Assignment1 stk-in4300

Vivian De La O Mellum

22 9 2020

In 1936 Ronald Fisher published a paper *The use of multiple measurements in taxonomic problems* and with time became perhaps the best known dataset in pattern recognition litterature. The dataset is sometimes called *the Anderson's Iris data set* and contains 150 cases and 5 variables; sepal length, sepal width, petal length, petal width and species. The data was collected on the following species; *Iris setosa, versicolor, and virginica*, which are depicted in fig. 1, respectively.

The purpose of this analysis is to compare three types of classification algorithms; linear discriminant analysis (LDA), classification and Regression Trees (CART) and k-Nearest Neighbors (kNN).

Iris Setosa

Iris Versicolor

Iris Virginica

Loading the data

We start by downloading the data from the URL: http://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data and specify the columnames.

A small note; in the "*Data set Information*" the following is disclosed; *This data differs from the data presented in Fishers article (identified by Steve Chadwick, spchadwick '@' espeedaz.net). The 35th sample should be: 4.9,3.1,1.5,0.2,"Iris-setosa" where the error is in the fourth feature. The 38th sample: 4.9,3.6,1.4,0.1,"Iris-setosa" where the errors are in the second and third features.* Data set dimensions; n = 150 p = 5

```
library(caret)
```

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

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

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

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
```

Inspecting and identifying the variables from the dataset

We observe that all of the nummerical values are of the same scale and same range, hence there is no need to apply any statistial procedures to modify the raw data. By inspectin we observe that Species has class "character", for simplisity we might want to change class to "factor". By applying as.factor() to the class we replace the columnvector with the corresponding factor. We need to do this before splitting into train and test sets.

```
summary(iris.UCI)
```

```
##   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
##  Length:150
##  Class :character
##  Mode  :character
##
##
##
```

```
#show(iris.UCI)
```

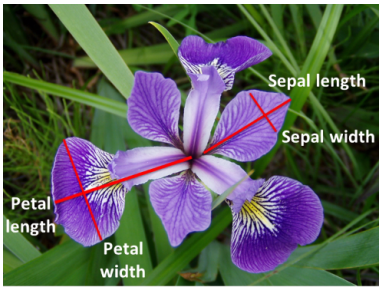
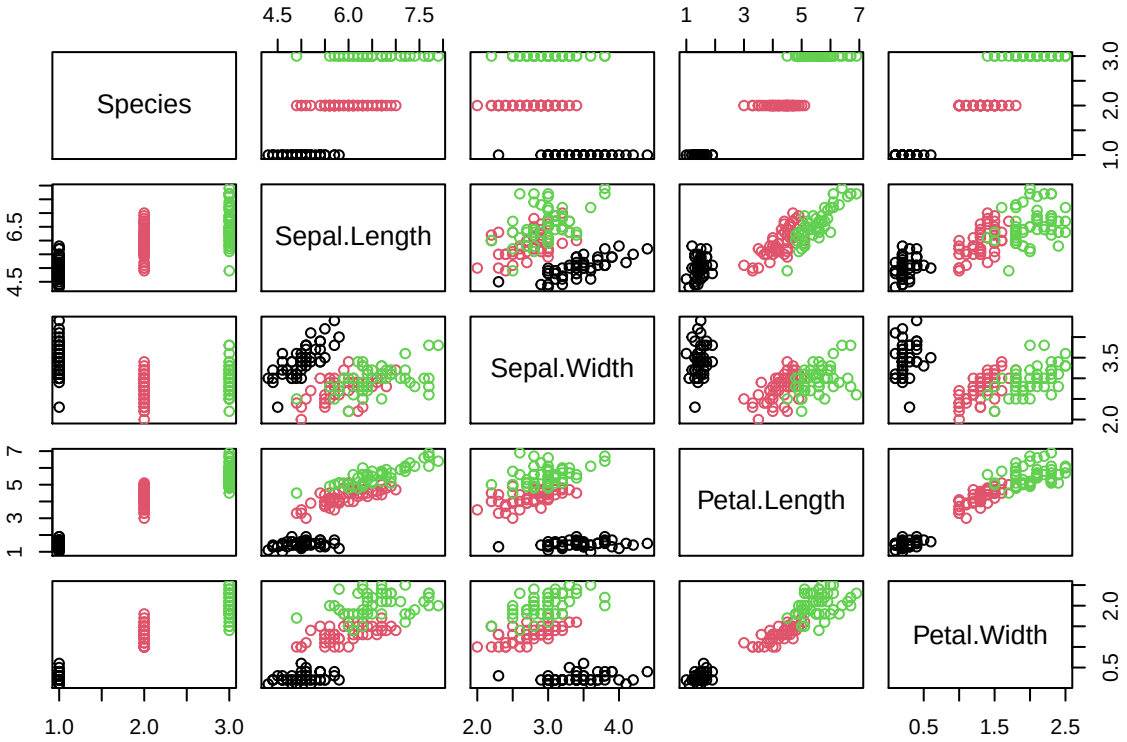


Figure 1: fig. 2

Fig.2 illustrates what the variables measure and how it is used to identify the species.

To better understand the data set we can create a general plot allowing us to visualize how the variables interact with eachother in pairs. As anticipated, species is a categorical variable

