CS3002: Introduction to Classification - ASSESSED

This worksheet must be assessed by a GTA. All assessed exercises must be completed and checked by a GTA in the lab. This is PASS / FAIL and all assessed sheets must be passed in order to pass the coursework for this module. You can make multiple attempts but do not ask to be assessed until you are ready.

In this lab we will:

- Experiment with Decision trees
- and K Nearest Neighbour

Datasets:

- Wine Data

Functions:

- `rpart, printcp, predict, prune`
- `knn`
- `table, plot`

Download wine.data from blackboard and store in your home directory.
These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines. The units are not important for the purposes of this problem as I recommend using the scaled data in your analysis anyway.

The class represents the cultivar/wine type (1,2, or 3)

The 13 attributes are:

1) Alcohol
2) Malic acid
3) Ash
4) Alcalinity of ash
5) Magnesium
6) Total phenols
7) Flavanoids
8) Nonflavanoid phenols
9) Proanthocyanins
10)Color intensity
11)Hue
12)OD280/OD315 of diluted wines
13)Proline

Now read it into R:

```
winedata = read.csv('J:\\Data\\Teaching\\2019-
2020\\CS3002\\NEWLAB_SCRIPT\\Data\\winedata.csv', sep=",")
```

The first column represents the three different classes and columns 2-14 represent the different attributes. Let's now separate the class and values into separate variables:

```
wineclass = winedata[,1]
winevalues = winedata[,-1]
```

Now let's build a training set of the first 100 values (rows) and test set of the remaining (101 – 178) values.

```
#set up a training set
wineclassTrain = wineclass[1:100]
winevaluesTrain = winevalues[1:100,]

#and testset
wineclassTest = wineclass[100:178]
winevaluesTest = winevalues[100:178,]
```

Decision trees

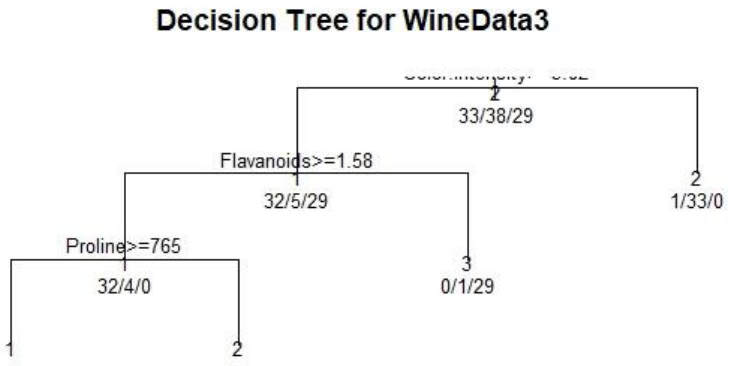
Let's start by using the same 3-class data in `winedata3.csv` to build a decision tree with the command `rpart`:

```
install.packages("rpart")
library(rpart)
fit <- rpart(wineclassTrain~., method="class", data=winevaluesTrain)
```

(the `wineclass~.` represents a formula to predict wineclass using all other variables)

Let's see what the decision tree looks like using a plot:

```
plot(fit, uniform=TRUE, main="Decision Tree for WineData3")
text(fit, use.n=TRUE, all=TRUE, cex=.8)
```



Now we are going to test the classifier on the test set by calculating the predictions for each testcase in our test set:

```
treepred <-predict(fit, winevaluesTest, type = 'class')
```

and then comparing to the actual test class values to get the accuracy:

```
n = length(wineclassTest) #the number of test cases
ncorrect = sum(treepred==wineclassTest) #the number of correctly
predicted
accuracy=ncorrect/n
print(accuracy)
```

We can also view the results as a table known as a confusion matrix:

```
table_mat = table(wineclassTest, treepred)
print(table_mat)
```

We can also see what effect pruning has (with a pruning parameter here, cp, set to 0.1):

```
pfit<- prune(fit, cp=0.1)
plot(pfit, uniform=TRUE, main="Pruned Decision Tree for WineData3")
text(pfit, use.n=TRUE, all=TRUE, cex=.8)
```

K Nearest Neighbour (KNN)

We can classify using the KNN method using `knn (here with k = 3):`

First we have to install the class library

```
library(class)
```

and then generate our predicted classes:

```
knn3pred =  knn(winevaluesTrain, winevaluesTest, wineclassTrain, k=3)
```

and finally calculating the accuracy as before:

```
n = length(wineclassTest) #the number of test cases
ncorrect = sum(knn3pred==wineclassTest) #the number of correctly
predicted
accuracy=ncorrect/n
print(accuracy)
```

ASSESSED EXERCISE:

Read in "`seeds_dataset.csv`" from last week's clustering lab

Write a script to do the following:

1. Learn a decision tree for the 3 classes of wheat kernel. Remember to save some data for testing. You will also need to mix up the ordering of the rows – use `seeds_rand=seeds[sample(150,150),]` to do this. Why do you think that this is necessary?

