Solution Lesson04: CV

Machine Learning using R

Exercise 1: Cross-validation on the Taxi data using own function

Take a look at the cross-validation function shown in the lecture and try to understand all steps of the function. Use it to perform a 10-fold cross-validation on the taxi data from the last exercise (again predicting the color of the taxis). Estimate the performance of the knn classifier for different values of k (e.g. k ranging from 1 to 10).

```
### Read in data:
dat <- read.csv('./taxi.csv', stringsAsFactors = TRUE)
### Load function:
KNN_crossVal <- function(data, label, k_fold=10, KNN_k=1){
  stopifnot(nrow(data)==length(label), is.factor(label),
            (1<k_fold & k_fold<=nrow(data)), all(apply(data, 2, is.numeric)),
            KNN_k<=nrow(data))
  # Create k sub-selections
  n <- nrow(data)
  ind_s <- sample(1:n)
  ind.L <- list()
  j1 <- 1
  for (i in 1:k_fold){
    j2 <- (i*n) %/% k_fold
    ind.L[[i]] <- ind_s[j1:j2]
    j1 <- j2+1
  }
  # Now run KNN on each selection (and collect results):
  confMat <- matrix(0,nrow=nlevels(label), ncol=nlevels(label))
  for(fold in 1:k_fold){
    ind_fold <- ind.L[[fold]]
    testDat <- data[ind_fold,]
    trainDat <- data[-ind_fold,]
    test.solu <- label[ind_fold]
    train.solu <- label[-ind_fold]
    knn.pred <- class::knn(train = trainDat, test = testDat,
                    cl = train.solu, k = KNN_k)
    confMat.fold <-  table(knn.pred, test.solu)
    confMat <- confMat + confMat.fold
  }
  # Calculate estimated test-error
  missMat <- confMat
  diag(missMat) <- 0
  missCount <- sum(missMat)
  test.err <- missCount/n
  L.res <- list(k_fold=k_fold, KNN_k=KNN_k, Indices=ind.L,
                confMatrix=confMat, errorRate=test.err)
  return(L.res)
}
### Apply function:
### Test performance for k ranging from 1 to 10:
set.seed(2449)
nk <- 10
res <- matrix(NA, ncol=2, nrow = nk)
colnames(res) <- c('k', 'testErr')
for(i in 1:nk){
  res_i <- KNN_crossVal(data = dat[,-7], label = dat$col, k_fold = 10, KNN_k = i)
  res[i, ] <- c(i, res_i$errorRate)
}
res
```

```
##        k testErr
##  [1,]  1  0.1298
##  [2,]  2  0.1452
##  [3,]  3  0.1278
##  [4,]  4  0.1342
##  [5,]  5  0.1335
##  [6,]  6  0.1401
##  [7,]  7  0.1430
##  [8,]  8  0.1488
##  [9,]  9  0.1508
## [10,] 10  0.1553
```

Although the differences in performance are rather small, the best performance was reached for k= 3.

Exercise 2: Cross-validation on the Taxi data using caret package

Perform a 10-fold cross-validation on the taxi data for different values of k using the caret package. Do the results agree with the results from our own function?

```
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
### 10-Fold CV:
### Define tune grid:
t_grid <- data.frame(k=c(1:10))   # Define k's for knn
### Define validation procedure:
train_crt <- trainControl(method = "cv", number = 10)
### "Train" the model:
set.seed(9823)
model <- train(col ~., data = dat, method = "knn", tuneGrid=t_grid,
               trControl = train_crt)
model
```

```
## k-Nearest Neighbors
##
## 10000 samples
##     6 predictor
##     2 classes: 'green', 'yellow'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 9000, 9000, 9000, 9000, 9000, 9000, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy  Kappa
##    1  0.8700    0.7400
##    2  0.8583    0.7166
##    3  0.8734    0.7468
##    4  0.8648    0.7296
##    5  0.8674    0.7348
##    6  0.8583    0.7166
##    7  0.8561    0.7122
##    8  0.8498    0.6996
##    9  0.8487    0.6974
##   10  0.8447    0.6894
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 3.
```

Again, the best performance is reached with k= 3.