Multinomial logistic regression and classification trees for classification of satellite terrain data

Overview

The dataset Satellite available in package mlbench contains 6435 images displaying different scenes recorded by the Landsat satellite program. Each row of the data correspond to an image, which encodes a scene in terms of 36 features extracted from multi-spectral values of pixels in 3x3 neighbourhoods of the image. According to the central pixel in each neighbourhood, images are classified into 6 type of scenery: cotton_crop, damp_grey_soil, grey_soil, red_soil, soil_with_vegetation_stubble, very_damp_grey_soil. The aim is to predict the classification of a satellite image, given the multi-spectral values.

Preparation

```r
#necessary packages
library(nnet)
library(partykit)
library(rpart)
library(mlbench)
library(knitr)
data("Satellite")

Satellite$classes <- gsub(" ", "_", Satellite$classes)
Satellite$classes <- factor( as.character(Satellite$classes) )

# setting seed to have the same initial
# split of data into training and validation sets
#to ensure consistencey across replications
set.seed(22021)
N <- nrow(Satellite)
keep <- sample(1:N, 5000)
test <- setdiff(1:N, keep)
dat <- Satellite[keep,]
dat_test <- Satellite[test,]
```

The data matrix dat will be employed for training and model comparison (validation), while the data matrix dat_test will be employed for testing.

Cross Validation

The training and validation set has 5000 observations I will use a 5 fold cross validation procedure when validating the models as it leaves each fold with 1000 observations which is an easy round number and not too computationally intensive, leave one out cross validation is not viable here as it would be too computatoinally expensive. The cross validation returns the table below which contains the validation accuracy for each iteration of the 5 fold cross validation.

```r
K<- 5
N <- nrow(dat)

folds <- rep( 1:K, ceiling(N/K) )
folds <- sample(folds)             # random permute
folds <- folds[1:N]                # ensure we got N data points


out<- matrix(nrow = K , ncol = 2)
for ( k in 1:K ) {
  train <- which(folds != k)
  test <- setdiff(1:N, train)

  # fit classificatoin tree
  fit <- rpart(classes ~ ., data = dat, subset = train)
  #fitting logistic regression
  regression <- multinom(classes~., data = dat, subset = train, maxit =1000)

  # classify the test data observations for classification tree
  predTest_tree <- predict(fit, type = "class", newdata = dat[test,])
  tabTest_tree <- table(dat$classes[test], predTest_tree)
  out[k,1] <- sum(diag(tabTest_tree))/sum(tabTest_tree)

  #classify the test data for logistic regression
  predTest_reg <- predict(regression, type = "class", newdata = dat[test,])
  tabTest_reg <- table(dat$classes[test], predTest_reg)
  out[k,2] <- sum(diag(tabTest_reg))/sum(tabTest_reg)

}

out
summary(out)
Results = data.frame(out)
colnames(Results) = c('Classification Tree' , 'Logistic Regression')
```

The table below shows the results of each of the folds. The results show the accuracy of a model fit on the training data and then applied to the validation set, we can see that on all 5 folds the logistic regression model has a greater overall accuracy.

Classification Tree	Logistic Regression
0.823	0.849
0.840	0.875
0.792	0.874
0.788	0.858
0.805	0.848

The logistic regrassion model is the clear winner here. It's classification accuracy is higher and the results are more consistent. The range of accuracy for the classification tree has a range of 0.052 while the Logistic Regression has a range of values of 0.027

Logistic Regression in depth analysis

So we will take the logistic regression forward to the test data to evaluate it's performance.

```
#--------------------#Testing on testing data----------------------------------------
Test_fit <-predict(regression, type = 'class' , newdata = dat_test)
tabTest <- table(dat_test$classes , Test_fit)
accuracy <- sum(diag(tabTest))/sum(tabTest)
accuracy
```

An accuracy of 0.8390244 is slightly worse than the average validation accuracy across the 5 folds, but only marginally. This similarity in performance indicates that the model is good at accurately classifying new unseen data and is simply just over-fit to the training data. If the training accuracy had been much greater than the test accuracy it would indicate that the model was over-fit. This doesn't appear to be the case and the logistic regression has good performance at classifying this data set.

Below is a table of the true classes vs what the model has classified them as. The columns represent the true values of the observations and the rows represent what the logistic regression classifier has classified them as. Any value on the diagonal represents a correct classification.

```
kable(tabTest)
```

	cotton_crop	damp_grey_soil	grey_soil	red_soil	vegetation_stubble	very_damp_grey_soil
cotton_crop	141	0	0	0	14	1
damp_grey_soil	1	57	24	0	2	62
grey_soil	0	19	280	1	0	1
red_soil	1	1	6	338	11	0
vegetation_stubble	12	0	1	6	112	22
very_damp_grey_soil	0	31	7	0	8	276

Take the column cotton_crop, the model correctly classified 141 observations and incorrectly classified 14 observations. Of those 14, 1 was classified as damp grey soil, 1 as red soil, and 12 as vegetation stubble. So it's clear most of the innacurate classifications of cotton crops is as vegetation stubble, this can be thought of as the model is having difficulty differentiating between the two.

Using this approach we can see that the majority of the innaccuracy comes from the classification of very damp grey soil, more specifically differentiating between damp grey soil and very damp grey soil. So if the model told us that an observation is very damp grey soil we would know there is only an approximately 0.7624309% chance that the classification is actually correct, so for this category we could hand classification to another model or manual intervention might be neccessary depending on what an acceptable level of accuracy is.