Assignment 10

EMATM0061: Statistical Computing and Empirical Methods, TB1, 2022

Dr. Rihuan Ke

Introduction

This is the 10th assignment for Statistical Computing and Empirical Methods (Unit EMATM0061) on the MSc in Data Science & MSc in Financial Technology with Data Science. This assignment is mainly based on Lectures 25, 26 and 27 (see the Blackboard).

The submission deadline for this assignment is 23:59, 12 December 2022. Note that this assignment will not count towards your final grade. However, it is recommended that you try to answer the questions to gain a better understanding of the concepts.

Create an R Markdown for the assignment

It is a good practice to use R Markdown to organize your code and results.

If you are considering submitting your solutions, please generate a PDF file. For example, you can choose the "PDF" option when creating the R Markdown file (note that this option may require Tex to be installed on your computer), or use R Markdown to output an HTML and print it as a PDF file in a browser, or use your own way of creating a PDF file that contains your solutions.

Only a PDF file will be accepted in the submission of this assignment. To submit the assignment, please visit the "Assignment" tab on the Blackboard page, where you downloaded the assignment.

Wish to know more about a perticular question?

You may want to ask a question during the computer lab.

Alternatively, we are collecting questions about this assignment that need to be addressed, through the following form. And this can be done either during the labs or outside the lab sessions. So, If you found a question in this assignment interesting but had difficulty in a particular step when trying to develop your answer, please put your remark in the form via the following link. A brief description of the difficulty would be very helpful. Giving your remark is optional, but we aim to know the most common questions that you might want to get some support.

https://forms.office.com/e/LrffuTGPj9

Load packages

Some of the questions in this assignment require the tidyverse package. If it hasn't been installed on your computer, please use `install.packages()` to install them first.

To road the tidyverse package:

```
library(tidyverse)
```

1. Linear discriminant analysis

(Q1)

Describe the probabilistic model that underpins linear discriminant analysis.

(Q2)

In this question, we will train a linear discriminant analysis model to carry out the classification task to predict whether a hawk belongs to either the "Sharp-shinned" or the "Cooper's" species of hawks, based on a four dimensional feature vector containing the weight, and the lengths of the wing, the tail and the hallux, which are generated by the following code.

```
library(Stat2Data)
data(Hawks)

hawks_total <- Hawks %>% select( Weight, Wing, Hallux, Tail, Species) %>%
  filter(Species=='SS' | Species =='CH') %>% drop_na() %>%
  mutate(Species=as.numeric(Species=='SS'))
```

Assume that we have the following train-test split of our dataset.

```
num_total <- hawks_total %>% nrow() # number of penguin data
num_train <- floor(num_total*0.6) # number of train examples
num_test <- num_total-num_train # number of test samples
set.seed(0) # set random seed for reproducibility

test_inds <- sample(seq(num_total),num_test) # random sample of test indicies
train_inds <- setdiff(seq(num_total),test_inds) # training data indicies

hawks_train <- hawks_total %>% filter(row_number() %in% train_inds) # train data
hawks_test <- hawks_total %>% filter(row_number() %in% test_inds) # test data
```

Now, train a linear discriminant analysis model to carry out the classification task described above. Compute and report the train error and the test error.

(Q3)

