Tutorial 6 Solution

1. (KNN and N-fold Cross Validation)

   Loan managers often need to take into account an applicant's demographic and socio-economic profiles in deciding whether to approve a loan to the applicant, to minimize losses due to defaults. In this exercise we will build and evaluate a classifier based on the German Credit Data to predict whether an applicant is considered as having good or bad credit risk. The features or predictors include (1) loan duration (in months), (2) credit amount, (3) Installment rate in percentage of disposable income and (4) age in years.

   (a) Read and explore the data from the file `German_credit.csv`.

   Answer:

```
> library(class)
> credit = read.csv("C:/Data/German_credit.csv")
> dim(credit)
[1] 1000    5
> head(credit)
  Creditability Duration Amount Instalment Age
1             1       18   1049          4  21
2             1        9   2799          2  36
3             1       12    841          2  23
4             1       12   2122          3  39
5             1       12   2171          4  38
6             1       10   2241          1  48
> set.seed (1)
```

   (b) Standardize the input features.

   Answer:

```
> credit[ ,2:5] <- lapply(credit[ ,2:5] ,scale ) # scaling data from 2nd column to 5th column.
```

   (c) Randomly select 800 customer records to form the training data, and the remaining 200 records will be the test data.

   Answer:

```
> train = sample (1:1000 , 800); #randomly sample a set of 800 indexes in 1:1000
> train.data = credit[train,] # 800 data points for the train set
> test.data = credit[-train,] # 200 data points for the test set
> train.x = train.data[ ,2:5]
> test.x = test.data[ ,2:5]
> train.y = train.data[ ,1]
> test.y = test.data[ ,1]
```

   (d) Use 1-nearest neighbor classifier for the training data to predict if a loan applicant is credible for the 200 test points. Compute the accuracy of the classifier.

   Answer:

```
> knn.pred = knn(train.x, test.x, train.y,k=1)
> confusion.matrix = table(knn.pred , test.y)
> confusion.matrix
```

```
        test.y
knn.pred   0   1
       0  16  42
       1  42 100
```

```
> accuracy = sum(diag(confusion.matrix))/sum(confusion.matrix); accuracy
```

```
[1] 0.58
```

The accuracy of the 1-NN classifier is 0.58.

(e) Use N-folds cross validation with $N = 5$ to find the average accuracy for the 1-nearest neighbor classifier.

Answer:

```
> n_folds=5 # each fold has 50 data points
> folds_j <- sample(rep(1:n_folds, length.out = dim(credit)[1] ))
> table(folds_j)
```

```
folds_j
  1   2   3   4   5
200 200 200 200 200
```

```
> ###############
> X = credit[,2:5] # input features after scaling
> Y = credit[,1] # response
> ###############
> acc=numeric(n_folds) # to store the accuracy for each iteration of n-fold CV
> ###############
> for (j in 1:n_folds) {
+         test_j <- which(folds_j == j) # index of the points in the test set
+         pred <- knn(train=X[ -test_j, ], test=X[test_j, ], cl=Y[-test_j ], k=1)
+
+         acc[j]=mean(Y[test_j] == pred)
+       # this acc[j] = sum(diag(confusion.matrix))/sum(confusion.matrix),
+             #where confusion.matrix=table(Y[test_j],pred)
+       }
> mean(acc)
```

```
[1] 0.624
```

(f) Repeat question 1e for K-nearest neighbor classifiers where $K = 1, 2, ...100$.

Answer:

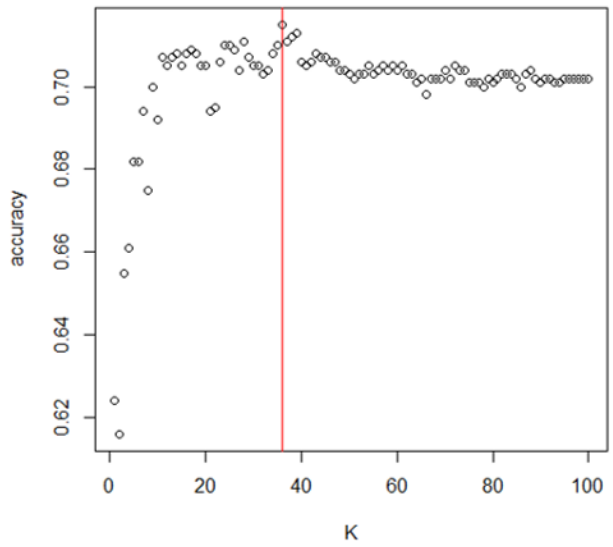
```
> K = 100 # KNN with k = 1,2,...K.
> accuracy=numeric(K) # to store the average accuracy of each k.
> acc=numeric(n_folds) # to store the accuracy for each iteration of n-fold CV
> for (i in 1:K){ # let k runs from 1 to 100
+
+ for (j in 1:n_folds) { # let j runs, from 1 to 5 for the folds
+         test_j <- which(folds_j == j)
+         pred <- knn(train=X[ -test_j, ], test=X[test_j, ], cl=Y[-test_j ], k=i)
+
```

```
+         acc[j]=mean(Y[test_j] == pred)
+       }
+ accuracy[i] = mean(acc)
+ }
> max(accuracy)
[1] 0.715
```

(g) Compare the 100 classifiers above, which few values of K give the best average accuracy?