```
data("iris")
pairs(Species~., data=iris, col=iris$Species)
```



Spliting the data

We now split the data into a training and a test set.

```
## 70% of the rows in the original dataset is used for training

iris.UCI$Species <- as.factor(iris.UCI$Species)

train_index <- createDataPartition(iris.UCI$Species, p=0.70, list=FALSE)
```

```
## The remaining 30% is used for testing the models after training
test <- iris.UCI[-train_index,]
nrow(test)      ## making sure that the % is correct; 30% of 150 is 45.
```

```
## [1] 45
```

```
## Training set; 70%
train <- iris.UCI[train_index,]
#show(train)

nrow(train)     ## making sure that the % is correct; 70% of 105 is 105.
```

```
## [1] 105
```

Dimensions of the traning and test set

```
dim(train)
```

```
## [1] 105   5
```

```
dim(test)
```

```
## [1] 45  5
```

Visualization of the test set

```
## further splitting the training set into input and output variables.
x <- train[,1:4]  ## input; Sepal.Length Sepal.Width Petal.Length Petal.Width
y <- train[,5]    ## output; Species
```

To further expand our understanding of the dataset we want to study the visuals through univariate and multivariate plots. This will give a better understanding of each attribute and the relationship between each attribute, this will then, possibly, uncover overlooked attributes from just looking at the tables. If we compare the summary of the full data set and training set we observe that there is close to no differece between the two, which can be though of as a good split in data.

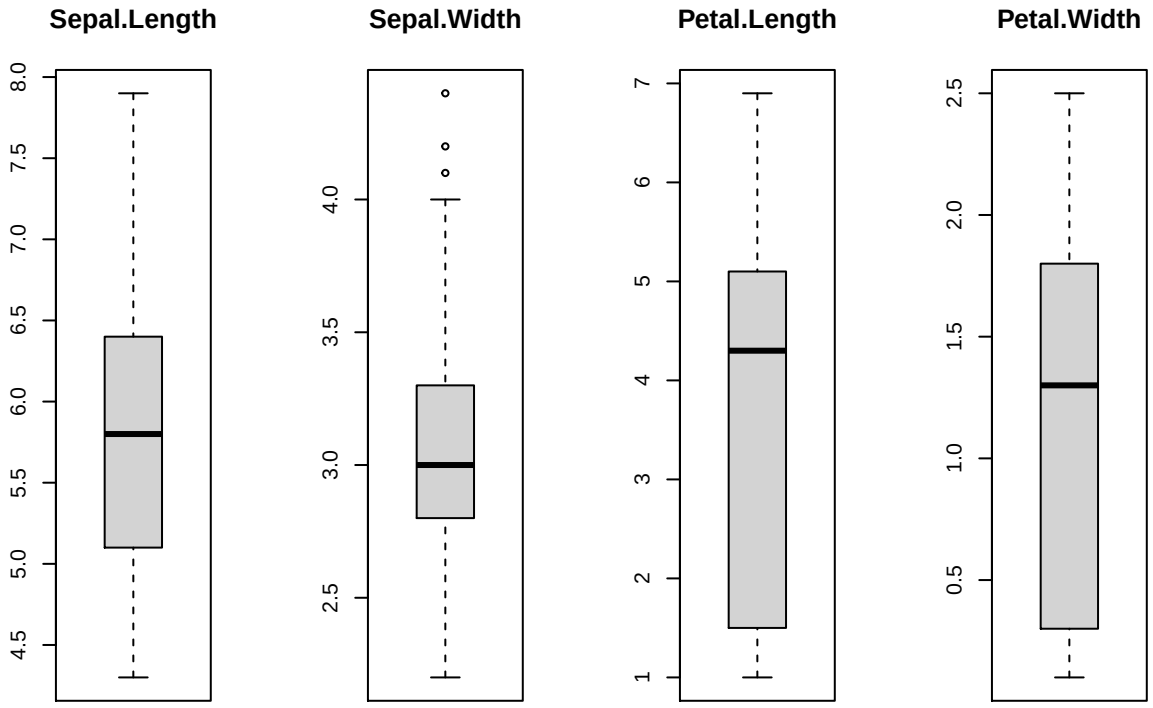
```
summary(train)
```

