Assignment 10 for Statistical Computing and Empirical Methods

Dr. Henry WJ Reeve

Teaching block 1 2021

Introduction

This document describes your tenth assignment for Statistical Computing and Empirical Methods (Unit EMATM0061) on the MSc in Data Science. Before starting the assignment it is recommended that you first watch video lectures 24 and 25.

1 Basic concepts in regression

Write down your explanation of each of the following concepts. Give an example where appropriate.

1. A regression model
2. Training data
3. Mean squared error on the test data
4. Mean squared error on the training data
5. Supervised learning
6. Linear regression model

2 Basic concepts in regularisation

Write down your explanation of each of the following concepts. Give an example where appropriate.

1. Regularisation
2. Hyper-parameter
3. The Euclidean norm
4. The ℓ_1 norm
5. Validation data
6. The train-validation-test split
7. Ridge regression
8. The Lasso

3 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 coloumns correspond to various chemical descriptors. In addition you will find a vector call "MP_Outcome" which contains the corresponding melting point for each of the examples. Begin by combining the dataframe of feature vectors "MP_Descriptors" together with the coloumn 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%>%
  add_column(melting_pt=MP_Outcome)
```

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

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.

Our goal is to find a linear regression model $\phi_{w,w^0} : \mathbb{R}^d \to \mathbb{R}$ by $\phi_{w,w^0}(x) = w\, x^\top + w^0$ 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.

Next generate a sequence of candidate hyper-parameters called "lambdas". The sequence should begin with 10^{-8} and increase geometrically in multiples of 1.25 and should be of length 100. That is, "lambdas" should contain the numbers

$$10^{-8}, 10^{-8} \times 1.25, 10^{-8} \times 1.25^2, \cdots, 10^{-8} \times 1.25^{99}, 10^{-8} \times 1.25^{100}.$$

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 upon the chemical descriptors. Consider all of the hyper-parameter values within your vector "lambda". Store the results of this procedure in a data frame.

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.

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. Does this lead to a biased estimate?

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 proceedure. Could it be easily improved? What memory implications would this have?

4 Comparing ℓ_1 and ℓ_2 regularisation for logistic regression

As an optional extra consider regularised logistic regression. Our goal is to learn a linear classifier $\phi_{w,w^0} : \mathbb{R}^d \to \{0,1\}$ of the form $\phi_{w,w^0}(x) = \mathbf{1}\left\{w\, x^\top + w^0\right\}$.Suppose we have a set of training data $\mathcal{D} = ((X_1, Y_1), \cdots, (X_n, Y_n))$ for a binary classification setting with feature vectors X_i taking values in $\mathbb{R}^d$ and binary labels Y_i taking values in $\{0,1\}$. The ℓ_2 regularised logistic regression minimises the following objective

$$\mathcal{R}^{\mathsf{bn}}_{0,\lambda}\left(\phi_{w,w^0}\right) = \frac{1}{n}\sum_{i=1}^{n}\left\{-\log \mathrm{S}\left((2Y_i - 1)\cdot(wX_i^\top + w^0)\right)\right\} + \frac{1}{2}\|w\|_2^2,$$

over weights $w \in \mathbb{R}^d$ and a bias $w^0 \in \mathbb{R}$, where $\lambda > 0$ is a hyper-parameter and $\|w\|_2$ denotes the ℓ_2 norm (also) known as the Euclidean norm.

The ℓ_1 regularised logistic regression minimises the following objective

$$\mathcal{R}^{\mathsf{bn}}_{1,\lambda}\left(\phi_{w,w^0}\right) = \frac{1}{n}\sum_{i=1}^{n}\left\{-\log \mathrm{S}\left((2Y_i - 1)\cdot(wX_i^\top + w^0)\right)\right\} + \|w\|_1,$$

over weights $w \in \mathbb{R}^d$ and a bias $w^0 \in \mathbb{R}$, where $\lambda > 0$ is a hyper-parameter and $\|w\|_1$ denotes the ℓ_1 norm.

Regularisation by penalising an ℓ_1 or ℓ_2 norm is often referred to as "weight-decay". Use your formulas for the derivatives to justify this name.

What does it mean for a high-dimensional vector $w \in \mathbb{R}^d$ to be sparse?

Do the formulas for the derivatives help explain why ℓ_1 regularisation is more likely to give rise to sparse solutions than ℓ_2 regularisation?

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