Multinomial Logistic Regression

Zhen Trinh

Read in data

```
subjects.new <- read.csv("Data/subjects.new.csv")
train <- read.csv("Data/train.csv")
test <- read.csv("Data/test.csv")
validation <- read.csv("Data/validation.csv")
```

We will use the multinom() function from the nnet package to estimate multinomial logistic regression model because it does not require the data to be reshaped (as the mlogit package does).

Fit the model

```
library(nnet)
model.lr <- multinom(Activity ~., data = train)
```

Check model performance

```
# Apply the model on validation dataset
pred.lr <- predict(model.lr, validation)

# Load the caret package
library(caret)
library(e1071)

# Create a confusion matrix comparing the predicted and true activity types
confusionMatrix(pred.lr, validation$Activity)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  L1 L10 L11 L12  L2  L3  L4  L5  L6  L7  L8  L9
       L1  620   0   0   0   0   0   0   4   0   0   0   0
       L10   0 456 141 110   0   0   0   0   0   0   0   0
       L11   0 106 447  46   0   0   0   1   0   0   0   0
       L12   0  36  13  48   0   0   0   0   0   0   0   0
       L2    0   0   0   0 596   0   4  23   0   0   0   0
       L3    0   0   0   0   0 634   0   0   0   0   0   0
       L4    0   0   2   0   0   0 533 110   2   2   0   0
       L5    0   0   1   0   0   0  65 350   8  29  24   0
       L6    0   3   0   0   0   0   1  12 471  17  69   0
       L7    0   0   0   0   0   0   0  29  29 593  20   0
       L8    0   1   0   1   0   0   2  41  29   7 566   0
       L9    0   0   0   0   0   0   0   0   0   0   0 640
```

```
Overall Statistics

               Accuracy : 0.8577
                 95% CI : (0.8492, 0.8658)
    No Information Rate : 0.0978
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.844
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: L1 Class: L10 Class: L11 Class: L12 Class: L2
Sensitivity            1.00000    0.75748    0.74007   0.234146   1.00000
Specificity            0.99937    0.96041    0.97586   0.992727   0.99575
Pos Pred Value         0.99359    0.64498    0.74500   0.494845   0.95666
Neg Pred Value         1.00000    0.97658    0.97524   0.977064   1.00000
Prevalence             0.08931    0.08672    0.08701   0.029530   0.08585
Detection Rate         0.08931    0.06569    0.06439   0.006914   0.08585
Detection Prevalence   0.08989    0.10184    0.08643   0.013973   0.08974
Balanced Accuracy      0.99968    0.85894    0.85796   0.613437   0.99787
                     Class: L3 Class: L4 Class: L5 Class: L6 Class: L7
Sensitivity            1.00000   0.88099   0.61404   0.87384   0.91512
Specificity            1.00000   0.98169   0.98007   0.98407   0.98761
Pos Pred Value         1.00000   0.82126   0.73375   0.82199   0.88376
Neg Pred Value         1.00000   0.98856   0.96597   0.98932   0.99123
Prevalence             0.09133   0.08715   0.08211   0.07764   0.09334
Detection Rate         0.09133   0.07678   0.05042   0.06785   0.08542
Detection Prevalence   0.09133   0.09349   0.06871   0.08254   0.09666
Balanced Accuracy      1.00000   0.93134   0.79705   0.92896   0.95137
                     Class: L8 Class: L9
Sensitivity            0.83358   1.00000
Specificity            0.98707   1.00000
Pos Pred Value         0.87481   1.00000
Neg Pred Value         0.98205   1.00000
Prevalence             0.09781   0.09219
Detection Rate         0.08153   0.09219
Detection Prevalence   0.09320   0.09219
Balanced Accuracy      0.91032   1.00000
```

The 95% prediction interval for the model is (84.92%, 86.58%), we can tune the parameters to get a higher accuracy result.