```
##   Sepal.length    Sepal.width     Petal.length    Petal.width
##  Min.   :4.300   Min.   :2.200   Min.   :1.000   Min.   :0.100
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.500   1st Qu.:0.300
##  Median :5.800   Median :3.000   Median :4.300   Median :1.300
##  Mean   :5.863   Mean   :3.063   Mean   :3.773   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    :35
##  Iris-versicolor:35
##  Iris-virginica :35
##
##
##
```

Univariate

The two first boxplots representing the sepal attributes are comparatively short, meaning that the sepal length and width are in general agreement across species. The conversly is interpreted by the boxplots representing the petal lenght and width. By inspection, if we compare the sepal attributes with the petal attributes it seems that the there might be similarities at certain parts of the attributes, like two species share similar measurments, and that the remaining species differs from the other two. This is also confirmed in the data set information given in the UCI repository; "https://archive.ics.uci.edu/ml/datasets/Iris" "One class is linearly separable from the other 2; the latter are NOT linearly separable from each other".

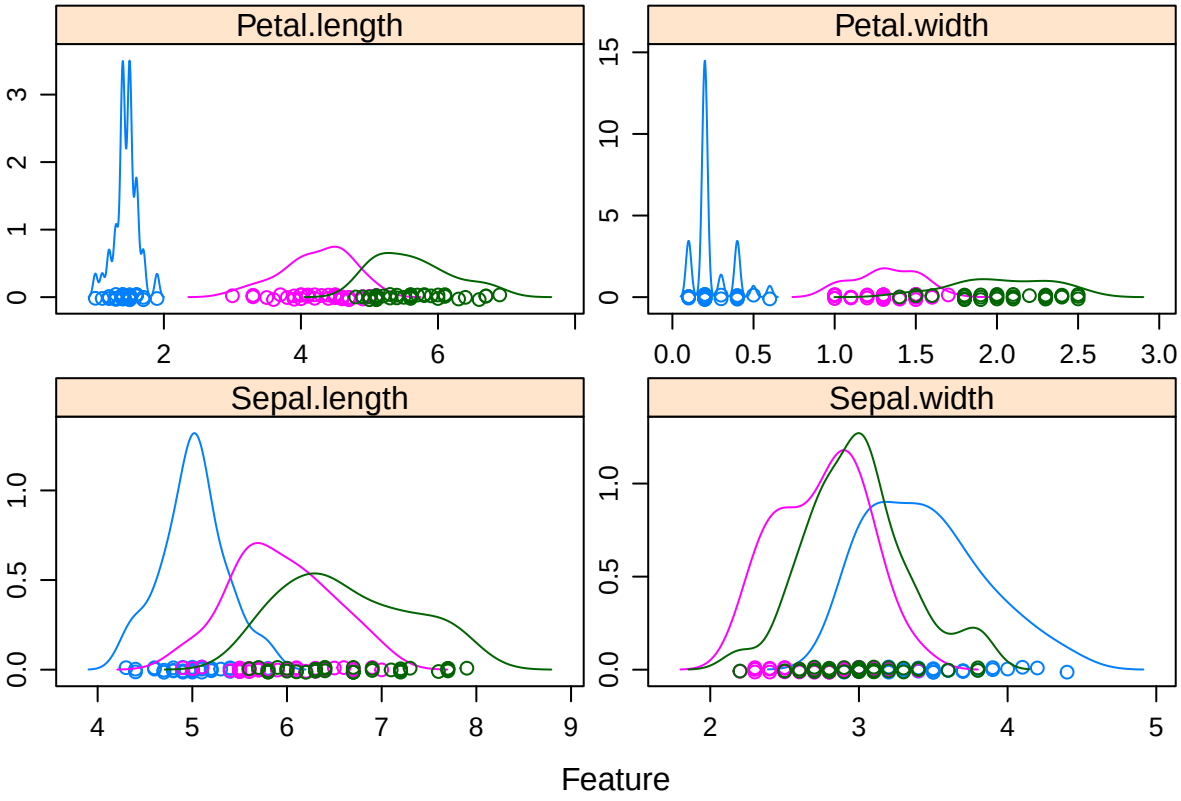
```
par(mfrow=c(1,4))
boxplot(x[,1], main=names(iris)[1])
boxplot(x[,2], main=names(iris)[2])
boxplot(x[,3], main=names(iris)[3])
boxplot(x[,4], main=names(iris)[4])
```



Multivariate

As mentioned two of the classes are not linearly separable from each other and from the density plot we observe that two of the disfributions seem to come from the same distribution with some difference in scale and location parameters. Another interesting observation is that the third class seems to be of the same distribution as the two classes which are not separable, but has a positice kurtosis. Hence, it could be of interest to investigate how similar/dissimilar the flowers are in terms of distribution, but that is not the purpose of our analysis.

