4.0 Ensemble Learning Overview

Jonathan De Los Santos

2/13/2021

Contents

1 Ensemble Forms

- Bagging and random forest
  - Aggregate pre-determined weak learners
- Boosting
  - Gradually connect weak learners

2 Bagging

- Effectively reduces the variance of an individual base learner
  - Improves prediction accuracy for high variance
  - The trees are not independent, called "tree correlation"
- Doesn't always improve upon an individual base learner
- Works for unstable and high variance learners
  - Decision trees and kNN
- Categorical Data

    – Most predicted values selected as final forecasts

- Continuous Data
    – Average of predicted values determines final forecasts

2.1 Bagging in R

```
library(adabag)
library(rpart)
library(rpart.plot)
library(caret)

set.seed(123)
```

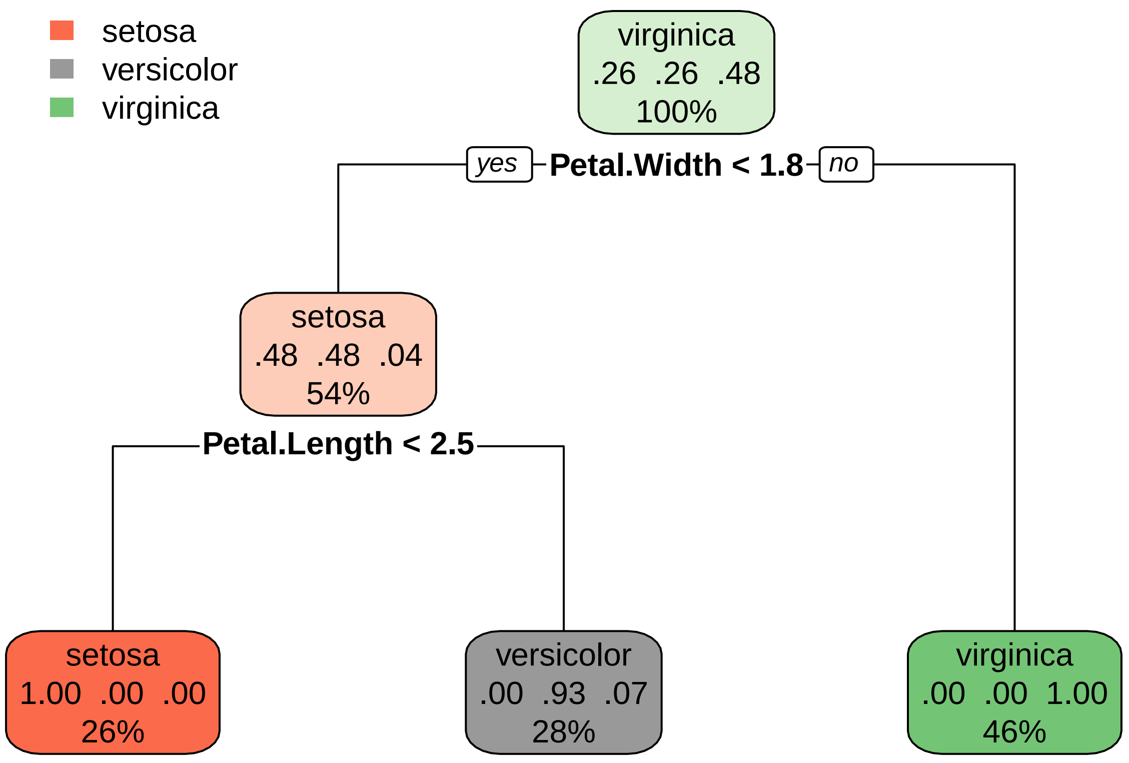
```
idx<-sample(1:150,100)
train<- iris[idx,]
test<- iris[-idx,]

# Single Tree
iris.single <- rpart(Species ~ ., data = train, method= 'class')

# Bagging with 5 trees
iris.bagging <- bagging(Species~ ., data = train, mfinal =5,
                        control = rpart.control(maxdepth=5, minsplit=5))

iris.bagging$importance
```

```
## Petal.Length  Petal.Width Sepal.Length  Sepal.Width
##     69.14617     30.85383      0.00000      0.00000
```

