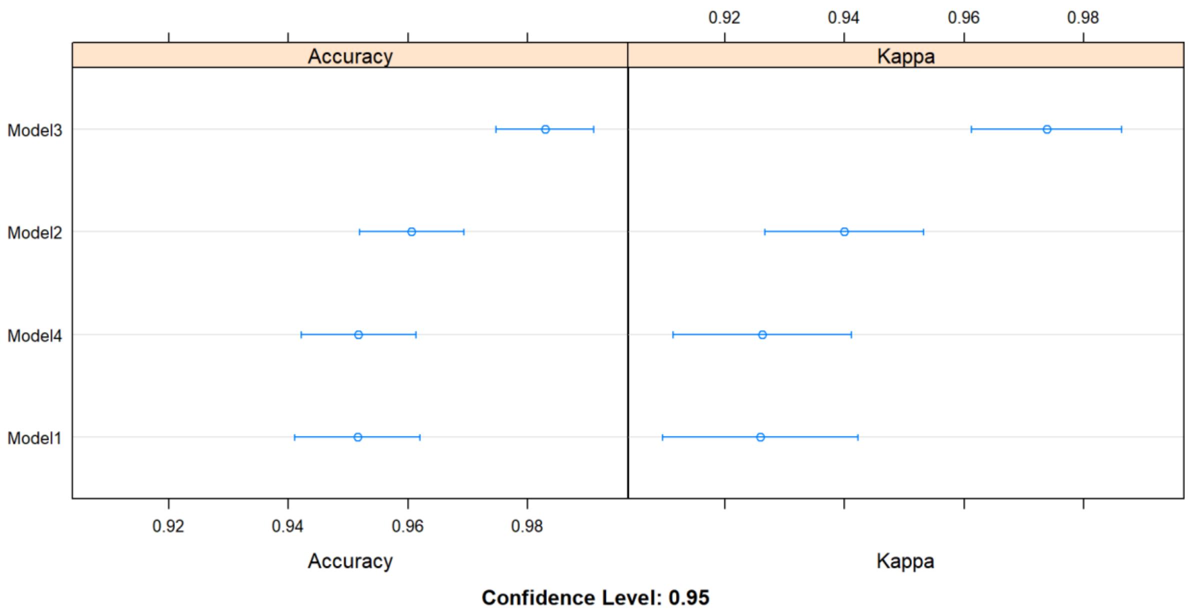
This sort of summary can be used to select the model just based on the training set data

```
models_compare <- resamples(list(cart_model,knn_model, nnet_model,randomforest_model))

# Summary of the models performances
summary(models_compare)
```

##
Call:
summary.resamples(object = models_compare)

```
##
## Models: Model1, Model2, Model3, Model4
## Number of resamples: 25
##
## Accuracy
##             Min.   1st Qu.    Median      Mean   3rd Qu. Max. NA's
## Model1 0.9000000 0.9361702 0.9545455 0.9515344 0.9729730 1.00    0
## Model2 0.9189189 0.9534884 0.9565217 0.9605841 0.9772727 1.00    0
## Model3 0.9347826 0.9772727 0.9800000 0.9828838 1.0000000 1.00    0
## Model4 0.9069767 0.9347826 0.9545455 0.9517384 0.9761905 0.98    0
##
## Kappa
##             Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## Model1 0.8434442 0.9036227 0.9312500 0.9259588 0.9578107 1.0000000    0
## Model2 0.8763920 0.9297386 0.9332366 0.9399519 0.9656250 1.0000000    0
## Model3 0.9001447 0.9652997 0.9698614 0.9738443 1.0000000 1.0000000    0
## Model4 0.8566667 0.9001447 0.9312500 0.9263153 0.9642553 0.9698614    0
```

```
# Dotplot of the models performances
dotplot(models_compare)
```



From the accuracy results, neural network model works best among the 4 models. Also, Petal Width and Petal Length are the key features for the classification.