Lab 2A: Classification (Easy)

1. Explore the Data

1. Set the working directory

```
setwd("C:/Workshop/Data")
```

2. Load Iris data

```
iris <- read.csv("Iris.csv")
```

3. Inspect the data

```
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
```

4. Load color brewer library

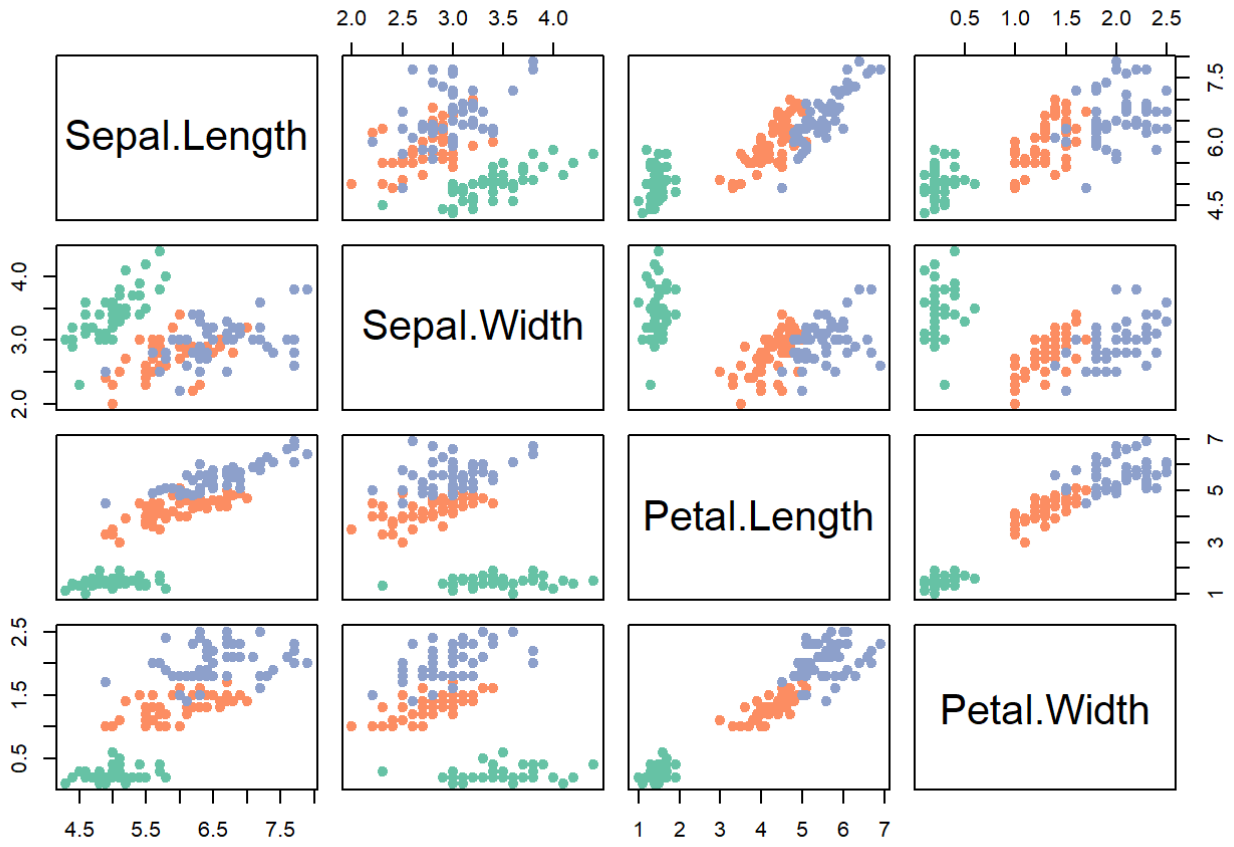
```
library(RColorBrewer)
```

5. Create a color palette

```
palette <- brewer.pal(3, "Set2")
```

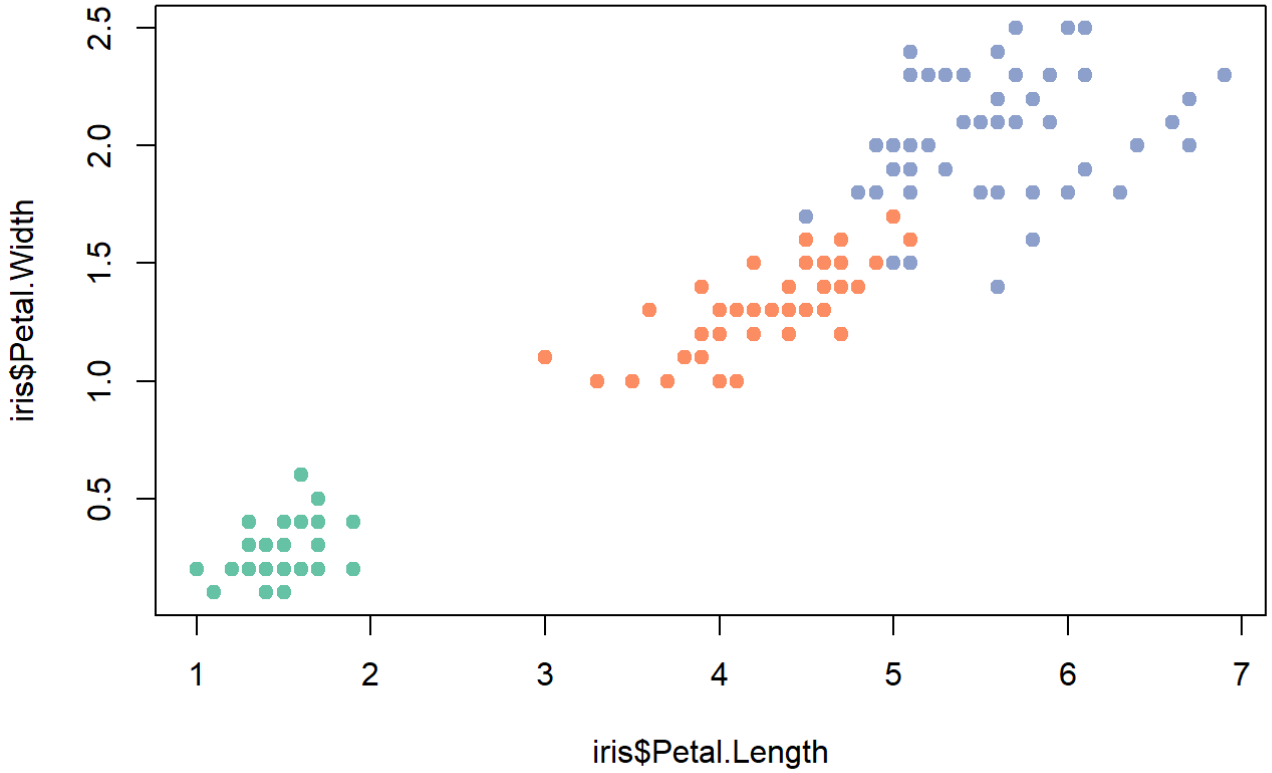
6. Create a scatterplot matrix colored by species

```
plot(
  x = iris[1:4],
  col = palette[as.numeric(iris$Species)],
  pch = 19)
```



7. View scatterplot of petal length vs width

```
plot(
  x = iris$Petal.Length,
  y = iris$Petal.Width,
  col = palette[as.numeric(iris$Species)],
  pch = 19)
```



2. Create Training and Test Sets

1. Set the random seed to make randomness reproducable

```
set.seed(42)
```

2. Randomly sample 100 of 150 row indexes

```
indexes <- sample(
  x = 1:150,
  size = 100)
```

3. Create training set from indexes

```
train <- iris[indexes, ]
```

4. Create test set from remaining indexes

```
test <- iris[-indexes, ]
```

3. Predict with K-Nearest Neighbors Classifier

1. Load the caret package

```
library(caret)
```

2. Train a knn model

```
knnModel <- knn3(
  formula = Species ~ .,
  data = train,
  k = 3)
```

3. Predict with model

```
knnPredictions <- predict(
  object = knnModel,
```

```
  newdata = test,
  type = "class")
```

