# Resampling Methods for Assessing Model Accuracy

Yifei Sun, Runze Cui

## Contents

```r
library(FNN)
library(kknn)
library(caret)
library(tidymodels)
library(tidyverse)
```

You can generate a simulated training dataset or use an existing dataset. For illustration, we use a simulated dataset with two predictors.

```r
# Data generating function - you can replace this with your own function
gen_data <- function(N)
{
  X1 <- rnorm(N, mean = 1)
  X2 <- rnorm(N, mean = 1)
  eps <- rnorm(N, sd = .5)
  Y <- sin(X1) + (X2)^2 + eps
  data.frame(Y = Y, X1 = X1, X2 = X2)
}

set.seed(2024)

# generate the TRAINING data
N <- 400
trainData <- gen_data(N)
```

## (Repeated) K-fold CV from scratch

```r
# ten-fold CV
cvSplits <- vfold_cv(trainData, v = 10)
cvSplits
```

```
## #  10-fold cross-validation
## # A tibble: 10 x 2
##    splits           id
##    <list>           <chr>
##  1 <split [360/40]> Fold01
##  2 <split [360/40]> Fold02
##  3 <split [360/40]> Fold03
##  4 <split [360/40]> Fold04
##  5 <split [360/40]> Fold05
##  6 <split [360/40]> Fold06
##  7 <split [360/40]> Fold07
##  8 <split [360/40]> Fold08
##  9 <split [360/40]> Fold09
## 10 <split [360/40]> Fold10
```

```r
# cvSplits$splits[[1]]$in_id
```

Sometimes we can repeat the K-fold CV multiple times and then calculate the average prediction error.

```r
# repeated ten-fold CV
rcvSplits <- vfold_cv(trainData, v = 10, repeats = 5)
rcvSplits
```

```
## #  10-fold cross-validation repeated 5 times
## # A tibble: 50 x 3
##    splits            id      id2
##    <list>            <chr>   <chr>
##  1 <split [360/40]> Repeat1 Fold01
##  2 <split [360/40]> Repeat1 Fold02
##  3 <split [360/40]> Repeat1 Fold03
##  4 <split [360/40]> Repeat1 Fold04
##  5 <split [360/40]> Repeat1 Fold05
##  6 <split [360/40]> Repeat1 Fold06
##  7 <split [360/40]> Repeat1 Fold07
##  8 <split [360/40]> Repeat1 Fold08
##  9 <split [360/40]> Repeat1 Fold09
## 10 <split [360/40]> Repeat1 Fold10
## # i 40 more rows
```

Repeated K-fold CV from scratch:

```
M <- nrow(rcvSplits)
mse_lm <- rep(NA, M)
mse_knn <- rep(NA, M)
mse_knn2 <- rep(NA, M)

for (m in 1:M)
{
  tsdata <- analysis(rcvSplits[[1]][[m]])
  vsdata <- assessment(rcvSplits[[1]][[m]])

  fit_lm <- lm(Y~X1+X2, data = tsdata)
  pred_lm <- predict(fit_lm, vsdata)

  pred_knn <- knn.reg(train = tsdata[2:3],
                      test = vsdata[2:3],
                      y = tsdata$Y,
                      k = 3)

  pred_knn2 <- kknn(Y~.,
                    train = tsdata,
                    test = vsdata,
                    k = 3)

  mse_lm[m] <- mean((vsdata$Y - pred_lm)^2)
  mse_knn[m] <- mean((vsdata$Y - pred_knn$pred)^2)
  mse_knn2[m] <- mean((vsdata$Y - pred_knn2$fitted.values)^2)
}

c(mean(sqrt(mse_lm)), mean(sqrt(mse_knn)), mean(sqrt(mse_knn2)))
```

```
## [1] 1.4238488 0.7497125 0.7154207
```

## Resampling in caret

```
# K-fold CV
ctrl.1 <- trainControl(method = "cv", number = 10)
# leave-one-out CV
```

```
ctrl.2 <- trainControl(method = "LOOCV")
# leave-group-out / Monte Carlo CV
ctrl.3 <- trainControl(method = "LGOCV", p = 0.75, number = 50)
# bootstrap
ctrl.4 <- trainControl(method = "boot632", number = 100)
# repeated K-fold CV
ctrl.5 <- trainControl(method = "repeatedcv", repeats = 5, number = 10)
# only fit one model to the entire training set
ctrl.6 <- trainControl(method = "none")
# user-specified folds
rcvSplits_list <- map(rcvSplits$splits, ~ .x$in_id)
ctrl.7 <- trainControl(index = rcvSplits_list)
```

```
# model comparison based on ten-fold cross validation
set.seed(1)
lmFit <- train(Y~.,
               data = trainData,
               method = "lm",
               trControl = ctrl.1)

set.seed(1)
knnFit <- train(Y~.,
                data = trainData,
                method = "knn",
                trControl = ctrl.1)

# same training/validation splits?
identical(lmFit$control$index, knnFit$control$index)
```

```
## [1] TRUE
```

```
summary(resamples(list(lmFit, knnFit)), metric = "RMSE")
```

```
##
## Call:
## summary.resamples(object = resamples(list(lmFit, knnFit)), metric = "RMSE")
##
## Models: Model1, Model2
## Number of resamples: 10
##
## RMSE
##             Min.   1st Qu.    Median      Mean   3rd Qu.     Max. NA's
## Model1 0.9796773 1.1495787 1.4127151 1.4356697 1.6854552 1.935875    0
## Model2 0.6279217 0.6861809 0.7499159 0.7796583 0.7589591 1.120673    0
```

```
# user-specified: from rcvSplit (ten-fold CV, repeated 5 times)
lmFit2 <- train(Y~.,
                data = trainData,
                method = "lm",
                trControl = ctrl.7)

mean(lmFit2$resample$RMSE)
```

```
## [1] 1.423849
```

```r
knnFit2 <- train(Y~.,
                 data = trainData,
                 method = "knn",
                 tuneGrid = data.frame(k = 3),
                 trControl = ctrl.7)

mean((knnFit2$resample$RMSE))
```

```
## [1] 0.7497125
```

## Resampling in `tidymodels`

```r
# K-fold CV
rs_1 <- vfold_cv(trainData, v = 10)
# leave-one-out CV
rs_2 <- loo_cv(trainData)
# leave-group-out / Monte Carlo CV
rs_3 <- mc_cv(trainData, prop = 0.75, times = 50)
# bootstrap
rs_4 <- bootstraps(trainData, times = 100, apparent = TRUE)
# repeated K-fold CV
rs_5 <- vfold_cv(trainData, v = 10, repeats = 5)
# user-specified folds
rs_7 <- rcvSplits
```

```r
lm_spec <- linear_reg() %>%
  set_engine("lm") %>%
  set_mode("regression")
knn_spec <- nearest_neighbor(neighbors = 3) %>%
  set_engine("kknn") %>%
  set_mode("regression")


workflow_set(preproc = list(Y ~ .),
             models = list(lm = lm_spec, knn = knn_spec)) %>%
workflow_map(resamples = rs_7) %>%
  collect_metrics() %>%
  filter(.metric == "rmse") %>%
  select(model, mean)
```

```
## # A tibble: 2 x 2
##   model             mean
##   <chr>            <dbl>
## 1 linear_reg       1.42
## 2 nearest_neighbor 0.715
```