```
# density plots for each attribute by class value
scales <- list(x=list(relation="free"), y=list(relation="free"))
featurePlot(x=x, y=y, plot="density", scales=scales)
```



Evaluating the algorithms

We now want to evaluate the three algorithms; linear discriminant analysis (LDA), classification and Regression Trees (CART) and k-Nearest Neighbors (kNN).

Due to bias-variance trade-off associated with the value of K, the range of choise usually falls between 5-10 folds and the most common choise of K is 10, hence our cholse of K. Of course, the bias will increase and the difference between the training and the test set becomes smaller -hence making the model prone to overfitting.

```
## Cross-validation, K=10

train_control <- trainControl(method = "cv", number=10)
metric <- "Accuracy"
```

The metric accuracy returns the percentage of accuratly predicted instances. This will later on be used when comparing the algorithms.

We now fit the models with our trainin set.

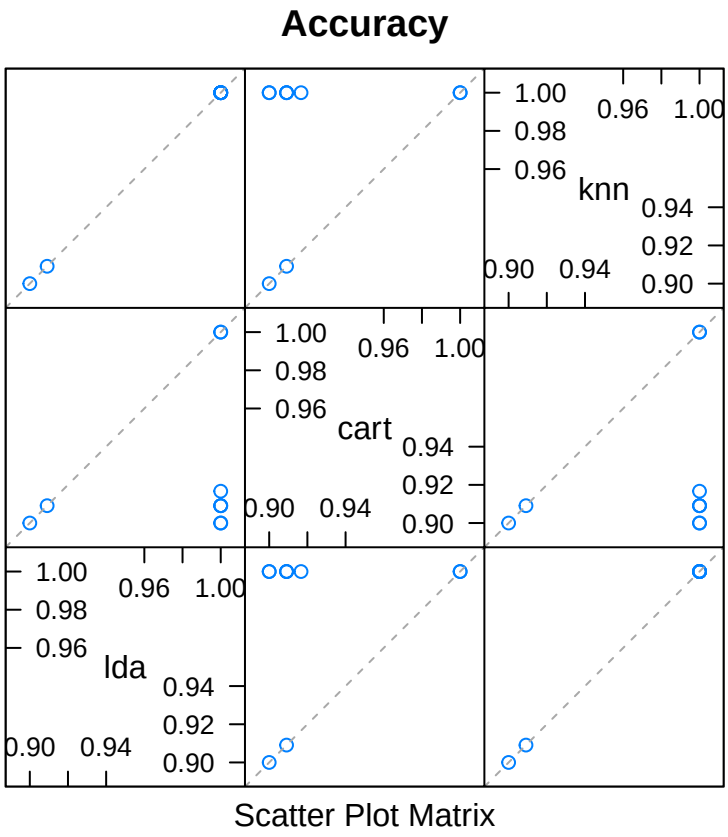
```
# LDA; Linear discriminant analysis
set.seed(1234)
fit.lda <- train(Species~., data=train, method="lda", metric=metric, trControl=train_control)
# CART; Classification- and regression trees
set.seed(1234)
fit.cart <- train(Species~., data=train, method="rpart", metric=metric, trControl=train_control)
# kNN; k neares neighbours
set.seed(1234)
fit.knn <- train(Species~., data=train, method="knn", metric=metric, trControl=train_control)

