Targil 2

Tamar Saad & Or Arbel & Rachel Weinberger 208812628

5/19/2022

Abstract

In this assignment we took a table containing data about different types of seeds, and we wanted to use the data in order to create a machine learning model that will classify seeds by their type.

Libraries uploading:

```
library(plyr)
library(dplyr)
library(stringr)
library(gplots)
library(ggplot2)
library(forcats)
library(data.table)
library(reshape2)
library(affycoretools)
library(factoextra)
library(ggvis)
library(class)
library(gmodels)
library(plotly)
library(C50)
```

Upload and examine the data

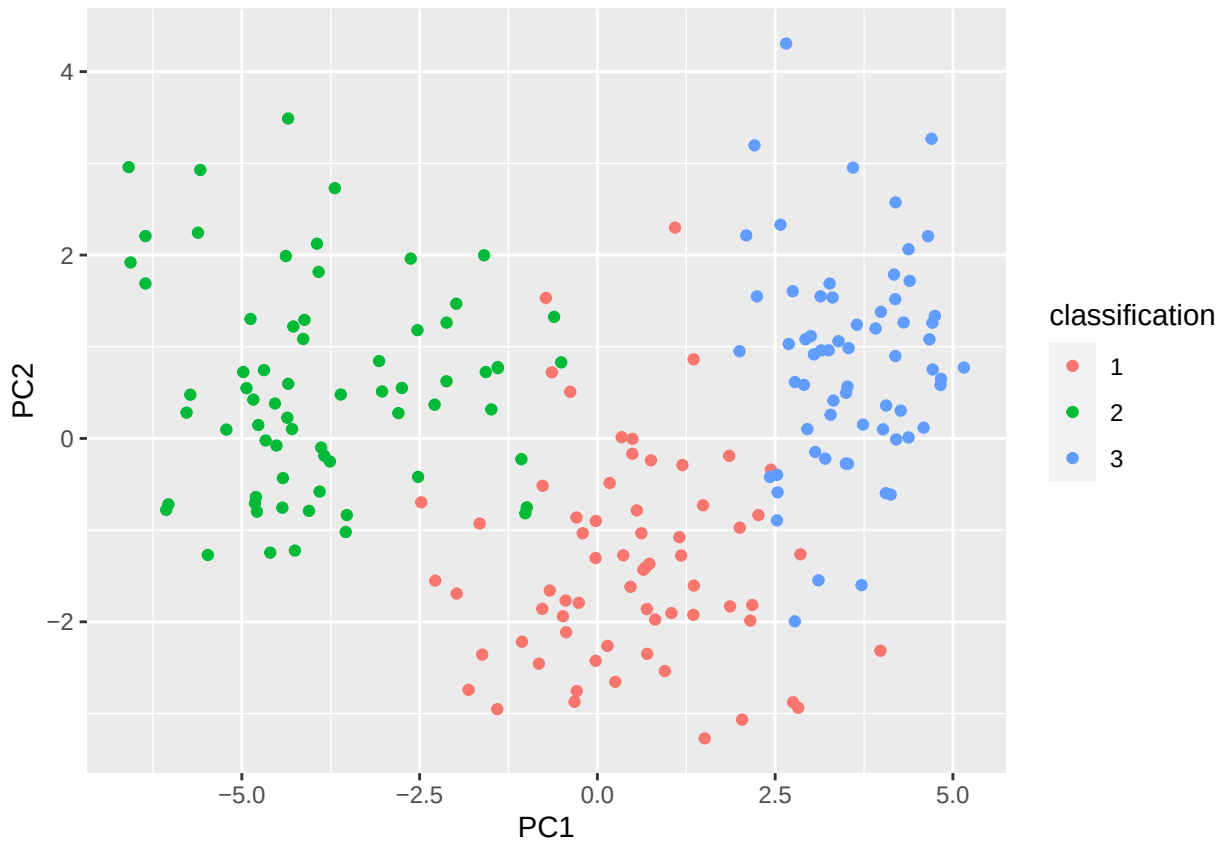
```
seeds<-read.csv("seeds.csv")
summary(seeds)
```

```
##       Area         Perimeter      Compactness     Kernel.Length
##  Min.   :10.59   Min.   :12.41   Min.   :0.8081   Min.   :4.899
##  1st Qu.:12.33   1st Qu.:13.47   1st Qu.:0.8571   1st Qu.:5.267
##  Median :14.43   Median :14.37   Median :0.8734   Median :5.541
##  Mean   :14.92   Mean   :14.60   Mean   :0.8708   Mean   :5.643
##  3rd Qu.:17.45   3rd Qu.:15.80   3rd Qu.:0.8868   3rd Qu.:6.002
##  Max.   :21.18   Max.   :17.25   Max.   :0.9183   Max.   :6.675
##   Kernel.Width   Asymmetry.Coeff  Kernel.Groove        Type
##  Min.   :2.630   Min.   :0.7651   Min.   :4.519   Min.   :1.000
##  1st Qu.:2.954   1st Qu.:2.5700   1st Qu.:5.046   1st Qu.:1.000
##  Median :3.245   Median :3.6310   Median :5.228   Median :2.000
##  Mean   :3.266   Mean   :3.6992   Mean   :5.421   Mean   :1.995
##  3rd Qu.:3.564   3rd Qu.:4.7990   3rd Qu.:5.879   3rd Qu.:3.000
##  Max.   :4.033   Max.   :8.3150   Max.   :6.550   Max.   :3.000
```

