Applied Machine Learning in Health Sciences 2024

Exercises

This document will be updated with additional checkpoints for each course day throughout the course.

Rev: 4 Mar 2024.

1 Introduction

Checkpoint 1: ✍

Think of some real-life applications for *supervised learning* from your research field. Describe two applications/research questions in which classification or regression might be useful. Describe the predictors/input variables as well as the response/output variable, data set size, number of variables etc. Is the goal of each application *inference* or *prediction*? Explain your answer.

Checkpoint 2: ✍

Think of some real-life applications for *unsupervised learning* from your research field. Describe two applications/research questions in which unsupervised learning might be useful. Describe the data variables, data set size, number of variables etc. What is the goal of the analysis? Explain your answer.

2 Linear regression

Checkpoint 3: ✍

Do section 3.7 conceptual exercise 4. in ISL (ISL page 122).

Checkpoint 4: ⌨) Linear regression - body density data
In this checkpoint you will analyse the body density data using linear regression. The goal is to build a linear regression model to predict *body density* from *chest circumference measurements.*

- Open the script `main1a.{py,R,m}` and review it to understand the different steps in the script. Some of the code needed to answer this checkpoint is already provided in the script, but you also need to do a bit of coding yourself.

Use the script `main1a.{py,R,m}` and your code to answer the following:

(a) Load the data set `bodyMeasurementsSingleTrainTest.txt`. Describe the data, what are the sizes of the training- and test sets? how many observations and features? what is the numerical range of the variables?

(b) Identify the part for of the script that is used for creating the polynomial expansion of the input variable. Try do create polynomial expansions with different polynomial order from 1 to 7, how many columns/features are there in the resulting data matrices?

(c) Part of the code is scaling the individual columns of the polynomial regressors. Explain how this scaling is done and why the scaling may be necessary (Hint: what is the numerical range of the individual columns?).

(d) Look at the help documentation of the model fitting functions and use a bit of time to familiarize yourself with this function. What are the inputs to the function and what is the output?

(e) Explain how training error and test error are quantified in the script?

(f) Run the analysis with polynomial order ranging from 1 to 7 (Hint: include a for-loop in the script). For each of these seven models, make a plot of body density (ordinate) vs. chest circumference (abscissa) for the training- and test data as well as the model's predictions (all three in the same plot). Include the plots in your report and describe/discuss the plots.

(g) Write your own code to make a plot of training error and test error vs. polynomial order (order 1 to 7). Include the plot in your report and describe/discuss the plot. Do you observe severe overfitting for some polynomial orders?

3 Bias-variance decomposition

Checkpoint 5: ✍ Training- and test errors

(a) Explain the difference between a training- and a test data. Also explain the difference between training- and test error. (b) Argue why we typically are interested in good model performance in terms of low test error rather than in terms of low training error.

Checkpoint 6: ✍

Do section 2.4 conceptual exercise 3. in ISL (ISL page 53).

4 Model evaluation and resampling

Checkpoint 7: ✍

Explain, in your own words, what a training set is, a validation set is, and a test set is? Why is this partition of data needed? Explain how k-fold cross validation is implemented. Explain how leave-one-out (LOO) cross validation is implemented. What are the advantages and disadvantages of k-fold cross validation relative to the validation set approach and the LOO cross validation approach?

Checkpoint 8: ⌨

Python users: By using the Python functions `modelSelection.train_test_split` and `modelSelection.KFold` we can create random partitions of data.

- Look at the function documentation and use a bit of time to familiarize yourself with these functions.

(a) Run the command `c = modelSelection.train_test_split(range(20),test_size=0.2, random_state=42)` and explain what the function does. Also look at the content of `c` and describe the variables contained.

(b) Run the commands
`cGen = modelSelection.KFold(n_splits=10, shuffle=True, random_state=42)` and
`c = list(cGen.split(np.zeros((20,1))))` and explain what the function does. Also look at the content of `c` and describe the variables contained.

(c) Explain why it is generally recommended to initialize the random generator before creating the random partitions.

R users: By using the functions `createDataPartition` and `createFolds` we can create random partitions of data.

- Look at the help text for these functions and use a bit of time to familiarize yourself with the functions.

(a) Run the command `c = createDataPartition(y = 1:20, p = 0.8)` and explain what the function does. Also describe the content of the variable `c`.

(b) Run the command `c = createFolds(y = 1:20, k = 10, returnTrain = TRUE)` and explain what the function does. Also explain the content of the variable `c`.

(c) Explain why it is generally recommended to run e.g. the command `set.seed(0)` before creating the random partitions.

Matlab users: By using the Matlab function `cvpartition` we can create random partitions of data.

- Type `doc cvpartition` and use a bit of time to familiarize yourself with this function.

(a) Run the command `c = cvpartition(20,'Holdout',0.2)` and explain what the function does. Also type `c.training` and `c.test` and describe the variables contained in these fields.

(b) Run the command `c = cvpartition(20,'KFold',10)` and explain what the function does. Also type `c.training(1)` and `c.test(1)`, `c.training(2)` and `c.test(2)`, `c.training(3)` and `c.test(3)` etc. and describe the variables contained in these fields.

(c) Explain why it is generally recommended to run e.g. the command `rng('default')` before creating the random partitions.

Checkpoint 9: ⌨) Linear regression - body density data - cross validation
In this checkpoint you will analyse the body density data using linear regression (with polynomial regressors), and the goal is to build a linear regression model to predict *body density* from *chest circumference measurements*. You will use k-fold cross validation to evaluate model performance for different polynomial order.

- Open the script `main1b.{py,R,m}` and review it to understand the different steps in the script. Some of the code needed to answer this checkpoint is already provided in the script, but you also need to do a bit of coding yourself.

Use the script and your code to answer the following:

(a) Load the data set `bodyMeasurementsSingleCV.txt`. Describe the data, what is the size of the data set? how many observations and features?

(b) Identify the part of the script where the random partition is performed. How many folds are used in the k-fold cross validation?

(c) The script contains two for-loops (one nested within the other). Explain, in your own words, how the analysis is performed/structured in the script.

(d) Run the script. Look at the plot of training and test error (cross-validation error) (ordinate) vs. polynomial order (abscissa) (both error curves in same plot). Include the plot in your report and describe/discuss it. Are the curves as expected? do you observe severe overfitting (compare with your result from Checkpoint 4)? if not, try to explain why not (Hint: look at the number of training observations and model flexibility). For which polynomial order do you observe the lowest test error?

- -
PMR, pmr@cfin.au.dk, Mar 2024.