2.1.1 Bagging Plot

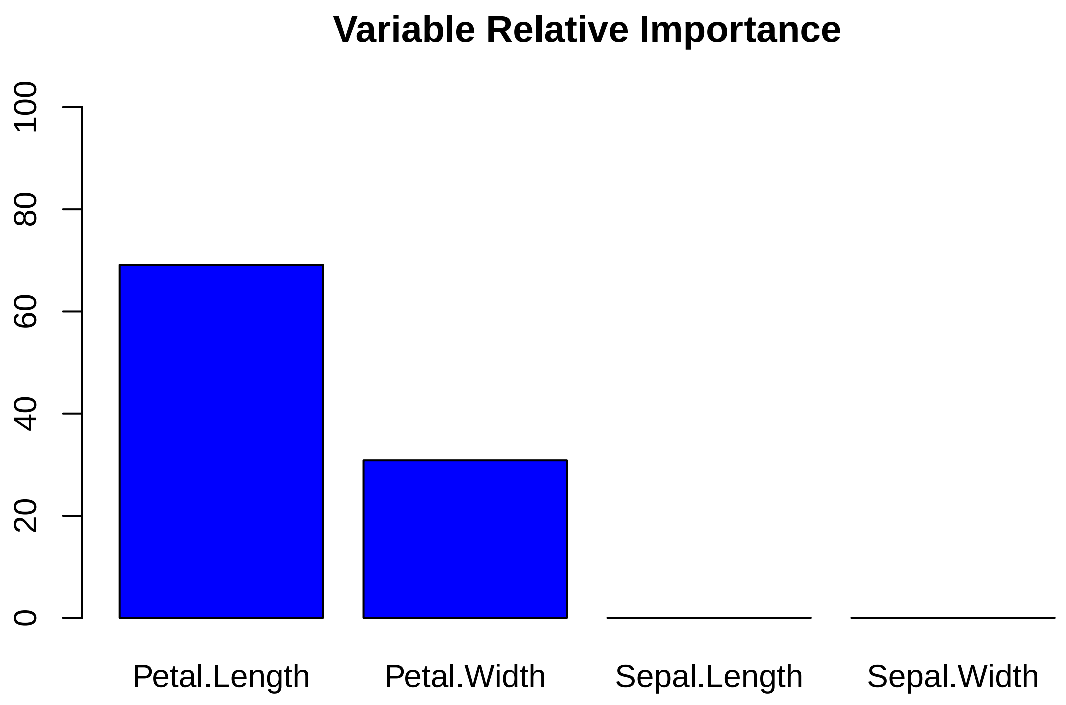
```
rpart.plot(iris.bagging$trees[[1]])
```

```
## Warning: Cannot retrieve the data used to build the model (so cannot determine roundint and is.binar
## To silence this warning:
##     Call rpart.plot with roundint=FALSE,
##     or rebuild the rpart model with model=TRUE.
```



2.1.2 Bagging Barplot

```r
barplot(iris.bagging$importance[order(iris.bagging$importance, decreasing = TRUE)], ylim = c(0, 100), ma
```