4. Summarize prediction results

```
table(
  x = knnPredictions,
  y = test$Species)
```

```
##             y
## x            setosa versicolor virginica
##   setosa         17          0         0
##   versicolor      0         17         1
##   virginica       0          1        14
```

5. Create a confusion matrix

```
knnMatrix <- confusionMatrix(
  data = knnPredictions,
  reference = test$Species)
```

6. Inspect results

```
print(knnMatrix)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         17          0         0
##   versicolor      0         17         1
##   virginica       0          1        14
##
## Overall Statistics
##
```

```
##                Accuracy : 0.96
##                  95% CI : (0.8629, 0.9951)
##     No Information Rate : 0.36
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9398
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                   1.00            0.9444           0.9333
## Specificity                   1.00            0.9688           0.9714
## Pos Pred Value                1.00            0.9444           0.9333
## Neg Pred Value                1.00            0.9688           0.9714
## Prevalence                    0.34            0.3600           0.3000
## Detection Rate                0.34            0.3400           0.2800
## Detection Prevalence          0.34            0.3600           0.3000
## Balanced Accuracy             1.00            0.9566           0.9524
```

4. Predict with Decision Tree Classifier

1. Load decision tree package

```
library(tree)
```

2. Train tree model

```
treeModel <- tree(
  formula = Species ~ .,
  data = train)
```

3. Inspect the model

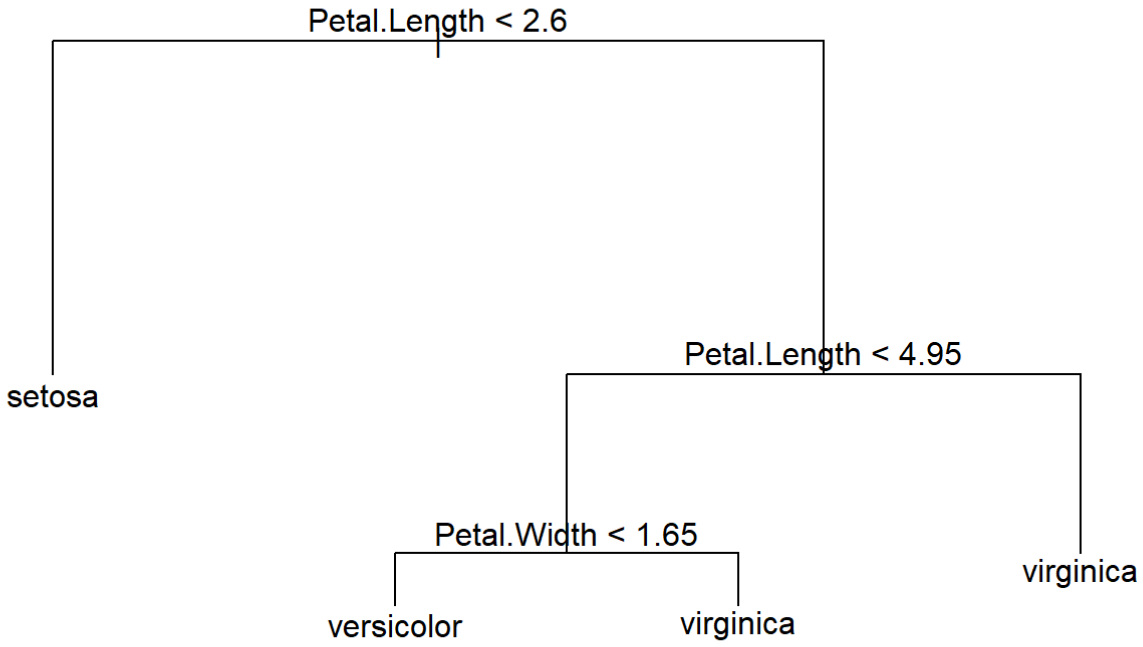
```
summary(treeModel)
```

```
##
## Classification tree:
## tree(formula = Species ~ ., data = train)
## Variables actually used in tree construction:
## [1] "Petal.Length" "Petal.Width"
## Number of terminal nodes:  4
## Residual mean deviance:  0.05213 = 5.004 / 96
## Misclassification error rate: 0.01 = 1 / 100
```