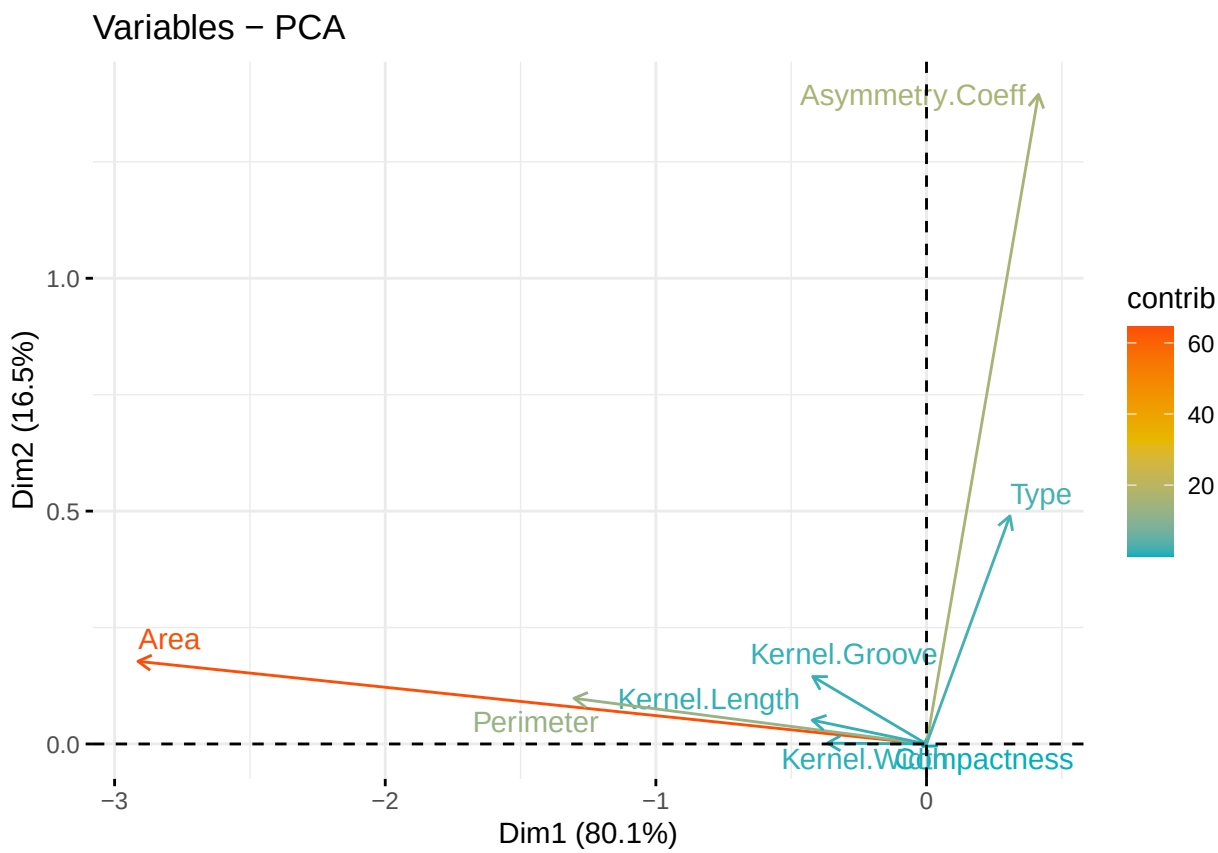
After examining the summary table, it seems there is no need in cleaning the data or normalizing one of the features. All the features are numeric, and all are in a standard range.