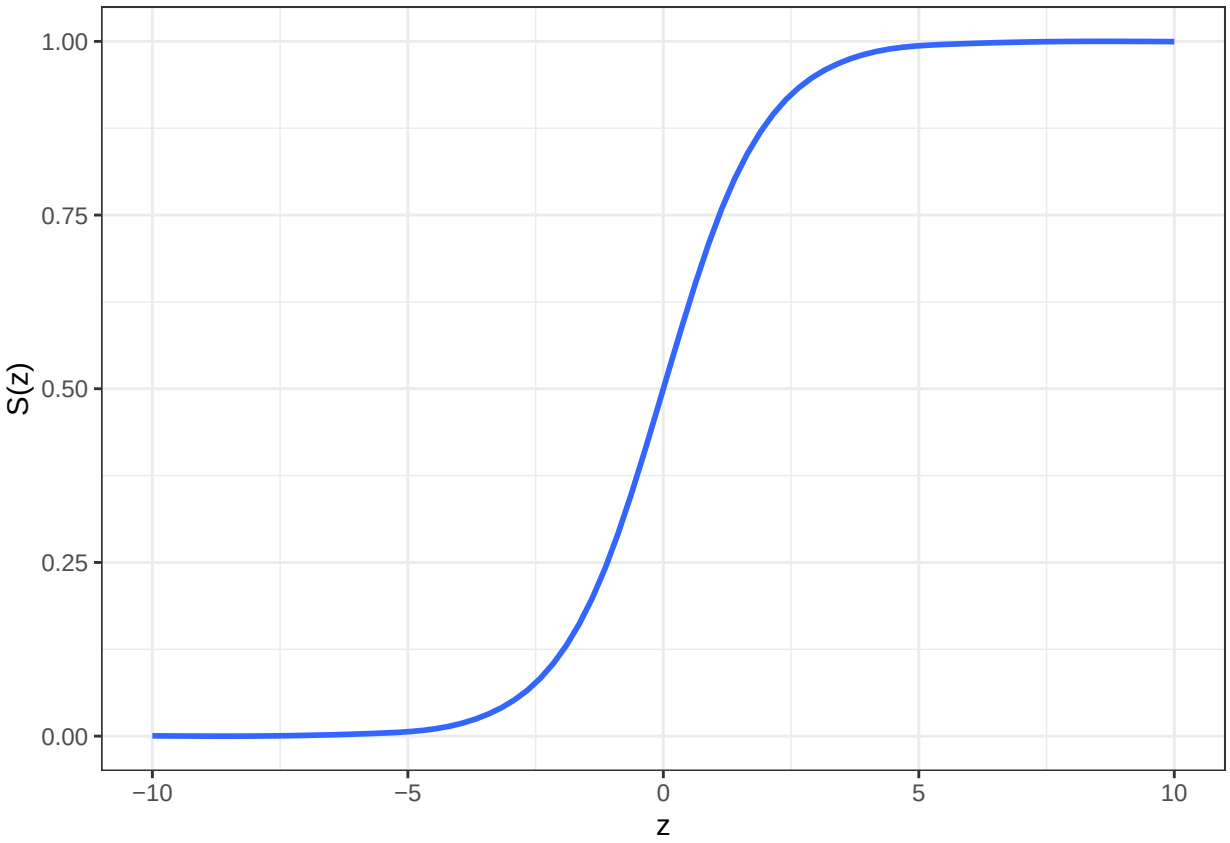
(*Optional) Implement your own linear discriminant analysis model.

2. Logistic regression

(Q1) Describe the probabilistic model which underpins logistic regression.

(Q2)

Recall that the sigmoid function $\mathcal{S} : \mathbb{R} \to (0, 1)$ is defined by $S(z) = 1/(1 + e^{-z})$. Generate the following plot which displays the sigmoid function:



(Q3) Now train a logistic regression model to predict whether a hawk belongs to either the "Sharp-shinned" or the "Cooper's" species of hawks, based on a four-dimensional feature vector containing the weight, and the lengths of the wing, the tail and the hallux (similar to what you did in the Linear discriminant analysis question above). Compute and report both the training error and the test error.

(Q4) (*optional)

Consider the following formula for the log-likelihood of the weights $w \in \mathbb{R}^d$ and bias $w^0 \in \mathbb{R}$, given data $\mathcal{D} = ((X_1, Y_1), \cdots, (X_n, Y_n))$:

$$\log l\big(\omega, \omega^0\big) = \sum_{i=1}^{n} \log S\big\{(2Y_i - 1) \cdot \big(wX_i^T + w^0\big)\big\}$$

Show that $\frac{\partial \log S(z)}{\partial z} = S(-z)$ and use it to demonstrate the following formulas for the derivatives:

$$\frac{\partial}{\partial \omega} \log l\big(\omega, \omega^0\big) = \sum_{i=1}^{n} 2(Y_i - 1) S\big\{(1 - 2Y_i) \cdot \big(wX_i^T + w^0\big)\big\} \cdot X_i$$

$$\frac{\partial}{\partial \omega^0} \log l\big(\omega, \omega^0\big) = \sum_{i=1}^{n} 2(Y_i - 1) S\big\{(1 - 2Y_i) \cdot \big(wX_i^T + w^0\big)\big\}$$

Explain the role the above formula has in training a logistic regression model.