## Evaluating Bagging Model

```r
library(e1071)
pred <- predict (iris.single, test, type = "class")
confusionMatrix(pred, test$Species)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         16          0         0
##   versicolor      0         20         1
##   virginica       0          1        12
##
## Overall Statistics
##
##                Accuracy : 0.96
##                  95% CI : (0.8629, 0.9951)
##     No Information Rate : 0.42
##     P-Value [Acc > NIR] : 3.498e-16
##
##                   Kappa : 0.9388
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
```

```
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                   1.00            0.9524           0.9231
## Specificity                   1.00            0.9655           0.9730
## Pos Pred Value                1.00            0.9524           0.9231
## Neg Pred Value                1.00            0.9655           0.9730
## Prevalence                    0.32            0.4200           0.2600
## Detection Rate                0.32            0.4000           0.2400
## Detection Prevalence          0.32            0.4200           0.2600
## Balanced Accuracy             1.00            0.9589           0.9480
```

```
pred2 <- predict(iris.bagging, test, type = "class")

# Convert pred2 class to factor before building confusion matrix
confusionMatrix(factor(pred2$class), test$Species)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         16          0         0
##   versicolor      0         19         0
##   virginica       0          2        13
##
## Overall Statistics
##
##                Accuracy : 0.96
##                  95% CI : (0.8629, 0.9951)
##     No Information Rate : 0.42
##     P-Value [Acc > NIR] : 3.498e-16
##
##                   Kappa : 0.9394
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                   1.00            0.9048           1.0000
## Specificity                   1.00            1.0000           0.9459
## Pos Pred Value                1.00            1.0000           0.8667
## Neg Pred Value                1.00            0.9355           1.0000
## Prevalence                    0.32            0.4200           0.2600
## Detection Rate                0.32            0.3800           0.2600
## Detection Prevalence          0.32            0.3800           0.3000
## Balanced Accuracy             1.00            0.9524           0.9730
```

3 Boosting

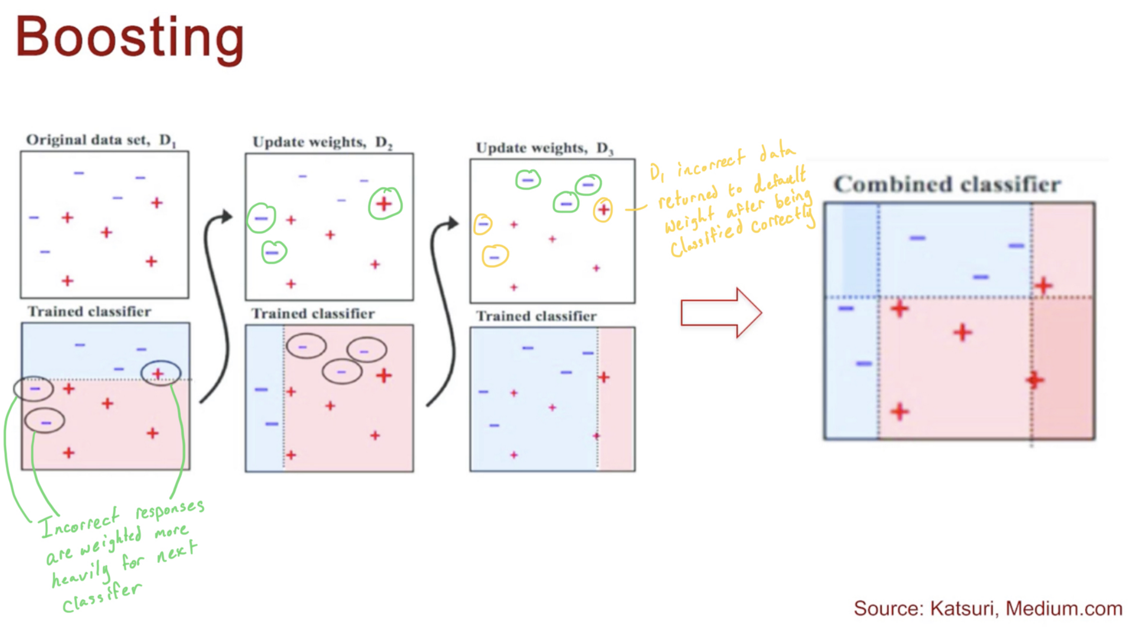

Boosting
Original data set, D_1
Update weights, D_2
Update weights, D_3
Trained classifier
Trained classifier
Trained classifier
Combined classifier
D_1 incorrect data returned to default weight after being classified correctly
Incorrect responses are weighted more heavily for next classifer
Source: Katsuri, Medium.com