Before we create a classification model using machine learning tools, we want to see if there is a distinct difference between the seed types. If the given data doesn't reflect the difference between the seeds types, maybe there will be no use in creating a classification model. In order to examine it, we create a PCA plot. In addition, we want to look at the contribution of each feature to the PCA components, so we can see what features are the most significant.

```
pca <- prcomp(seeds)
pca_to_show <- data.frame(
  PC1 = pca$x[, 1],
  PC2 = pca$x[, 2],
  classification = as.factor(seeds$Type)
)

ggplot(pca_to_show, aes(x = PC1, y = PC2, col = classification)) +
  geom_point()
```




```
fviz_pca_var(pca,
             col.var = "contrib", # Color by contributions to the PC
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
             repel = TRUE     # Avoid text overlapping
             )
```



As we can see, the different types of seeds cluster separately, so the features that we have can be used to create a classification model. Moreover, we can see that the features that contribute to the variability the most are the 'Area' feature, the 'Asymmetry Coeff' and the 'Perimeter'. So, if we will have trouble with getting good results from the classification models, perhaps we could remove some of our less contributing features in order to reduce background noise.

Classification models:

We chose to use two different machine learning models in order to classify the data: KNN and Decision Tree.

KNN:

We wanted to see the distribution of the types in the data, to see if we have enough samples from each type.

```
seeds$Type<-as.factor(seeds$Type)
table(seeds$Type)
```

```
##
##  1  2  3
## 66 68 65
```

Normalize the data by MIN/MAX:

```
# create the min/max normalization function:
normalize <- function(x) {
return ((x - min(x)) / (max(x) - min(x)))
}
```

```r
#normalize the data without the type column(8)
seeds_normalize <- as.data.frame(lapply(seeds[1:7], normalize))

str(seeds_normalize)
```

```
## 'data.frame':    199 obs. of  7 variables:
##  $ Area           : num  0.441 0.405 0.349 0.307 0.524 ...
##  $ Perimeter      : num  0.502 0.446 0.347 0.316 0.533 ...
##  $ Compactness    : num  0.571 0.662 0.879 0.793 0.865 ...
##  $ Kernel.Length  : num  0.486 0.369 0.221 0.239 0.427 ...
##  $ Kernel.Width   : num  0.486 0.501 0.504 0.534 0.664 ...
##  $ Asymmetry.Coeff: num  0.1928 0.0335 0.2561 0.1979 0.0781 ...
##  $ Kernel.Groove  : num  0.345 0.215 0.151 0.141 0.323 ...
```

In the beginning we shuffle the data and split it to train (1) and test (2) by creating an index

```r
# create an index with the desired proportions
set.seed(123)
ind <- sample(2, nrow(seeds), replace=TRUE, prob=c(0.8, 0.2))
```

Create a training set and a test set:

```r
# training set
seeds.training_n <- seeds_normalize[ind==1, 1:7]

# Compose training labels
seeds.trainLabels <- seeds[ind==1,8]

# test set
seeds.test_n <- seeds_normalize[ind==2, 1:7]

# Compose test labels
seeds.testLabels <- seeds[ind==2, 8]

# check if we got about 30% from each seed type
prop.table(table(seeds.testLabels))
```

```
## seeds.testLabels
##         1         2         3
## 0.3513514 0.3513514 0.2972973
```

```r
prop.table(table(seeds.trainLabels))
```

```
## seeds.trainLabels
##         1         2         3
## 0.3271605 0.3395062 0.3333333
```

