PRACTICAL 9

Aim:

The aim of this practical is to illustrate how classification models can be evaluated in R.

Before you start

Load the `tidyverse`, `rpart` and `caret` packages.

Instructions:

This document contains discussions of some basic concepts, and instructions which you should execute in RStudio. **Text in bold** indicate actions that are to be taken within RStudio. `Text highlighted in grey` shows the code you should enter in RStudio.

Due date:

For Tutorial 9, it will be assumed that you have mastered the basic concepts discussed here.

Preliminaries

1. We will again be using the `iris` dataset, which was also used in Tutorials 2 and 3. This dataset is already available in R / RStudio.

Examine the first few rows of the data, as well as the internal structure of the dataset. Also obtain summary statistics for the dataset.

```
head(iris)
str(iris)
summary(iris)
```

2. We would like to focus on a binary classification problem in this practical. However, the target variable `Species` in the `iris` dataset contains 3 categories: Virginica, Versicolor and Setosa. There are also 4 attributes in the dataset, namely sepal width, sepal length, petal width and petal length. For easy visualisation, we also want to limit ourselves to only 2 attributes in this practical.

Remove the `setosa` species from the dataset, and also only select the width attributes (i.e. `Sepal.Width` and `Petal.Width`). Save the resulting subset in an object named `myiris`.

You should also make sure that the target variable `Species` now only consists of 2 factor levels.

```
myiris <- iris %>% filter(Species != "setosa") %>%
select(Sepal.Width, Petal.Width, Species)
myiris$Species <- factor(myiris$Species)
str(myiris)
```

3. **Plot the `myiris` data in a scatterplot. `Sepal.Width` should go on the *x*-axis, while `Petal.Width` should be on the *y*-axis. The different `Species` should be indicated by different coloured points on the scatterplot.**

```
ggplot(data = myiris, mapping = aes(x = Sepal.Width, y =
Petal.Width, color = Species)) + geom_point()
```

Remember that overplotting can be avoided by using `position = "jitter"` in a scatterplot:

```
ggplot(data = myiris, mapping = aes(x = Sepal.Width, y =
Petal.Width,  color  =  Species))  +  geom_point(position  =
"jitter")
```

Fitting classifier models

4. The first classifier considered will be a decision tree. We will use the `rpart()` function to fit the tree.

4.1 **Fit a decision tree to the `myiris` data to predict the species of an iris based on its sepal width and petal width. Save your model in an object named `dectree`.**

```
dectree <- rpart(Species~., data = myiris)
```

4.2 **Obtain the predicted values for the decision tree model fit in Step 4.1. Store these predicted values in an object named `dectree.pred`.**

```
dectree.pred  <-  predict(dectree,  myiris[,-3],  type  =
"class")
```

4.3 **Get the confusion matrix for the `dectree` model. Store this confusion matrix in an object named `dectree.conf`.**

```
dectree.conf <- confusionMatrix(dectree.pred,
myiris$Species)
dectree.conf
```

Take note of the following important values from the output from the `confusionMatrix()` function:

- Accuracy
- No Information Rate
- Sensitivity
- Specificity
- Pos Pred Value
- Neg Pred Value
- ‘Positive’ Class

The other measures are not covered in this module.

5. Now we will fit a one-nearest neighbour (1NN) classifier to the dataset.

5.1 **Fit a 1NN model to the `myiris` data to predict the species of an iris based on its sepal width and petal width. Use the `knn3()` function. Save your model in an object named `onenn`.**

```
onenn <- knn3(Species~., data = myiris, k = 1)
```

5.2 **Obtain the predicted values for the 1NN model fit in Step 5.1. Store these predicted values in an object named `onenn.pred`.**

```
onenn.pred <- predict(onenn, myiris[,-3], type = "class")
```

5.3 **Get the confusion matrix for the `onenn` model. Store this confusion matrix in an object named `onenn.conf`.**

```
onenn.conf <- confusionMatrix(onenn.pred,
myiris$Species)
onenn.conf
```

6. Lastly we will fit a five-nearest neighbour (5NN) classifier to the dataset.

6.1 **Fit a 5NN model to the `myiris` data to predict the species of an iris based on its sepal width and petal width. Use the `knn3()` function. Save your model in an object named `fivenn`.**

```
fivenn <- knn3(Species~., data = myiris, k = 5)
```

6.2 **Obtain the predicted values for the 5NN model fit in Step 6.1. Store these predicted values in an object named `fivenn.pred`.**

```
fivenn.pred  <-  predict(fivenn,  myiris[,-3],  type  =
"class")
```

6.3 **Get the confusion matrix for the `fivenn` model. Store this confusion matrix in an object named `fivenn.conf`.**

```
fivenn.conf <- confusionMatrix(fivenn.pred,
myiris$Species)
fivenn.conf
```

Comparison of models

Comparing the confusion matrices for the 3 models that were fit show the following:

- The 1NN model performs best on the training data. However, recall that when evaluating model accuracy on the training data (which is what we are doing here), we expect the 1NN model to perform very well!
- The decision tree and 5NN model have the same accuracy. However, the sensitivity and specificity differ. For this particular problem, it probably is not sensible to think about preferring one type of error above the other. Keep in mind however that for many other problems, this can be an important consideration.
- Remember that – as discussed when kNN models were covered a few weeks ago – the performance of models should not be evaluated on the training data. In this practical, we have done so, simply to practice the calculation of different evaluation measures. However, to get a true indication of the performance of a model, we should investigate the performance on previously unseen data, i.e. a test data set. We will consider this next week.

You should ensure that you are able to manually calculate the evaluation measures mentioned in Step 4.3 above, using only the confusion matrix. A document showing the manual calculations for the decision tree will be made available in due course; you should consider the equivalent calculations for the 1NN and 5NN models on your own as revision / preparation for the assessment.