4. Plot the tree model

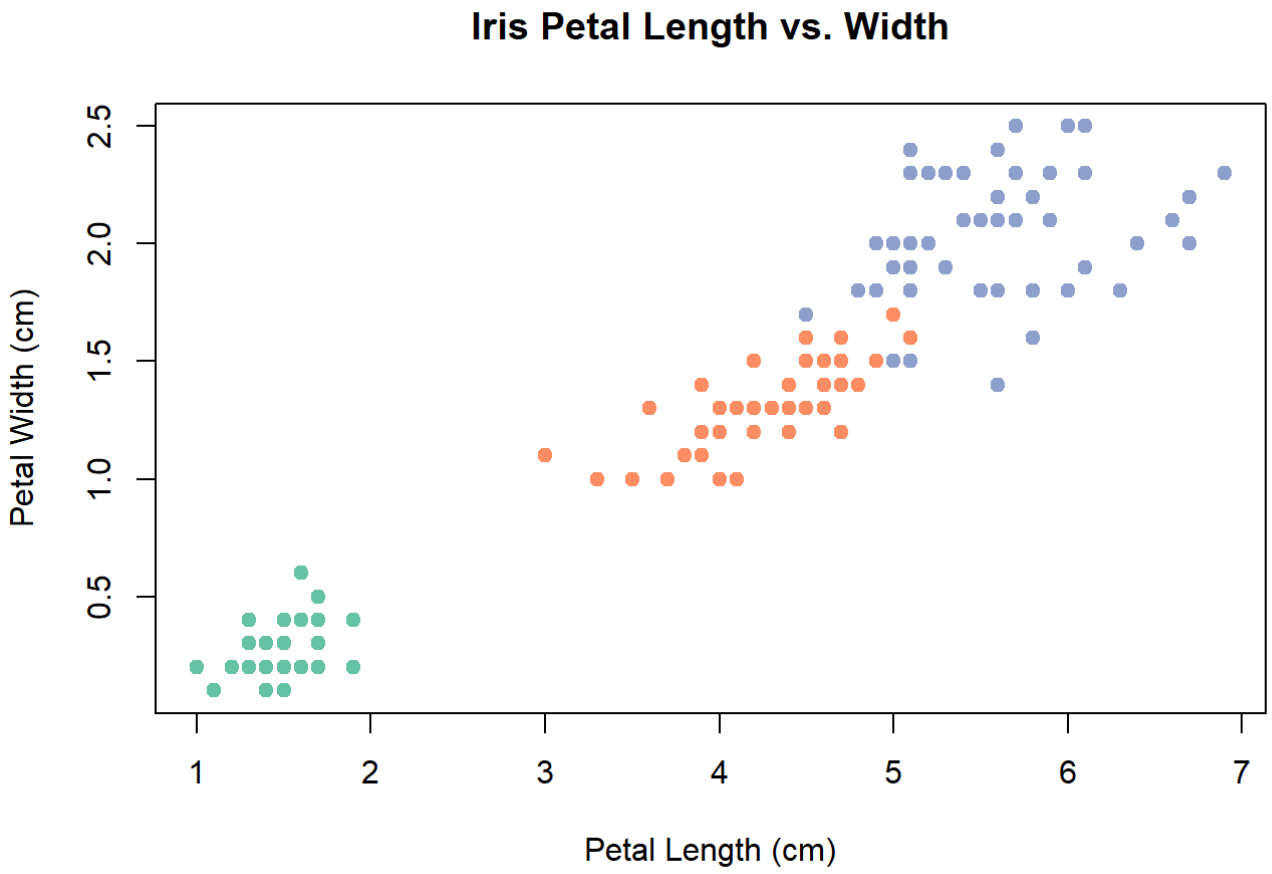
```
plot(treeModel)

text(treeModel)
```