Build 3 models with different values of k: 3, 9, and 15

```r
seeds_pred_k3 <- knn(train = seeds.training_n, test = seeds.test_n,
                     cl = seeds.trainLabels, k=3)
seeds_pred_k9 <- knn(train = seeds.training_n, test = seeds.test_n,
                     cl = seeds.trainLabels, k=9)
seeds_pred_k15 <- knn(train = seeds.training_n, test = seeds.test_n,
                      cl = seeds.trainLabels, k=15)
```

Let's Examine the results of the different models:

```r
CrossTable(x = seeds.testLabels, y = seeds_pred_k3, prop.chisq=FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  37
##
##
##                  | seeds_pred_k3
## seeds.testLabels |         1 |         2 |         3 | Row Total |
## -----------------|-----------|-----------|-----------|-----------|
##                1 |        12 |         1 |         0 |        13 |
##                  |     0.923 |     0.077 |     0.000 |     0.351 |
##                  |     1.000 |     0.071 |     0.000 |           |
##                  |     0.324 |     0.027 |     0.000 |           |
## -----------------|-----------|-----------|-----------|-----------|
##                2 |         0 |        13 |         0 |        13 |
##                  |     0.000 |     1.000 |     0.000 |     0.351 |
##                  |     0.000 |     0.929 |     0.000 |           |
##                  |     0.000 |     0.351 |     0.000 |           |
## -----------------|-----------|-----------|-----------|-----------|
##                3 |         0 |         0 |        11 |        11 |
##                  |     0.000 |     0.000 |     1.000 |     0.297 |
##                  |     0.000 |     0.000 |     1.000 |           |
##                  |     0.000 |     0.000 |     0.297 |           |
## -----------------|-----------|-----------|-----------|-----------|
##     Column Total |        12 |        14 |        11 |        37 |
##                  |     0.324 |     0.378 |     0.297 |           |
## -----------------|-----------|-----------|-----------|-----------|
##
##
```

```r
CrossTable(x = seeds.testLabels, y = seeds_pred_k9, prop.chisq=FALSE)
```

```
##
```

```
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  37
##
##
##                  | seeds_pred_k9
## seeds.testLabels |         1 |         2 |         3 | Row Total |
## -----------------|-----------|-----------|-----------|-----------|
##                1 |        12 |         1 |         0 |        13 |
##                  |     0.923 |     0.077 |     0.000 |     0.351 |
##                  |     0.923 |     0.071 |     0.000 |           |
##                  |     0.324 |     0.027 |     0.000 |           |
## -----------------|-----------|-----------|-----------|-----------|
##                2 |         0 |        13 |         0 |        13 |
##                  |     0.000 |     1.000 |     0.000 |     0.351 |
##                  |     0.000 |     0.929 |     0.000 |           |
##                  |     0.000 |     0.351 |     0.000 |           |
## -----------------|-----------|-----------|-----------|-----------|
##                3 |         1 |         0 |        10 |        11 |
##                  |     0.091 |     0.000 |     0.909 |     0.297 |
##                  |     0.077 |     0.000 |     1.000 |           |
##                  |     0.027 |     0.000 |     0.270 |           |
## -----------------|-----------|-----------|-----------|-----------|
##     Column Total |        13 |        14 |        10 |        37 |
##                  |     0.351 |     0.378 |     0.270 |           |
## -----------------|-----------|-----------|-----------|-----------|
##
##
```

```
CrossTable(x = seeds.testLabels, y = seeds_pred_k15, prop.chisq=FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  37
##
##
##                  | seeds_pred_k15
```

```
## seeds.testLabels |         1 |         2 |         3 | Row Total |
## -----------------|-----------|-----------|-----------|-----------|
##                1 |        11 |         1 |         1 |        13 |
##                  |     0.846 |     0.077 |     0.077 |     0.351 |
##                  |     0.917 |     0.071 |     0.091 |           |
##                  |     0.297 |     0.027 |     0.027 |           |
## -----------------|-----------|-----------|-----------|-----------|
##                2 |         0 |        13 |         0 |        13 |
##                  |     0.000 |     1.000 |     0.000 |     0.351 |
##                  |     0.000 |     0.929 |     0.000 |           |
##                  |     0.000 |     0.351 |     0.000 |           |
## -----------------|-----------|-----------|-----------|-----------|
##                3 |         1 |         0 |        10 |        11 |
##                  |     0.091 |     0.000 |     0.909 |     0.297 |
##                  |     0.083 |     0.000 |     0.909 |           |
##                  |     0.027 |     0.000 |     0.270 |           |
## -----------------|-----------|-----------|-----------|-----------|
##     Column Total |        12 |        14 |        11 |        37 |
##                  |     0.324 |     0.378 |     0.297 |           |
## -----------------|-----------|-----------|-----------|-----------|
##
##
```