Figure 1: Boosting

3.1 Bagging vs Boosting

Bagging: - Taught in parallel - Decreases variance - More errors - Appropriate when overfit is a problem Boosting: - Taught sequentially - Decreases bias - Slow and may overfit - Appropriate when low performance is a problem

3.2 Boosting in R Overview

Uses similar setup as bagging with a few differences

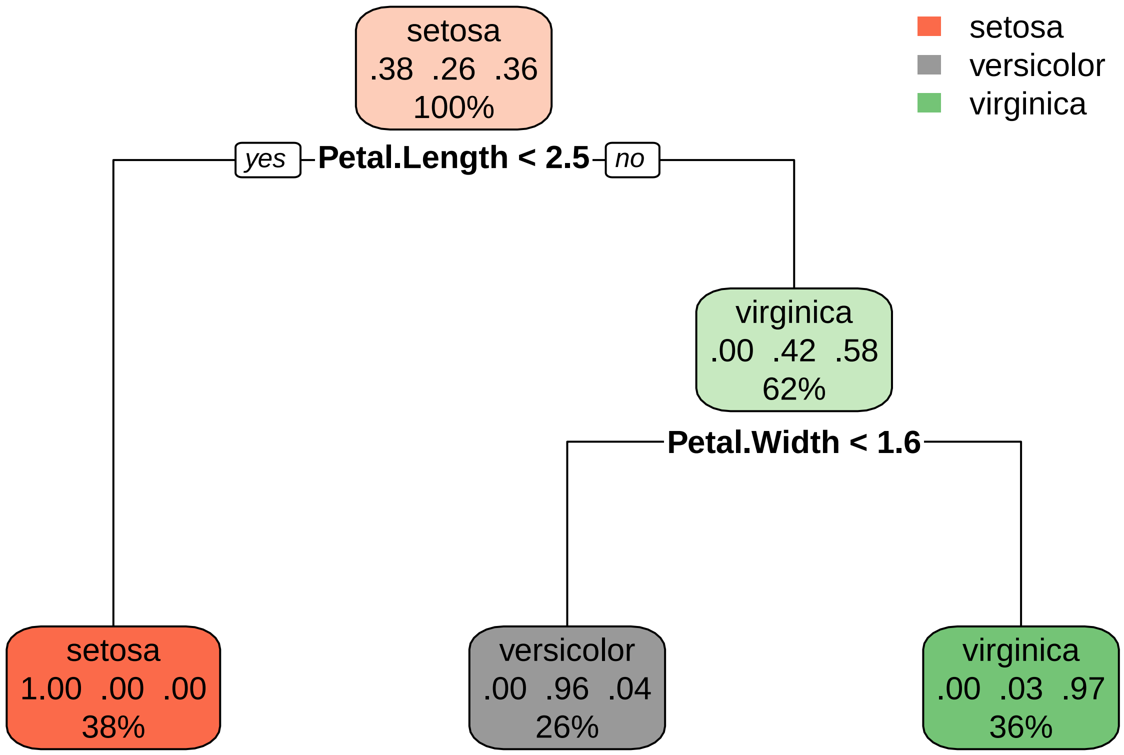
```
# Single tree setup
iris.single <- rpart(Species ~ ., data = train, method= 'class')

# Boosting with 5 trees
# Note change from "bagging" to "boosting" model, otherwise same as bagging
iris.boosting<- boosting(Species~ ., data = train, mfinal =5,
                       control = rpart.control(maxdepth=5, minsplit=5))

iris.boosting$importance
```

```
## Petal.Length  Petal.Width Sepal.Length  Sepal.Width
##    58.238326    36.321655     5.024207     0.415812
```