5. Create a scatterplot colored by species

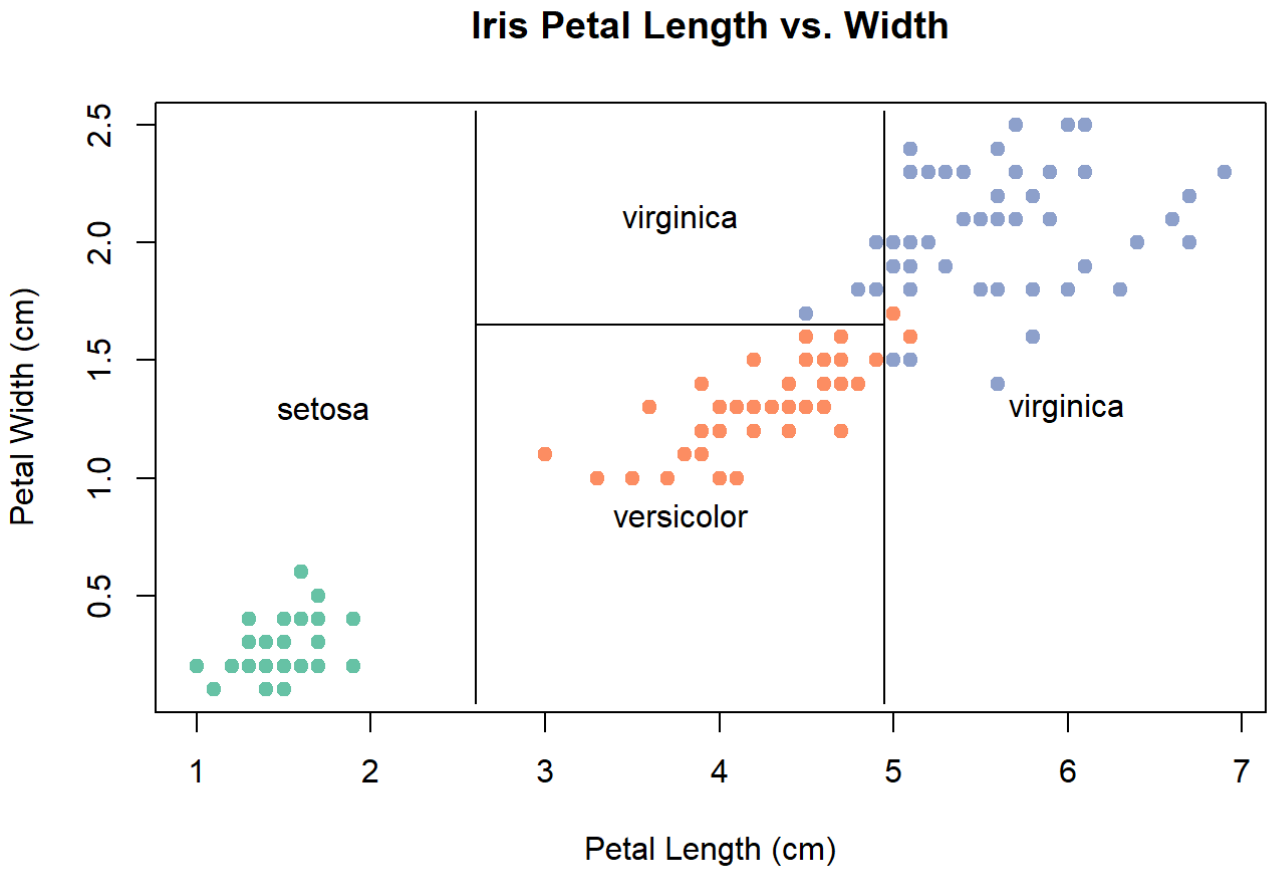
```
plot(
  x = iris$Petal.Length,
  y = iris$Petal.Width,
  pch = 19,
  col = palette[as.numeric(iris$Species)],
  main = "Iris Petal Length vs. Width",
  xlab = "Petal Length (cm)",
  ylab = "Petal Width (cm)")
```