We tried to do z-score normalization too, we're doing the same steps as in min-max normalization:

```r
# z-score normalization
seeds_z <- as.data.frame(scale(seeds[-8]))

seeds.train_z <- seeds_z[ind==1, 1:7]
seeds.test_z <- seeds_z[ind==2,1:7 ]
seeds.trainLabels_z <- seeds[ind==1, 8]
seeds.testLabels_z <- seeds[ind==2, 8]
```

We got the best results for k=3 in min-max normalization so we will check the same k in z-score normalization:

```r
seeds_test_pred_z_k3 <- knn(train = seeds.train_z, test = seeds.test_z, cl =
                            seeds.trainLabels_z, k=3)
CrossTable(x = seeds.testLabels_z, y = seeds_test_pred_z_k3, prop.chisq=FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  37
##
##
```

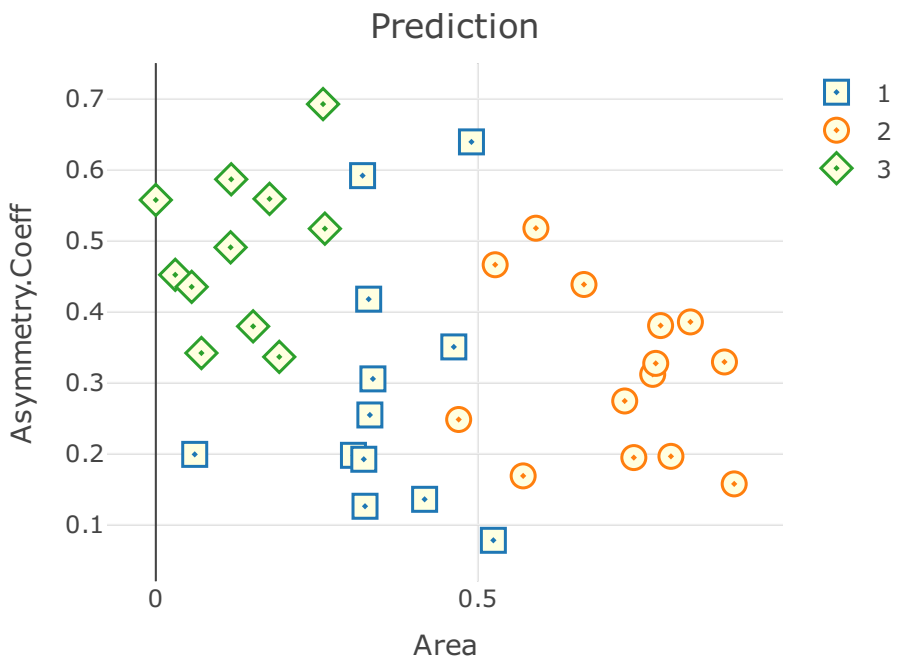
```
##                     | seeds_test_pred_z_k3
## seeds.testLabels_z |         1 |         2 |         3 | Row Total |
## -------------------|-----------|-----------|-----------|-----------|
##                  1 |        13 |         0 |         0 |        13 |
##                    |     1.000 |     0.000 |     0.000 |     0.351 |
##                    |     1.000 |     0.000 |     0.000 |           |
##                    |     0.351 |     0.000 |     0.000 |           |
## -------------------|-----------|-----------|-----------|-----------|
##                  2 |         0 |        13 |         0 |        13 |
##                    |     0.000 |     1.000 |     0.000 |     0.351 |
##                    |     0.000 |     1.000 |     0.000 |           |
##                    |     0.000 |     0.351 |     0.000 |           |
## -------------------|-----------|-----------|-----------|-----------|
##                  3 |         0 |         0 |        11 |        11 |
##                    |     0.000 |     0.000 |     1.000 |     0.297 |
##                    |     0.000 |     0.000 |     1.000 |           |
##                    |     0.000 |     0.000 |     0.297 |           |
## -------------------|-----------|-----------|-----------|-----------|
##       Column Total |        13 |        13 |        11 |        37 |
##                    |     0.351 |     0.351 |     0.297 |           |
## -------------------|-----------|-----------|-----------|-----------|
##
##
```