resamps <- resamples(list(lda=fit.lda, cart=fit.cart, knn=fit.knn))
resamps
```

```
##
## Call:
## resamples.default(x = list(lda = fit.lda, cart = fit.cart, knn = fit.knn))
##
## Models: lda, cart, knn
## Number of resamples: 10
## Performance metrics: Accuracy, Kappa
## Time estimates for: everything, final model fit
```

```
summary(resamps)
```

```
##
## Call:
## summary.resamples(object = resamps)
##
## Models: lda, cart, knn
## Number of resamples: 10
##
## Accuracy
##      Min.   1st Qu.    Median      Mean   3rd Qu. Max. NA's
## lda   0.9 1.0000000 1.0000000 0.9809091 1.0000000    1    0
## cart  0.9 0.9022727 0.9090909 0.9253030 0.9147727    1    0
## knn   0.9 1.0000000 1.0000000 0.9809091 1.0000000    1    0
##
## Kappa
##           Min.   1st Qu.    Median      Mean   3rd Qu. Max. NA's
## lda  0.8484848 1.0000000 1.0000000 0.9710985 1.0000000    1    0
## cart 0.8484848 0.8536847 0.8633488 0.8876111 0.8722994    1    0
## knn  0.8484848 1.0000000 1.0000000 0.9710985 1.0000000    1    0
```

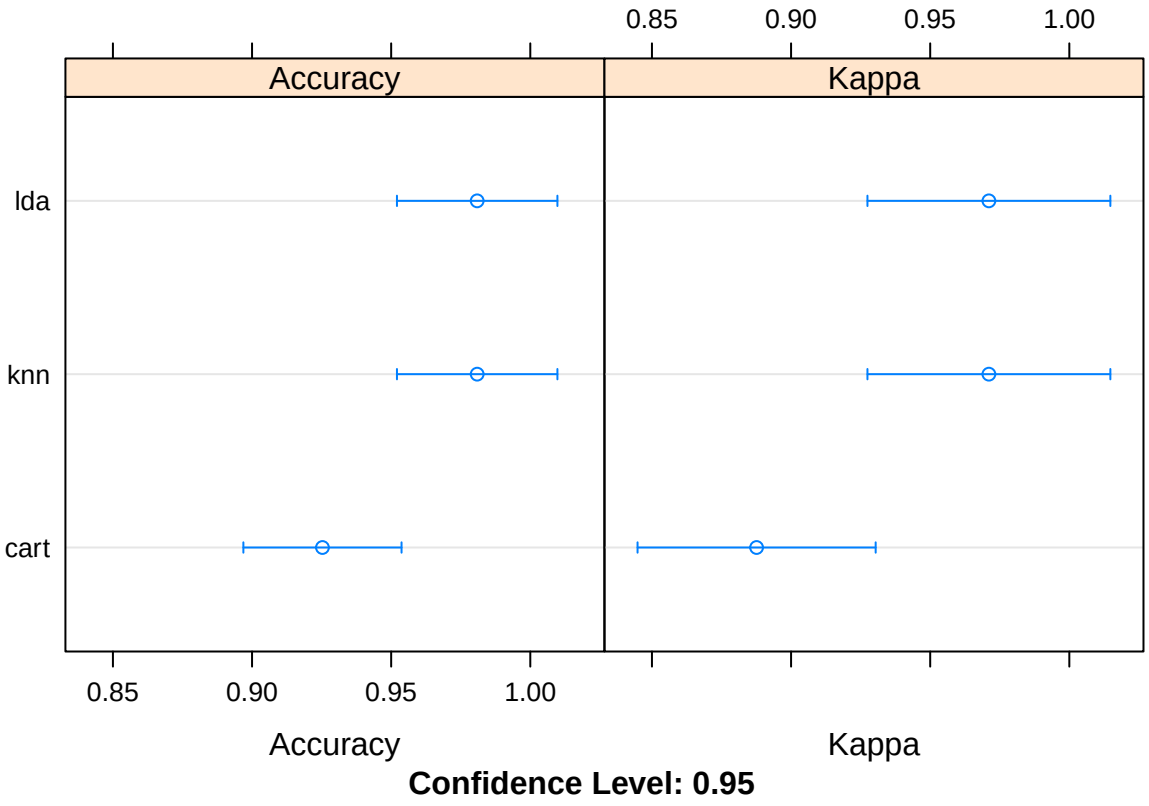
```
splom(resamps)
```



In terms of correlation there seems not to be any corelation between algorithms.

We now whish to evaluate which algorithm was the most accurate and can do so by comparing the accuracy of the models