6. Plot the decision boundaries

```
plot(
  x = iris$Petal.Length,
  y = iris$Petal.Width,
  pch = 19,
  col = palette[as.numeric(iris$Species)],
  main = "Iris Petal Length vs. Width",
  xlab = "Petal Length (cm)",
```

```
  ylab = "Petal Width (cm)")
partition.tree(
  tree = treeModel,
  label = "Species",
  add = TRUE)
```



7. Predict with model

```
treePredictions <- predict(
  object = treeModel,
```

```
  newdata = test,
  type = "class")
```

8. Create confusion matrix

```
treeMatrix <- confusionMatrix(
  data = treePredictions,
  reference = test$Species)
```

9. Inspect results

```
print(treeMatrix)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         17          0         0
##   versicolor      0         16         0
##   virginica       0          2        15
##
## Overall Statistics
##
##                Accuracy : 0.96
##                  95% CI : (0.8629, 0.9951)
##     No Information Rate : 0.36
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.94
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                  1.00            0.8889           1.0000
## Specificity                  1.00            1.0000           0.9429
```

```
## Pos Pred Value              1.00          1.0000          0.8824
## Neg Pred Value              1.00          0.9412          1.0000
## Prevalence                  0.34          0.3600          0.3000
## Detection Rate              0.34          0.3200          0.3000
## Detection Prevalence        0.34          0.3200          0.3400
## Balanced Accuracy           1.00          0.9444          0.9714
```

5. Predict with Neural Network Classifier

1. Load Neural Network package

```
library(nnet)
```

2. Train neural network model

```
neuralModel <- nnet(
  formula = Species ~ .,
  data = train,
  size = 4,
  decay = 0.0001,
  maxit = 500)
```

```
## # weights:  35
## initial  value 138.780405
## iter  10 value 50.853151
## iter  20 value 46.445736
## iter  30 value 46.436374
## iter  40 value 46.429588
## iter  50 value 46.416896
## iter  60 value 33.133720
## iter  70 value 4.154332
## iter  80 value 1.208767
## iter  90 value 0.926725
## iter 100 value 0.595340
## iter 110 value 0.577559
```

```
## iter 120 value 0.497661
## iter 130 value 0.424292
## iter 140 value 0.402469
## iter 150 value 0.397311
## iter 160 value 0.356799
## iter 170 value 0.345507
## iter 180 value 0.339297
## iter 190 value 0.336169
## iter 200 value 0.329560
## iter 210 value 0.327629
## iter 220 value 0.325959
## iter 230 value 0.324628
## iter 240 value 0.321737
## iter 250 value 0.319371
## iter 260 value 0.317631
## iter 270 value 0.316962
## iter 280 value 0.316919
## iter 290 value 0.316662
## iter 300 value 0.316482
## iter 310 value 0.316374
## iter 320 value 0.316254
## iter 330 value 0.316229
## iter 340 value 0.316217
## final  value 0.316209
## converged
```

3. Inspect the model

```
summary(neuralModel)
```

```
## a 4-4-3 network with 35 weights
## options were - softmax modelling  decay=1e-04
##  b->h1 i1->h1 i2->h1 i3->h1 i4->h1
##  -0.33  -0.82  -1.20   2.58   1.19
##  b->h2 i1->h2 i2->h2 i3->h2 i4->h2
##   0.08   0.47   0.14   0.54   0.18
```