When we tried to use the z-score normalization further we saw that it led us to over-fitting, so we decided to continue with min-max normalization.

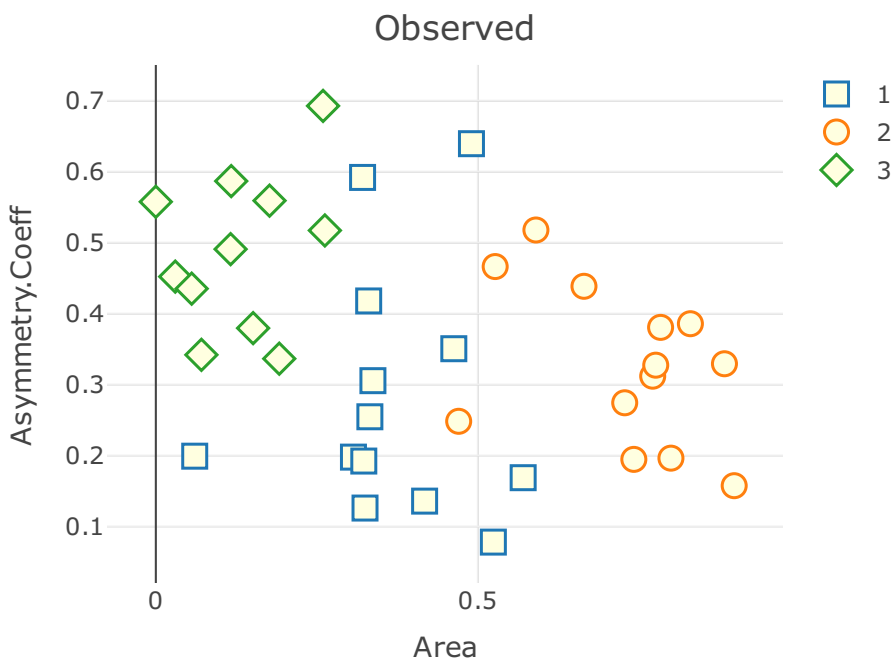
Plots of the classification results by KNN algorithm: We chose the axes to be the 'Area' and 'Asymmetry Coeff' because we saw earlier that these were the features that contributed to the variability the most.

```r
# prediction table
seeds.test_n_pred<-seeds.test_n
seeds.test_n_pred$Type<-seeds_pred_k3

# observed test
seeds.test_n$Type<-seeds.testLabels

# prediction
fig_p <- plot_ly()
fig_p <- fig_p %>% add_trace(data=seeds.test_n_pred, x = ~Area, y = ~Asymmetry.Coeff,
                             symbol = ~Type, split = ~Type, symbols =
                               c('square-dot','circle-dot','diamond-dot'),
               type = 'scatter', mode = 'markers',
               marker = list(size = 12, line = list(width = 1.5), color = 'lightyellow'))%>%
  layout(title="Prediction")

# observed
fig_o <- plot_ly()
fig_o <- fig_o %>% add_trace(data=seeds.test_n, x = ~Area, y = ~Asymmetry.Coeff, symbol =
                               ~Type, split = ~Type, symbols = c('square','circle','diamond'),
               type = 'scatter', mode = 'markers',
               marker = list(size = 12, line = list(width = 1.5), color = 'lightyellow'))%>%
  layout(title="Observed")
```

```
fig_p
```




```
fig_o
```



We can see that the plots are almost identical, which means that the model predicted almost all of the samples correctly.

