Statistical Machine Learning

Abstract

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.

We would use logistic regression and random forest to predict the classification of images. Followed by their comparison and performance of the best model on the data.

Theory

Random forests are a very powerful supervised classification method. A random forest is simply a collection of classification trees estimated on random subsets of data.

Multinomial logistic regression is an extension of logistic regression to accommodate categorical target variable with more than two outcomes.

Methodology and Observations

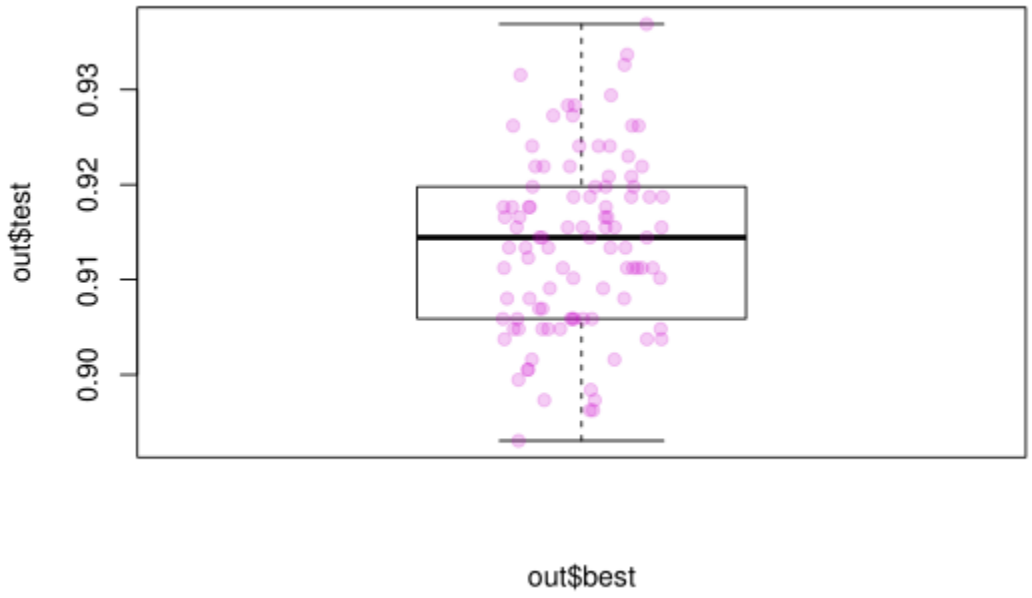
- Load the satellite data set, followed by removing spaces between the final class followed by factoring the data.
- Then we set a seed so that we can reproduce the exact result.
- We then divide the data into keep and test, where keep would contain training and validation set whereas the model would be tested on a relatively new test set.
- We then fit both the models to check their base performance on the image data, we get the following result.

```
                  predValCt
                   cotton_crop damp_grey_soil grey_soil red_soil vegetation_stubble very_damp_grey_soil
cotton_crop                 96              0         0        0                 0                    0
damp_grey_soil               0             86         9        0                 0                    6
grey_soil                    0              0       174        0                 0                    0
red_soil                     0              0         1      230                 1                    0
vegetation_stubble           0              0         0        2               110                    2
very_damp_grey_soil          0              1         0        0                 1                  216
                  predValLog
                   cotton_crop damp_grey_soil grey_soil red_soil vegetation_stubble very_damp_grey_soil
cotton_crop                 92              0         0        0                 4                    0
damp_grey_soil               1             53        20        1                 1                   25
grey_soil                    0              6       168        0                 0                    0
red_soil                     0              0         1      229                 2                    0
vegetation_stubble           2              1         0        3                97                   11
very_damp_grey_soil          0             14         5        0                 7                  192
```

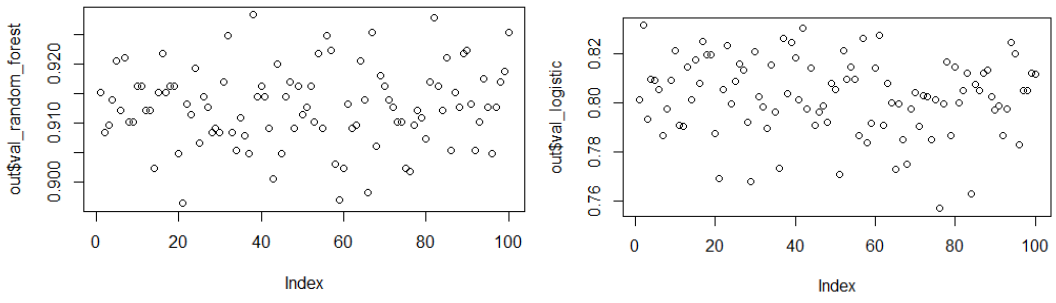
```
randomforest  multinomial
   0.9112299    0.8534759
```

- We can see that random forest performs better than multinomial in our initial analysis.

- We can also see that random forest classifies each and every class more efficiently than multinomial regression, with least difference in red_soil highest misclassification in damp_grey_soil .
- We then perform a best analysis on our test data and random forest outperforms multinomial in case of this image classification.
- We repeat by reshuffling our data 100 times and dividing our original training data into training(70%) and validation set(30%), we re-train our model again and store the results of accuracy for all 100 reshuffles.
- This is followed by storing the best model for the data which comes out to be random forest each time.