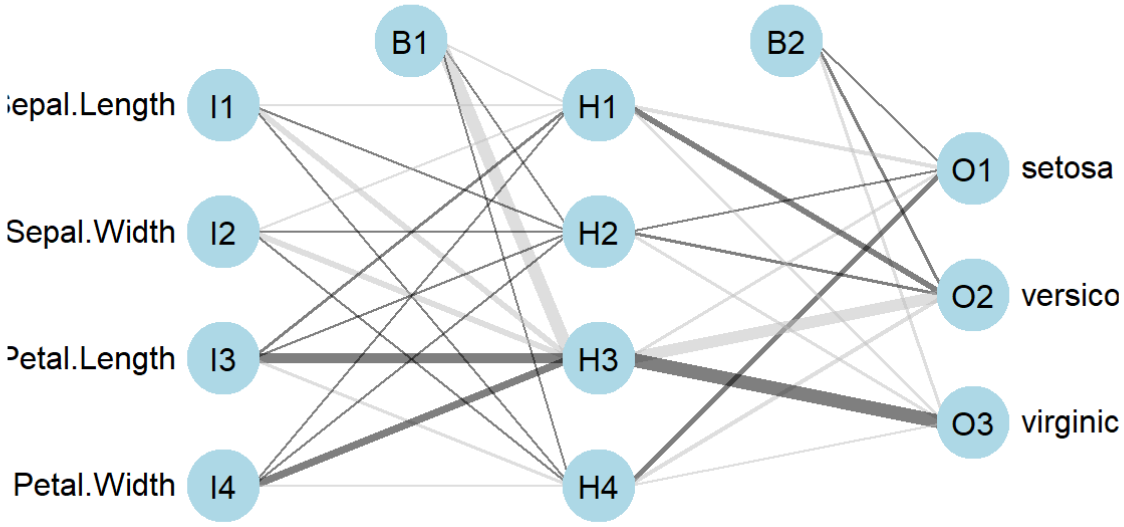
```
##  b->h3 i1->h3 i2->h3 i3->h3 i4->h3
## -23.58  -9.42  -9.00  17.13  14.04
##  b->h4 i1->h4 i2->h4 i3->h4 i4->h4
##   0.33   0.78   1.22  -2.48  -1.15
##  b->o1 h1->o1 h2->o1 h3->o1 h4->o1
##   1.17  -5.27   1.10  -2.47   6.41
##  b->o2 h1->o2 h2->o2 h3->o2 h4->o2
##   2.15   8.47   2.25 -22.32  -6.25
##  b->o3 h1->o3 h2->o3 h3->o3 h4->o3
##  -3.32  -3.20  -3.35  24.79  -0.16
```

4. Load neural net tools

```
library(NeuralNetTools)
```

5. Plot the neural network

```
plotnet(neuralModel, alpha=0.5)
```



6. Predict with model

```
neuralPredictions <- predict(
  object = neuralModel,
  newdata = test[, 1:4],
  type = "class")
```

7. Create confusion matrix

```
neuralMatrix <- confusionMatrix(
  data = neuralPredictions,
  reference = test$Species)
```

8. Inspect results

```
print(neuralMatrix)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         17          0         0
##   versicolor      0         17         0
##   virginica       0          1        15
##
## Overall Statistics
##
##                Accuracy : 0.98
##                  95% CI : (0.8935, 0.9995)
##     No Information Rate : 0.36
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.97
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                   1.00            0.9444           1.0000
## Specificity                   1.00            1.0000           0.9714
## Pos Pred Value                1.00            1.0000           0.9375
## Neg Pred Value                1.00            0.9697           1.0000
## Prevalence                    0.34            0.3600           0.3000
## Detection Rate                0.34            0.3400           0.3000
```

```
## Detection Prevalence          0.34          0.3400          0.3200
## Balanced Accuracy             1.00          0.9722          0.9857
```

6. Evaluate Classifiers

1. Compare accuracy of all three classifiers

```
print(knnMatrix$overall[1])
```

```
## Accuracy
##     0.96
```

```
print(treeMatrix$overall[1])
```

```
## Accuracy
##     0.96
```

```
print(neuralMatrix$overall[1])
```

```
## Accuracy
##     0.98
```