```
dotplot(resamps)
```



We observe that the most accurate model was the LDA, but kNN also seems to be a good alternative.

```
print(fit.lda)
```

```
## Linear Discriminant Analysis
##
## 105 samples
##   4 predictor
##   3 classes: 'Iris-setosa', 'Iris-versicolor', 'Iris-virginica'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 93, 96, 95, 94, 95, 95, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.9809091  0.9710985
```

```
print(fit.knn)
```

```
## k-Nearest Neighbors
##
## 105 samples
##   4 predictor
##   3 classes: 'Iris-setosa', 'Iris-versicolor', 'Iris-virginica'
```

```
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 93, 96, 95, 94, 95, 95, ...
## Resampling results across tuning parameters:
##
##   k  Accuracy   Kappa
##   5  0.9718182  0.9575182
##   7  0.9809091  0.9710985
##   9  0.9709091  0.9561731
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 7.
```

```
print(fit.cart)
```

```
## CART
##
## 105 samples
##   4 predictor
##   3 classes: 'Iris-setosa', 'Iris-versicolor', 'Iris-virginica'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 93, 96, 95, 94, 95, 95, ...
## Resampling results across tuning parameters:
##
##   cp         Accuracy   Kappa
##   0.0000000  0.9253030  0.88761110
##   0.4428571  0.8519697  0.78046825
##   0.5000000  0.3557576  0.08571429
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.
```

Predictions and confusion matrix; LDA

To see how the algorithm is performing we can study the confusion matrix and inspect sencitivity and specifisity. The predictions are overall satisfying with one missclassification between Iris-versicolor and Iris-virginica, this was almost expected since the flowers seem to posess many similar attributes. The overall statistics show that our model is 95.5% accurate, which is fairly good. From the sensitivity and specificity statistics we observe that the models ability determine the true positive rate (sensitivity) and true negative rate (specificity) for the classes “versicolor” and “virginica” was not to accurat. But as mentioned, this is probably just due to the similarities between the classes and not necessarily due to our model.

```r
pred <- predict(fit.lda, test)
confusionMatrix(pred, test$Species)
```

```
## Confusion Matrix and Statistics
##
##                  Reference
## Prediction        Iris-setosa Iris-versicolor Iris-virginica
##   Iris-setosa              15               0              0
##   Iris-versicolor           0              14              0
##   Iris-virginica            0               1             15
##
## Overall Statistics
##
##                Accuracy : 0.9778
##                  95% CI : (0.8823, 0.9994)
##     No Information Rate : 0.3333
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9667
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Iris-setosa Class: Iris-versicolor
## Sensitivity                      1.0000                 0.9333
## Specificity                      1.0000                 1.0000
## Pos Pred Value                   1.0000                 1.0000
## Neg Pred Value                   1.0000                 0.9677
## Prevalence                       0.3333                 0.3333
## Detection Rate                   0.3333                 0.3111
## Detection Prevalence             0.3333                 0.3111
## Balanced Accuracy                1.0000                 0.9667
##                      Class: Iris-virginica
## Sensitivity                         1.0000
## Specificity                         0.9667
## Pos Pred Value                      0.9375
## Neg Pred Value                      1.0000
## Prevalence                          0.3333
## Detection Rate                      0.3333
## Detection Prevalence                0.3556
## Balanced Accuracy                   0.9833
```

Conclusion

Of the three classification algorithms the LDA was the most accurate. For further studies it would as mentioned be interesting to study the underlying distributions to better understand how we could improve the algorithms used when predicting data sets which have classes that are linearly separable and not linearly separable.