# How can I perform leave-one-out cross-validation in R, and what are some examples of its use?

Authored by

**stats writer**

April 21, 2024

Leave-one-out cross-validation is a technique used to evaluate the performance of a predictive model in R. It involves systematically leaving out one data point from the training set and using the remaining data to train the model. The model is then used to predict the value of the omitted data point, and this process is repeated for each data point in the dataset. The performance of the model is then evaluated by comparing the predicted values with the actual values of the omitted data points. This method can be useful for determining the robustness of a model and its ability to generalize to new data. Examples of its use include evaluating the accuracy of regression or classification models, selecting the optimal number of features in a model, and comparing the performance of different models.

## Leave-One-Out Cross-Validation in R (With Examples)

**To evaluate the performance of a model on a dataset, we need to measure how well the predictions made by the model match the observed data.**

**One commonly used method for doing this is known as leave-one-out cross-validation (LOOCV), which uses the following approach:**

**1. Split a dataset into a training set and a testing set, using all but one observation as part of the training set.**

**2. Build a model using only data from the training set.**

**3. Use the model to predict the response value of the one observation left out of the model and calculate the mean squared error (MSE).**

**4. Repeat this process *n* times. Calculate the test MSE to be the average of all of the test MSE's.**

**The easiest way to perform LOOCV in R is by using the trainControl() function from the caret library in R.**

**This tutorial provides a quick example of how to use this function to perform LOOCV for a given model in R.**

### Example: Leave-One-Out Cross-Validation in R

**Suppose we have the following dataset in R:**

```
#create data frame
df <- data.frame(y=c(6, 8, 12, 14, 14, 15, 17, 22, 24, 23),
x1=c(2, 5, 4, 3, 4, 6, 7, 5, 8, 9),
x2=c(14, 12, 12, 13, 7, 8, 7, 4, 6, 5))

#view data frame
df

y x1 x2
6 2 14
8 5 12
12 4 12
14 3 13
```

14 4 7

15 6 8

17 7 7

22 5 4

24 8 6

23 9 5

The following code shows how to fit a multiple linear regression model to this dataset in R and perform LOOCV to evaluate the model performance:

```
library(caret)

#specify the cross-validation method
ctrl <- trainControl(method = "LOOCV")

#fit a regression model and use LOOCV to evaluate performance
model <- train(y ~ x1 + x2, data = df, method = "lm", trControl = ctrl)

#view summary of LOOCV
print(model)
```

Linear Regression

**10 samples**

**2 predictor**

**No pre-processing**

**Resampling: Leave-One-Out Cross-Validation**

**Summary of sample sizes: 9, 9, 9, 9, 9, 9, ...**

**Resampling results:**

**RMSE Rsquared MAE**

**3.619456 0.6186766 3.146155**

**Tuning parameter 'intercept' was held constant at a value of TRUE**

**Here is how to interpret the output:**

**10 different samples were used to build 10 models. Each model used 2 predictor variables.No pre-processing occured. That is, we didn't scale the data in any way before fitting the models.The resampling method we used to generate the 10 samples was Leave-One-Out Cross Validation.The sample size for each training set was 9.RMSE: The root mean squared error. This measures the average difference between the predictions made by the model and the actual**

observations. The lower the RMSE, the more closely a model can predict the actual observations.Rsquared: This is a measure of the correlation between the predictions made by the model and the actual observations. The higher the R-squared, the more closely a model can predict the actual observations.MAE: The mean absolute error. This is the average absolute difference between the predictions made by the model and the actual observations. The lower the MAE, the more closely a model can predict the actual observations.

Each of the three metrics provided in the output (RMSE, R-squared, and MAE) give us an idea of how well the model performed on previously unseen data.

In practice we typically fit several different models and compare the three metrics provided by the output seen here to decide which model produces the lowest test error rates and is therefore the best model to use.