- On the validation set, the multinomial regression performs with a mean of 80% accuracy and a variance of 0.0002, the random forest algorithm performs with a mean accuracy of 91.28%.



Conclusion:

The best classifier performs with a mean value of around 96.55% on the test set with 50% of the results within 96.23% and 96.82%. It is safe to say that the random forest algorithm outperforms multinomial regression for this particular image classification.

Appendix

---

title: "SML Assignment 3"

author: "Shubhang Periwal 19201104"

date: "4/18/2020"

output: pdf_document

```{r}
library(mlbench)
data("Satellite")
# this will re-order alphabetically class labels and remove spacing
Satellite$classes <- gsub(" ", "_", Satellite$classes)
Satellite$classes <- factor( as.character(Satellite$classes) )
# to have the same initial split
set.seed(777222)
D <- nrow(Satellite)
keep <- sample(1:D, 5500)
test <- setdiff(1:D, keep)
dat <- Satellite[keep,]
dat_test <- Satellite[test,]
```

Multinomial Logistic Regression

```{r}

library(nnet)

library(randomForest)

fitLog <- multinom(classes ~ .,data = dat,maxit=300, trace=FALSE) #multi with maximum number of iteration set to 300

fitCt <- randomForest(classes ~ ., data = dat,maxit=300, trace=FALSE) # randomforest

```

```{r}

#  Random Forest

predValCt <- predict(fitCt, type = "class", newdata = dat_test)

tabValCt <- table(dat_test$classes, predValCt)

tabValCt

accCt <- sum(diag(tabValCt))/sum(tabValCt)

#

# Multinomial Regression

predValLog <- predict(fitLog, type = "class", newdata = dat_test)

tabValLog <- table(dat_test$classes, predValLog)

tabValLog

accLog <- sum(diag(tabValLog))/sum(tabValLog)

```

```{r}
```

```
# print accuracy
acc <- c(random_forest = accCt, multinomial = accLog)
acc
```

```{r}
# use the method that did best on the validation data
# to predict the test data
best <- names( which.max(acc) )
switch(best,
       random_forest = {
         predTestCt <- predict(fitCt, type = "class", newdata = dat_test)
         tabTestCt <- table(dat_test$classes, predTestCt)
         accBest <- sum(diag(tabTestCt))/sum(tabTestCt)
       },
       multinomial = {
         predTestLog <- predict(fitLog, type = "class", newdata = dat_test)
         tabTestLog <- table(dat_test$classes, predTestLog)
         accBest <- sum(diag(tabTestLog))/sum(tabTestLog)
       }
)
best
accBest

```

```{r}
# replicate the process a number of times
R <- 100
out <- matrix(NA, R, 4)
colnames(out) <- c("val_random_forest", "val_logistic", "best", "test")
out <- as.data.frame(out)

for ( r in 1:R ) {

  # split the data
  keep <- sample(1:D, 5500)
  test <- setdiff(1:D, keep)

  dat_test <- as.data.frame(Satellite[test,])



  train <- sample(keep, size = 0.7*5500)                  # 70% of data points are used as training data
  val <- sample( setdiff(keep, train) )     # 30% of data points are used as validation data
  dat <- as.data.frame(Satellite[train,])
  dat_val <- as.data.frame(Satellite[val,])

  # fit classifiers to only the training data
  fitCt <- randomForest(classes ~ ., data = dat,trace=FALSE)        # Random Forest
  fitLog <- multinom(classes ~ ., data =  dat,trace=FALSE)    # multinomial logistic regression

  # classify the validation data observations
  predValCt <- predict(fitCt, type = "class", newdata = dat_val)    # Random forest
  tabValCt <- table(dat_val$classes, predValCt)
```

```
tabValCt
accCt <- sum(diag(tabValCt))/sum(tabValCt)
#
predValLog <- predict(fitLog, type = "class", newdata = dat_val)  # logistic regression
tabValLog <- table(dat_val$classes, predValLog)
tabValLog
accLog <- sum(diag(tabValLog))/sum(tabValLog)

# accuracy
acc <- c(random_Forest = accCt, multinomial = accLog)
out[r,1] <- accCt
out[r,2] <- accLog



# use the method that did best on the validation data
# to predict the test data
best <- names( which.max(acc) )
switch(best,
       random_Forest = {
         predTestCt <- predict(fitCt, type = "class", newdata = dat_test)
         tabTestCt <- table(dat_test$classes, predTestCt)
         accBest <- sum(diag(tabTestCt))/sum(tabTestCt)
       },
       multinomial = {
         predTestLog <- predict(fitLog, type = "class", newdata = dat_test)
         tabTestLog <- table(dat_test$classes, predTestLog)
         accBest <- sum(diag(tabTestLog))/sum(tabTestLog)
       }
```

```
  )
  out[r,3] <- best
  out[r,4] <- accBest

}


```


```{r}
# check out the error rate summary statistics
table(out[,3])
tapply(out[,4], out[,3], summary)
boxplot(out$test ~ out$best)
stripchart(out$test ~ out$best, add = TRUE, vertical = TRUE,
           method = "jitter", pch = 19, col = adjustcolor("magenta3", 0.2))
mean(out$val_random_forest)
plot(out$val_random_forest)
mean(out$val_logistic)
plot(out$val_logistic)
var(out$val_random_forest)
var(out$val_logistic)
```