Answer:

```
> index = which(accuracy == max(accuracy)); index #index = the value of k.
[1] 36
> plot(x=1:K, accuracy)
> abline(v = index, col = "red")
```



2. (Decision Trees)

Consider the famous Iris Flower Data set which was first introduced in 1936 by the famous statistician Ronald Fisher. This data set consists of 50 observations from each of three species of Iris (Iris setosa, Iris virginica and Iris versicolor).

Four features were measured from each observation: the length and the width of the sepals and petals (in cm).

(a) Use decision tree method to predict Iris species based on all four features.

(b) Visualize the decision tree above, using the `rpart.plot` function.

Answer:

```
> library("rpart")
> library("rpart.plot")
> iris = read.csv("C:/Data/iris.csv")
> head(iris)
```

Source: `http://suruchifialoke.com`

```
  sepal.length sepal.width petal.length petal.width     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
```

```
> fit.iris <- rpart(species ~ sepal.length + sepal.width + petal.length + petal.width,
+ method = "class", data =iris, control = rpart.control( minsplit =1),
+ parms = list( split =' information '))
> rpart.plot(fit.iris , type =4, extra =2, clip.right.labs =FALSE , varlen =0, faclen =0)
```

The fitted tree is given in the figure below.

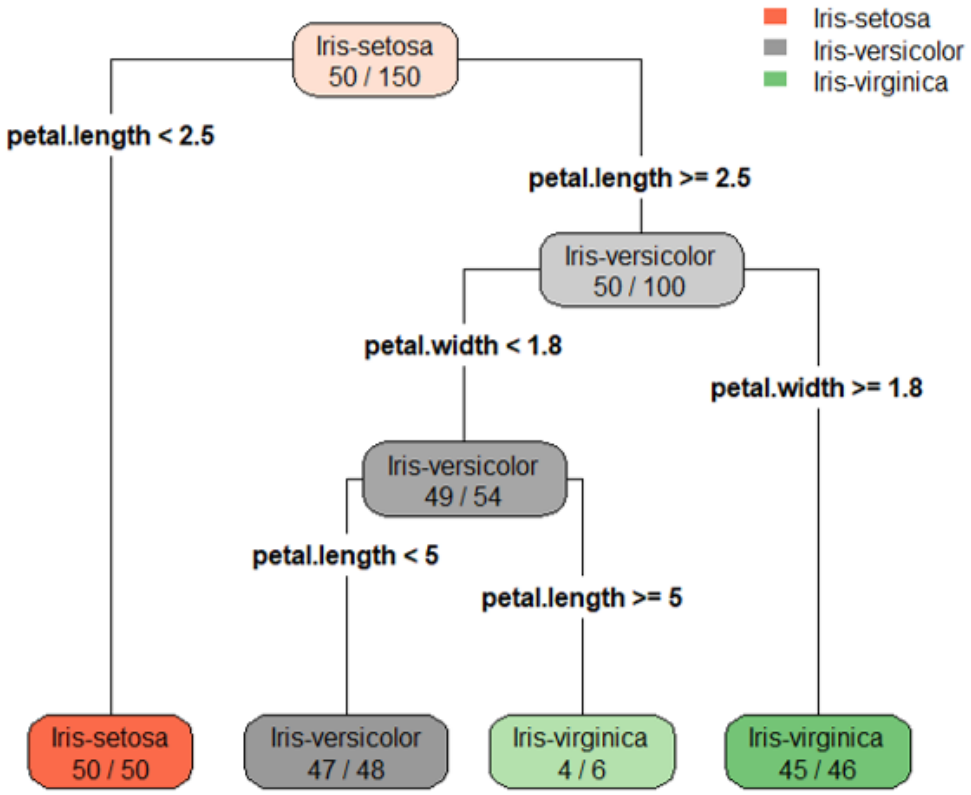
If the measurement of petal length is less than 2.5 cm then the flower is of Iris-setosa.

If the petal length is ≥ 2.5 cm with the petal width is ≥ 1.8 cm then high chance (45/46) it will be an Iris-virginica.

If the petal length is ≥ 2.5 cm with the petal width is < 1.8 cm then we continue to check if the petal length is in the interval $[2.5, 5)]$. If yes, then high chance (47/48) it is Iris-versicolor.

(c) What are the more important features in the fitted tree above?

Answer: It seems the sepal length and sepal width are not important in the classification while the petal length and petal width are more important.


Iris-setosa
Iris-versicolor
Iris-virginica
Iris-setosa
50 / 150
petal.length < 2.5
petal.length >= 2.5
Iris-versicolor
50 / 100
petal.width < 1.8
petal.width >= 1.8
Iris-versicolor
49 / 54
petal.length < 5
petal.length >= 5
Iris-setosa
50 / 50
Iris-versicolor
47 / 48
Iris-virginica
4 / 6
Iris-virginica
45 / 46