10-fold cross validation

```r
# Set train control using 10-fold cross validation
ctrl <- trainControl(method = "cv", number = 10, savePredictions = TRUE)

# set seed to obtrain reproducible result
set.seed(7)

# Set up tuning parameters for multinomial logistic regression model
```

```r
m.lr <- train(Activity ~., data = rbind(train,validation), method = 'multinom',
              trControl = ctrl, tuneLength = 5)
```

```r
# Examine the result of 10-fold cross validation
m.lr
```

```
Penalized Multinomial Regression

28195 samples
    8 predictor
   12 classes: 'L1', 'L10', 'L11', 'L12', 'L2', 'L3', 'L4', 'L5', 'L6', 'L7', 'L8', 'L9'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 25377, 25374, 25375, 25376, 25376, 25376, ...
Resampling results across tuning parameters:

  decay  Accuracy   Kappa
  0e+00  0.8541591  0.8402171
  1e-04  0.8551169  0.8412656
  1e-03  0.8549394  0.8410734
  1e-02  0.8549747  0.8411051
  1e-01  0.8540526  0.8400996

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was decay = 1e-04.
```

As the footnote describes, the model with the largest accuracy was selected. This was the model that used a penalized multinomial regression with decay = 0. However, the accuracy did not improve much as compared to the previous model.

Apply tuned model on unseen test data

```r
# Make prediction
p.lr <- predict(m.lr, test)
confusionMatrix(p.lr, test$Activity)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  L1 L10 L11 L12  L2  L3  L4  L5  L6  L7  L8  L9
       L1  611   0   0   0   0   0   1   4   0   0   0   0
       L10   0 496 108  85   0   0   0   1   0   0   0   0
       L11   0 106 468  44   0   0   0   0   0   0   0   0
       L12   0  21  17  48   0   0   0   0   0   0   0   0
       L2    0   0   0   0 611   0   6  19   0   0   0   0
       L3    0   0   0   0   0 568   0   0   0   0   0   0
       L4    0   0   1   0   0   0 553 107   1   0   0   0
       L5    0   0   1   0   0   0  59 368  14  20  13   0
       L6    0   1   0   0   0   0   0  20 545  15  80   0
       L7    0   2   0   0   0   0   1  34  35 575  24   0
       L8    0   0   0   2   0   0   1  64  40   3 529   0
       L9    0   2   0   1   0   0   0   0   0   0   0 654
```

```
Overall Statistics

               Accuracy : 0.8634
                 95% CI : (0.8552, 0.8714)
    No Information Rate : 0.0937
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8503
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: L1 Class: L10 Class: L11 Class: L12 Class: L2
Sensitivity            1.00000    0.78981    0.78655   0.266667   1.00000
Specificity            0.99921    0.96945    0.97650   0.994411   0.99607
Pos Pred Value         0.99188    0.71884    0.75728   0.558140   0.96069
Neg Pred Value         1.00000    0.97901    0.98003   0.980850   1.00000
Prevalence             0.08755    0.08998    0.08526   0.025792   0.08755
Detection Rate         0.08755    0.07107    0.06706   0.006878   0.08755
Detection Prevalence   0.08826    0.09887    0.08855   0.012323   0.09113
Balanced Accuracy      0.99961    0.87963    0.88153   0.630539   0.99804
                     Class: L3 Class: L4 Class: L5 Class: L6 Class: L7
Sensitivity            1.00000   0.89050   0.59643   0.85827   0.93801
Specificity            1.00000   0.98286   0.98318   0.98172   0.98492
Pos Pred Value         1.00000   0.83535   0.77474   0.82451   0.85693
Neg Pred Value         1.00000   0.98924   0.96172   0.98575   0.99398
Prevalence             0.08139   0.08898   0.08841   0.09099   0.08783
Detection Rate         0.08139   0.07924   0.05273   0.07809   0.08239
Detection Prevalence   0.08139   0.09486   0.06806   0.09471   0.09615
Balanced Accuracy      1.00000   0.93668   0.78981   0.91999   0.96146
                     Class: L8 Class: L9
Sensitivity            0.81889   1.00000
Specificity            0.98263   0.99953
Pos Pred Value         0.82786   0.99543
Neg Pred Value         0.98155   1.00000
Prevalence             0.09256   0.09371
Detection Rate         0.07580   0.09371
Detection Prevalence   0.09156   0.09414
Balanced Accuracy      0.90076   0.99976
```

The 95% prediction interval for the tuned model is (85.53%, 86.26%), which is a tighter range than that of the previous model.