As we can see, the KNN model gave us good classification results so we recommend to use it further on this data.

Decision Tree:

Using Decision Tree (C5.0) algorithm to predict the seeds types. We used a hyper parameter called "minCases" that determines that 3 is the smallest number of samples that must be put in at least two of the splits. This hyper parameter turned to cause the algorithm to predict the seeds type in the most accurate way. We tried different kinds of hyper parameters, and this one gave us the best results. We used different groups of train and test than the ones we created in the KNN model, because it gave as better results.

```
# sample 800 observations out of the total 1000
train_sample <- sample(199, 150)

# split into train/test
seeds_train <- seeds[train_sample, ]
seeds_test <- seeds[-train_sample, ]

# check that we got about 30% defaulted loans in each dataset:
prop.table(table(seeds_train$Type))
```

```
##
##         1         2         3
## 0.3133333 0.3400000 0.3466667
```

```r
prop.table(table(seeds_test$Type))
```

```
##
##         1         2         3
## 0.3877551 0.3469388 0.2653061
```

Applying the model:

```r
# apply model in training data (8th column is the label to be predicted)
seeds_model <- C5.0(seeds_train[-8], seeds_train$Type, control = C5.0Control(minCases  = 3))

seeds_model
```

```
##
## Call:
## C5.0.default(x = seeds_train[-8], y = seeds_train$Type, control
##  = C5.0Control(minCases = 3))
##
## Classification Tree
## Number of samples: 150
## Number of predictors: 7
##
## Tree size: 6
##
## Non-standard options: attempt to group attributes, minimum number of cases: 3
```