3.2.1 Boosting Plot

```r
rpart.plot(iris.boosting$trees[[1]])
```

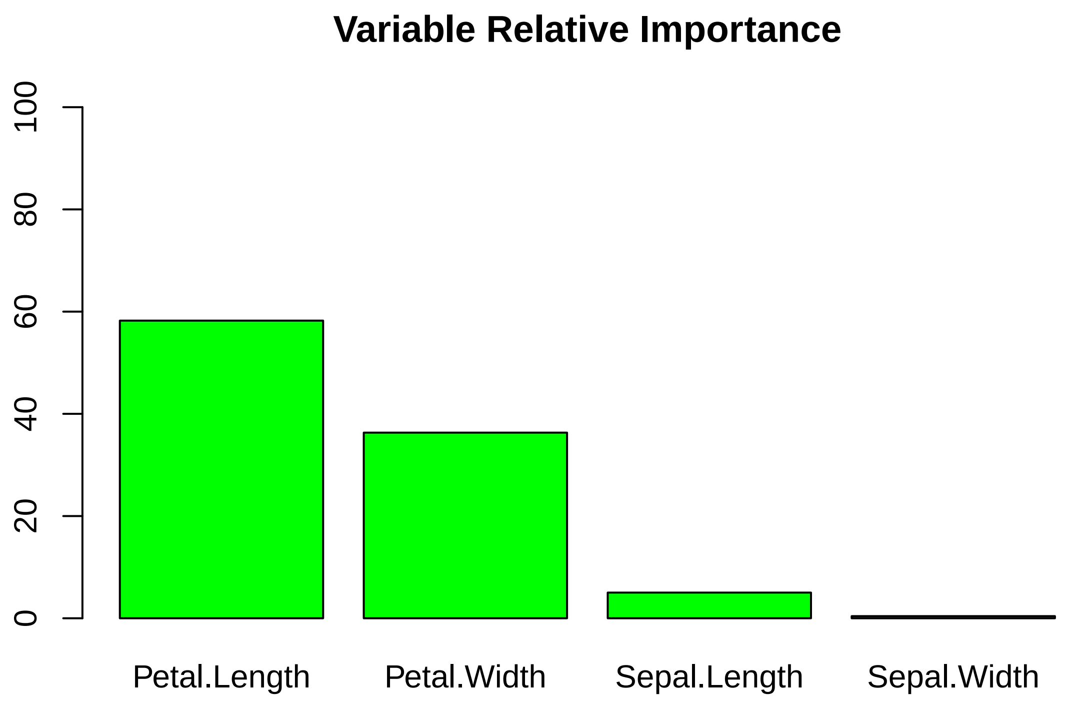
```
## Warning: Cannot retrieve the data used to build the model (so cannot determine roundint and is.binar
## To silence this warning:
##     Call rpart.plot with roundint=FALSE,
##     or rebuild the rpart model with model=TRUE.
```



3.2.2 Boosting Barplot

Note the extremely important difference from bagging: we made it green

```
barplot(iris.boosting$importance[order(iris.boosting$importance, decreasing = TRUE)], ylim = c(0, 100),
```



3.3 Evaluating Boosting Model

```
library(e1071)
pred3 <- predict(iris.single, test, type = "class")
confusionMatrix(pred3, test$Species)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         16          0         0
##   versicolor      0         20         1
##   virginica       0          1        12
##
## Overall Statistics
##
##                Accuracy : 0.96
##                  95% CI : (0.8629, 0.9951)
##     No Information Rate : 0.42
##     P-Value [Acc > NIR] : 3.498e-16
##
```

```
##                   Kappa : 0.9388
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                   1.00            0.9524           0.9231
## Specificity                   1.00            0.9655           0.9730
## Pos Pred Value                1.00            0.9524           0.9231
## Neg Pred Value                1.00            0.9655           0.9730
## Prevalence                    0.32            0.4200           0.2600
## Detection Rate                0.32            0.4000           0.2400
## Detection Prevalence          0.32            0.4200           0.2600
## Balanced Accuracy             1.00            0.9589           0.9480
```

```
pred4 <- predict(iris.boosting, test, type = "class")
confusionMatrix(factor(pred4$class), test$Species)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         16          0         0
##   versicolor      0         19         0
##   virginica       0          2        13
##
## Overall Statistics
##
##                Accuracy : 0.96
##                  95% CI : (0.8629, 0.9951)
##     No Information Rate : 0.42
##     P-Value [Acc > NIR] : 3.498e-16
##
##                   Kappa : 0.9394
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                   1.00            0.9048           1.0000
## Specificity                   1.00            1.0000           0.9459
## Pos Pred Value                1.00            1.0000           0.8667
## Neg Pred Value                1.00            0.9355           1.0000
## Prevalence                    0.32            0.4200           0.2600
## Detection Rate                0.32            0.3800           0.2600
## Detection Prevalence          0.32            0.3800           0.3000
## Balanced Accuracy             1.00            0.9524           0.9730
```