2. Test it by scoring the accuracy on test data using different degrees of pruning using `prune`.

3. Scatterplot 2 selected variables of the data (and colour code according to the decision tree output), display the learnt tree and calculate the accuracy.

4. Compare the accuracy of the different pruned trees to KNN with different values of *k*

SAMPLE EXAM QUESTIONS:

Q1) name one advantage and one disadvantage of using the KNN classifier

Q2) Given two classifiers learnt from a training set and tested on unseen data to give the following two confusion matrices:

Classifier 1:

	T	F
T	12	8
F	15	5

Classifier 2:

	T	F
T	17	3
F	4	16

a) Which classifier is the most accurate

b) Calculate the Sensitivity and Specificity of the two classifiers

Q3) Briefly compare two classifiers stating one advantage that each has over the other.

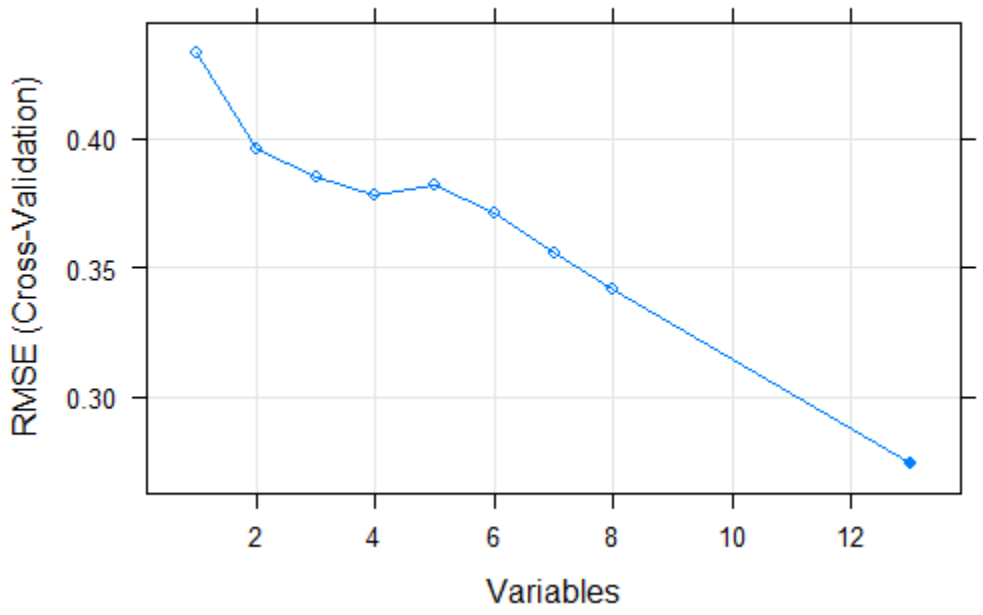
EXTRA:

i) Explore Random Forests for classifying data and selecting features (note this performs cross validation so we do not need to split into training and testing):

```
install.packages("randomForest")
library(randomForest)
fit <- randomForest(wineclass~.,   data=winevalues)
print(fit) # view results
importance(fit) # importance of each predictor
```

ii) Explore `rfecontrol` to perform recursive feature selection with entropy as the criteria to find the top 30 features:

```
install.packages("caret")
library(caret)
# define the control using a linear selection function (lmFuncs)
control <- rfeControl(functions=lmFuncs, method="cv", number=10)
# run the RFE algorithm
results <- rfe(winevalues, wineclass, sizes=c(1:8), rfeControl=control)
# summarize the results
print(results)
# list the chosen features
predictors(results)
# plot the results
plot(results, type=c("g", "o"))
```



Exercise:

Use `rfecontrol` and `knn`, to see which criterion finds the best features for classifying the prostate data. Remember, to use a train and test set (or cross validation) so that feature selection and classifier learning is only applied to the training set. Try experimenting with different functions for the feature selection.