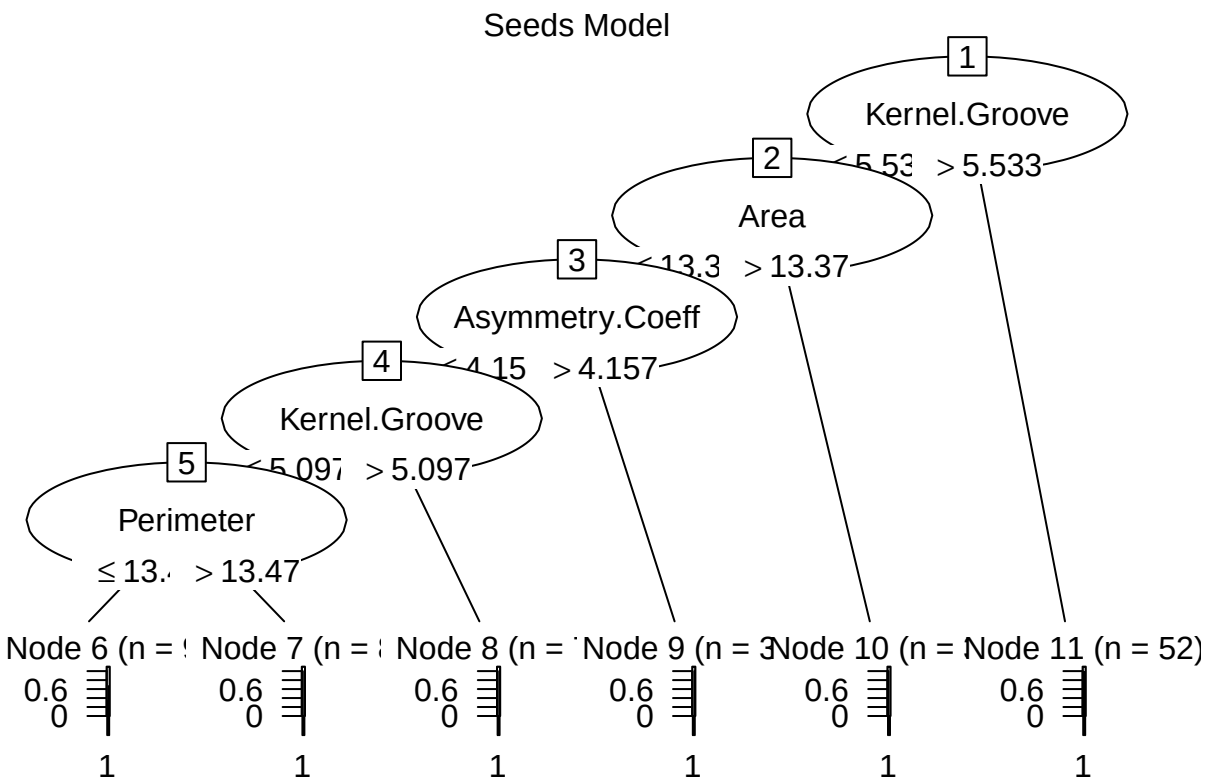
Plot and summary of the model:

```r
summary(seeds_model)
```

```
##
## Call:
## C5.0.default(x = seeds_train[-8], y = seeds_train$Type, control
##  = C5.0Control(minCases = 3))
##
##
## C5.0 [Release 2.07 GPL Edition]      Tue May 24 21:09:50 2022
## -------------------------------
##
## Class specified by attribute ‘outcome’
##
## Read 150 cases (8 attributes) from undefined.data
##
## Decision tree:
##
## Kernel.Groove > 5.533: 2 (52/1)
## Kernel.Groove <= 5.533:
## :...Area > 13.37: 1 (35)
##     Area <= 13.37:
##     :...Asymmetry.Coeff > 4.157: 3 (39)
##         Asymmetry.Coeff <= 4.157:
```

```
##         :...Kernel.Groove > 5.097: 3 (7)
##             Kernel.Groove <= 5.097:
##             :...Perimeter <= 13.47: 3 (9/3)
##                 Perimeter > 13.47: 1 (8)
##
##
## Evaluation on training data (150 cases):
##
##      Decision Tree
##    ----------------
##    Size      Errors
##
##       6    4( 2.7%)   <<
##
##
##     (a)   (b)   (c)    <-classified as
##    ----  ----  ----
##      43     1     3    (a): class 1
##            51          (b): class 2
##                  52    (c): class 3
##
##
##  Attribute usage:
##
##  100.00% Kernel.Groove
##   65.33% Area
##   42.00% Asymmetry.Coeff
##   11.33% Perimeter
##
##
## Time: 0.0 secs
```

```
plot(seeds_model, main = 'Seeds Model')
```



We can see that the model is correct for about 98% of the samples, which means that the model is good. To make sure that there isn't over fitting, we will predict the validation set, and see if the predictions are good for it too.

```
# apply model on test data
seeds_pred <- predict(seeds_model, seeds_test)

CrossTable(seeds_test$Type, seeds_pred, prop.chisq = FALSE, prop.c = FALSE,
           prop.r = FALSE, dnn = c('actual type', 'predicted type'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  49
##
##
##              | predicted type
##  actual type |         1 |         2 |         3 | Row Total |
## -------------|-----------|-----------|-----------|-----------|
##            1 |        18 |         0 |         1 |        19 |
```

```
##              |     0.367 |     0.000 |     0.020 |           |
## -------------|-----------|-----------|-----------|-----------|
##            2 |         1 |        16 |         0 |        17 |
##              |     0.020 |     0.327 |     0.000 |           |
## -------------|-----------|-----------|-----------|-----------|
##            3 |         0 |         0 |        13 |        13 |
##              |     0.000 |     0.000 |     0.265 |           |
## -------------|-----------|-----------|-----------|-----------|
## Column Total |        19 |        16 |        14 |        49 |
## -------------|-----------|-----------|-----------|-----------|
##
##
```

We can see that the prediction was correct for most seeds, so it is possible to say that the model is good for this data-set.

In conclusion, we created here two classification models, one of KNN and one of Decision tree. Both models gave us good results, and we can use both of them to classify our data. We do think that for this data the KNN model gave better results, so this is the model we will prefer to use.