4 Extensions

These ensemble forms can be extended with further models:

Bagging ⇒ Random Forests Boosting ⇒ Gradient Boosting

4.1 Random Forest

4.1.1 Algorithm Structure

```
Given a training data set
Select number tree to build (n_trees)
for i=l to n_tree do
  Generate a bootstrap sample
  Grow tree
    for
    | Select m_try variables at random from all p variables
    | Pick the best variable
    | Split the node into two child nodes end
    end
  Use tree model stopping criteria to determine when a tree is complete

end
```

4.1.2 Hyperparameters

- The number of trees in the forest
  - Start with p x 10 trees and adjust as necessary
- For each split: m_try
  - The number of features to consider at any given split
  - Rule of thumb
    - Classification: m=Vp
    - Regression: m=p/3
- Majority voting

4.1.3 Random Forests in R

- `nodesize = 3` Number of nodes at the end of the tree
- `mtry = 2` Number of variables

The confusion table shows us the error rate of each category

```r
#install.packages("randomForest")
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.0.2

## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
# Create training data
train <- sample(1:150, 100)

# Build model
myForest <- randomForest (Species ~., nodesize = 3, mtry = 2, ntree = 15, data = iris)

# Observe confusion table
myForest$confusion
```

```
##            setosa versicolor virginica class.error
## setosa         50          0         0        0.00
## versicolor      0         47         3        0.06
## virginica       0          6        44        0.12
```

4.1.4 Evaluating Random Forest

```
# Load with test data
predict(myForest, newdata = iris[-train, ])
```

```
##          1          2          5          6          9         11         13
##     setosa     setosa     setosa     setosa     setosa     setosa     setosa
##         14         25         26         28         29         33         35
##     setosa     setosa     setosa     setosa     setosa     setosa     setosa
##         41         42         43         46         50         55         57
##     setosa     setosa     setosa     setosa     setosa versicolor versicolor
##         60         61         64         65         68         71         75
## versicolor versicolor versicolor versicolor versicolor versicolor versicolor
##         76         80         82         83         84         88         89
## versicolor versicolor versicolor versicolor versicolor versicolor versicolor
##         94         95         98        101        102        115        117
## versicolor versicolor versicolor  virginica  virginica  virginica  virginica
##        124        126        134        136        137        141        148
##  virginica  virginica  virginica  virginica  virginica  virginica  virginica
##        150
##  virginica
## Levels: setosa versicolor virginica
```

```
# Create confusion matrix
tt <- table(iris$Species[-train], predict(myForest, iris[-train,]))
```

Calculate success and error rate:

```
# Success Rate
sum(tt[row(tt) == col(tt)]) / sum(tt)
```

