Lecture 18 Cross-Validation

Today’s lecture is about cross-validation. Please download tutorial 11 from the course website, unzip the zip folder and open the r Markdown file in RStudio.

Introduction

We’re going to use the water, air temperature data again in this tutorial to learn how to do cross-validation.

* In the previous tutorial, we split the datasets based on locations. The testing set contains data from 3 randomly selected rivers and the training set contains data from the rest 28 rivers.
* Therefore, not all locations are used in the training set and not all locations are used in the testing set.
* We did model selection based on the testing set. If we plan to use the selected model in the future for all the rivers, it may not perform well.
* For large dataset with enough observations in both training and testing sets, it may not be a problem. However, for small dataset, the model we selected may not be good.
* In today’s tutorial, we’re going to use cross-validation method to select a polynomial model that fits the data best.

Part 1:

Chunk 1

Before learning cross-validation, there are a few new functions that are useful in this part.

* Please run Chunk 1 in Part 1. In this part, group\_by and nest functions are used.
* nest() function is a function from tidyverse package.
* It converts rows of a dataframe into a list, and stores this list in a column called data.
* In Chunk1, the original data is first grouped by location. Then the data of each group is stored as a sub-tibble in the data column.
* There are 31 different rivers in the data set. Therefore, there are 31 rows in the Nested data set. For each location, the corresponding data is saved in a tibble in the data column.

Chunk 2

There is a function unnest(). As you can imagine, it does the opposite of nest() function. It is used to restore the original data frame. Please run Chunk 2 in part 1.

* We can do data filtering based on the nested data and use unnest() function to restore the original data frame.
* The first line of code results in all the observations from location 103 in a year.
* The second line of code are all the observations not from location 103.
* This can help us do data splitting.

Chunk 3

Next, please run Chunk3 and try to understand the code.

* You can use View() function on DATA2 to scan through the data.
* What do you notice?
* (Give 3 minutes)
* In this part, we fit a linear regression model on observations not from `L103` and test the model on data from `L103`.
* A column linpred is created and filled with missing values.
* Then we split the data into training and testing where the training data contains all the observations not from location 103, and testing data contains all the observations from location 103.
* A linear regression model is fitted on the training data set. Then the predictions for the testing data are generated based on the fitted model.
* Finally, all the predictions are saved to the corresponding rows. Since the predictions are made for data from location 103, we only have values of linpred for observations from location 103. Linpred are still missing values for all the other locations.

Chunk4:

In chunk 4, please create a loop that repeats the process in chunk 3 for each location.

* In each iteration, data from a certain location is used as the test set. A linear regression model is fitted on data from all the other locations.
* Then we make predictions for the test set based on the fitted model. These are called out-of-sample predictions as the model is fitted without the testing data.
* Please write a for-loop that complete predictions for each location and save the results to the linpred column.
* (Give 5 minutes)

Chunk 5:

Chunk 5 contains a function that outputs root of MSE for certain predictions.

* In our data set, we have actual water temperatures in W column, and we have out-of-sample predicted water temperatures.
* The differences between the two are residuals.
* We take average of squares of the residuals then take square root of it and get root of MSE.

Part 2:

In this part, let me first introduce two useful cross-validation techniques.

Leave-one-out cross-validation:

* LOOCV involves splitting the set of observations into two parts. However, instead of creating two subsets of comparable size, a single observation is used for the validation set, and the remaining observations make up the training set.
* The model is fit on the n − 1 training observations, and a single prediction is made for the excluded observation.
* Since this single observation was not used in the fitting process, we can calculate a test error based on this single observation.
* This process is repeated n times. Every time, one observation is put into the validation set and a test error is calculated. The LOOCV estimate for the test MSE is the average of these n test errors.

K-Fold Cross Validation

An alternative to LOOCV is k-fold CV.

* This approach involves randomly dividing the set of observations into k groups of approximately equal size.
* The first fold is treated as a validation set, and the method is fit on the remaining k − 1 folds.
* The mean squared error, MSE1, is then computed on the observations in the held-out fold.
* This procedure is repeated k times; each time, a different group of observations is treated as a validation set.
* This process results in k estimates of the test error, MSE1, MSE2, . . . , MSEk. The k-fold CV estimate is computed by averaging these values.
* It is not hard to see that LOOCV is a special case of k-fold CV in which k is set to equal n. In practice, one typically performs k-fold CV using k = 5 or k = 10.

Part 1 can be viewed as 31-Fold Cross-validation. (More)

Part 2:

* Visualize the relationship between day of year, water temperature and air temperature.
* High water and air temperature in Summer.
* Both day of year and air temp can be used in model.