You can learn more about the glmnet approach to logistic regression here:

https://glmnet.stanford.edu/articles/glmnet.html#logistic-regression

3. Basic concepts in regularisation

Regularisation refers to the general technique within supervised learning of modifying an objective in some way so as to reduce the gap between test error and training error. Typically, this will increase the error on the training data. However, by reducing the gap between test and training errors, we can often improve performance (on unseen data).

Examples include

1) l_2 regularisation in the context of ridge regression for regression
2) l_1 or l_2 regularised logistic regression for linear classification

(Q1)

Let's review some key concepts relevant to regularisation. Write down your explanation of each of the following concepts.

1. Hyper-parameter (and give an example of a hyper-parameter)
2. Validation data
3. The train-validation-test split

(Q2) What is the Euclidean (l_2) norm and what is the l_1 norm of a vector?

(Q3) The Ridge regression method and the Lasso method are both for learning a linear regression model from the data, by minimising an objective function. Describe what kinds of terms are included in their objective functions. What is the difference between the two objective function?

4. An investigation into ridge regression for high-dimensional regression

In this question we consider a high-dimensional regression problem. We consider a problem of predicting the melting point of a chemical compound from a relatively high-dimensional feature vector of chemical descriptors.

To do this we shall use data from the "`QSARdata`" data library. Begin by installing the "`QSARdata`" library as follows.

```
install.packages("QSARdata")
```

Next load the "`QSARdata`" library and loading the "`MeltingPoint`" data set.

```
library(QSARdata)
data(MeltingPoint)
```

You will find a data frame called "`MP_Descriptors`". The rows of the data frame correspond to different examples of chemical compounds and the columns correspond to various chemical descriptors. In addition you will find a vector called "`MP_Outcome`" which contains the corresponding melting point for each of the examples.

(Q1)

Begin by combining the data-frame of feature vectors "`MP_Descriptors`" together with the column vector of melting points "`MP_Outcome`". Combine these together into a single data frame entitled "`mp_data_total`" as follows.

```
mp_data_total<-MP_Descriptors %>%
  mutate(melting_pt=MP_Outcome)
```

How many variables are in your data frame? How many examples?

(Q2)

Next carry out a train-validate-test split of the "`mp_data_total`" data frame. You should use about 50% of the data to train the algorithm, about 25% to validate and about 25% to train.

(Q3)

Our goal is to find a linear regression model $\phi_{w,w^0} : \mathbb{R}^d \to \mathbb{R}$ by $\phi_{w,w^0}(x) = wx^T + w^0$ which estimates the melting point based upon the chemical descriptors. Create a function which takes as input training data (a matrix of features for training data and a vector of labels for training data), validation data (a matrix of features for validation data and a vector of labels for validation data) and a hyper-parameter λ. The function should train a ridge regression model using the specified value of λ, then compute the validation error and output a single number corresponding to the validation error. Your function should not be specific to this particular regression problem, but should apply to ridge regression problems in general. You can use the `glmnet` library within your function.

(Q4)

Next generate a sequence of candidate hyper-parameters called "`lambdas`". The sequence should begin with 10^{-5} and increase geometrically in multiples of 1.25 and should be of length 70. That is, "`lambdas`" should contain the numbers

$$10^{-5}, 10^{-5} \times 1.25, 10^{-5} \times 1.25^2, \cdots, 10^{-5} \times 1.25^{69}, 10^{-5} \times 1.25^{70},$$