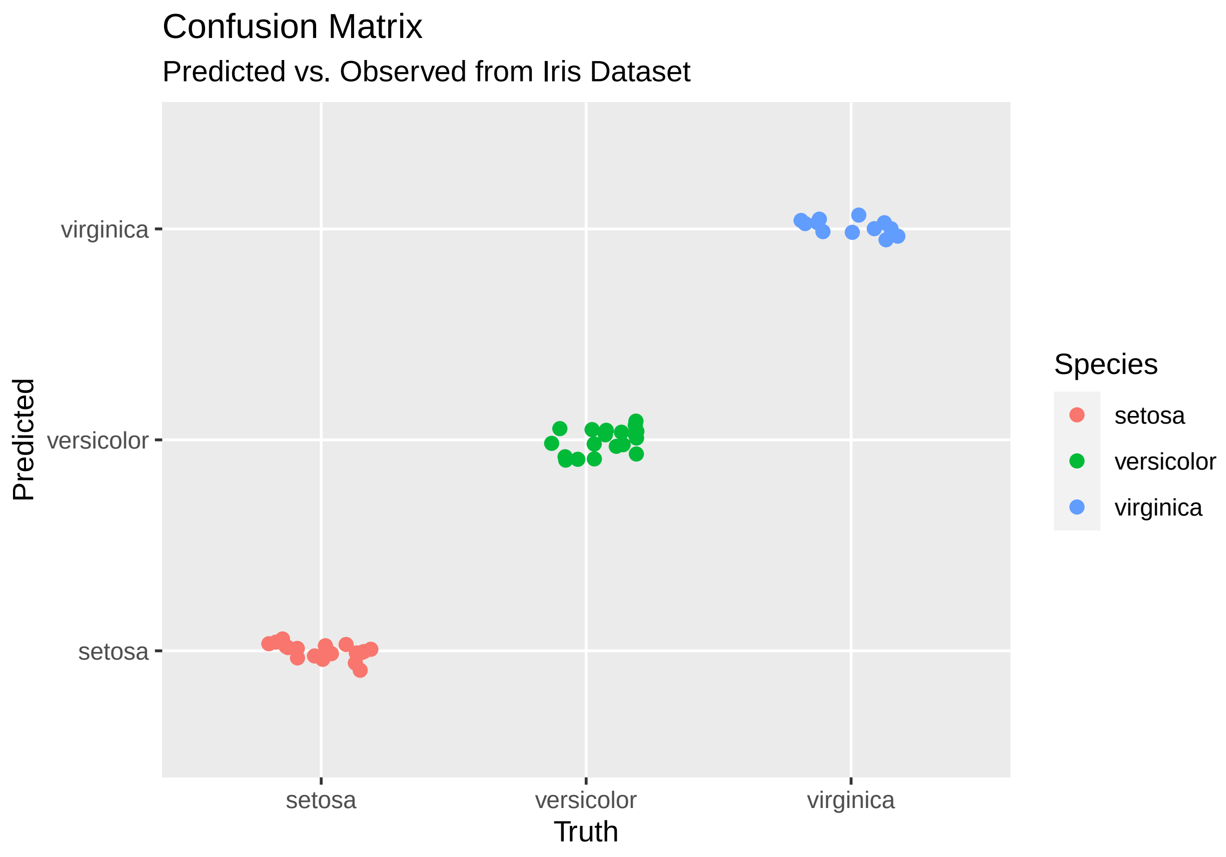
```
## [1] 1
```

```r
# Error Rate (1 - success rate)
1 - sum(tt[row(tt) == col(tt)]) / sum(tt)
```

```
## [1] 0
```

4.1.5 Evaluation Plotting

```r
library(ggplot2)
test <- iris[-train,]
test$pred <- predict(myForest, iris [-train,])

ggplot(test, aes(Species, pred, color = Species)) +
  geom_jitter(width = 0.2, height = 0.1, size = 2) +
  labs(title = "Confusion Matrix",
    subtitle = "Predicted vs. Observed from Iris Dataset",
    y = "Predicted", x = "Truth")
```


Confusion Matrix
Predicted vs. Observed from Iris Dataset
Predicted
virginica
versicolor
setosa
setosa
versicolor
virginica
Truth
Species
setosa
versicolor
virginica