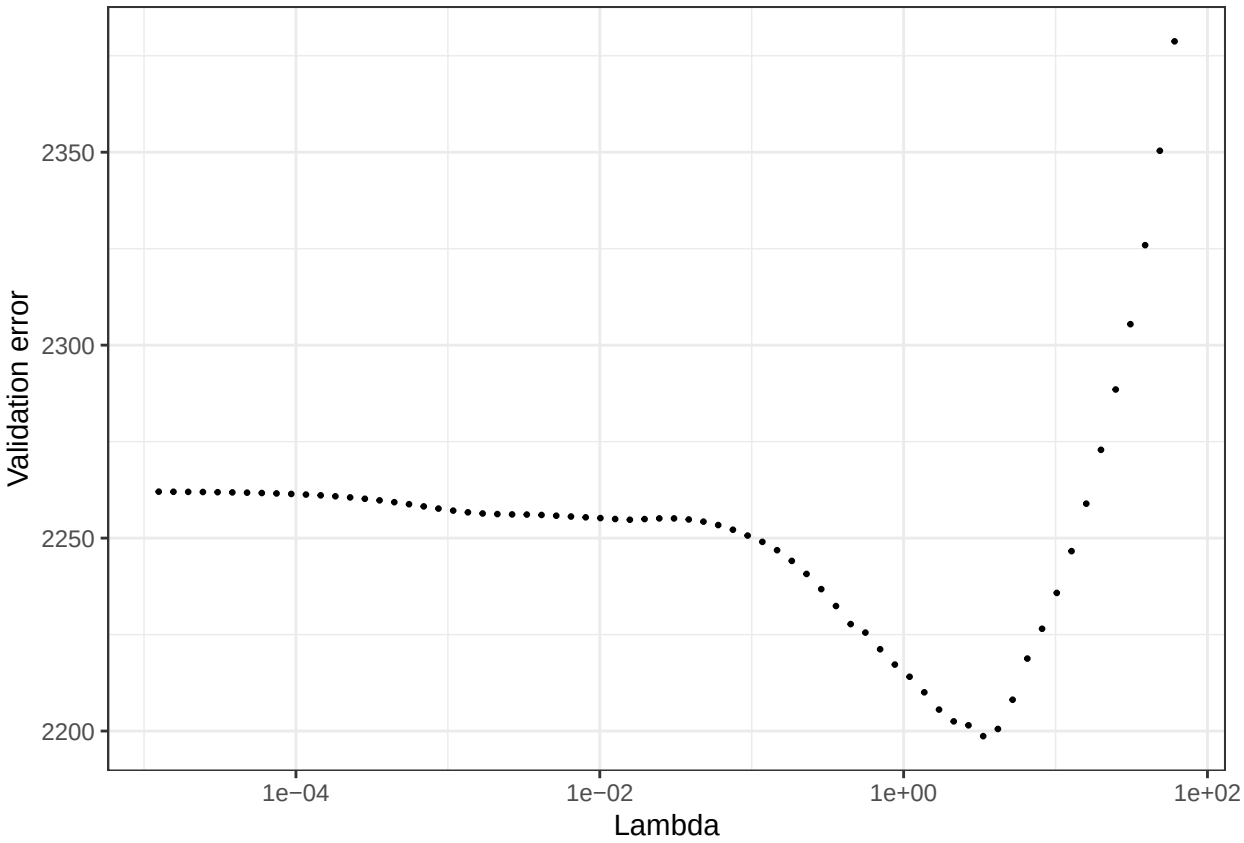
(Q5)

Now use your function to estimate the mean squared error on the validation data for a ridge regression model for the problem of predicting the melting point based on the chemical descriptors. Consider all of the hyperparameter values within your vector "`lambdas`". Store the results of this procedure in a data frame.

(Q6)

Plot the validation error as a function of the hyper-parameter λ. Use the `scale_x_continuous()` function to plot the λ coordinate on a logarithmc scale.

Your plot may look like this:



(Q7)

Now use your results data frame to determine the hyper-parameter λ with the lowest validation error. Retrain your ridge regression model with your selected value of the hyper-parameter λ and estimate the test error by computing the mean squared error on the test data.

(Q8) (*optional)

Does the test error computed above lead to a biased estimate of the mean squared error? Why can't we use the mean squared error on validation data for the ridge regression model with the selected hyper-parameter as an estimate of the mean squared error on test data? Observe that the ridge regression model with the selected choice of λ has actually been trained twice: It was trained in order to compute the validation error, and then again to compute the test error. Comment on the computational efficiency of this procedure. Could it be easily improved? What memory implications would this have?