Supervised learning: Regression 2

Contents

Introduction **1**

Best subset selection **1**

Regularisation with glmnet **4**

Tuning lambda **5**

Exercise: method comparison **5**

Introduction

In this practical, you will learn how to handle many variables with regression by using variable selection techniques, and how to tune hyperparameters for these techniques. This practical has been derived from chapter 6 of ISLR.

One of the packages we are going to use is `glmnet`. For this, you will probably need to `install.packages("glmnet")` before running the `library()` functions.

```
library(ISLR)
library(glmnet)
library(tidyverse)
```

Best subset selection

Our goal for today is to use the `Hitters` dataset from the `ISLR` package to predict `Salary`.

---

1. **Prepare a dataframe `baseball` from the `Hitters` dataset where you remove the baseball players for which the `Salary` is missing. How many baseball players are left?**

---

2. **Create `baseball_train` (50%), `baseball_valid` (30%), and `baseball_test` (20%) datasets.**

---

3. **Create a function called `lm_mse()` with as inputs (1) a formula, (2) a training dataset, and (3) a test dataset which outputs the mse on the test dataset for predictions from a linear model.**

Start like this:

```r
lm_mse <- function(formula, train_data, valid_data) {
  y_name <- as.character(formula)[2]
  y_true <- valid_data[[y_name]]

  # The remainder of the function here
}
```

4. **Try out your function with the formula `Salary ~ Hits + Runs`, using `baseball_train` and `baseball_valid`.**

We have pre-programmed a function for you to generate as a character vector *all* formulas with a set number of `p` variables. You can load the function into your environment by *sourcing* the `.R` file it is written in:

```r
source("generate_formulas.R")
```

You can use it like so:

```r
generate_formulas(p = 2, x_vars = c("x1", "x2", "x3", "x4"), y_var = "y")
```

```
## [1] "y ~ x1 + x2" "y ~ x1 + x3" "y ~ x1 + x4" "y ~ x2 + x3" "y ~ x2 + x4"
## [6] "y ~ x3 + x4"
```

5. **Create a character vector of all predictor variables from the `Hitters` dataset. `colnames()` may be of help. Note that `Salary` is not a predictor variable.**

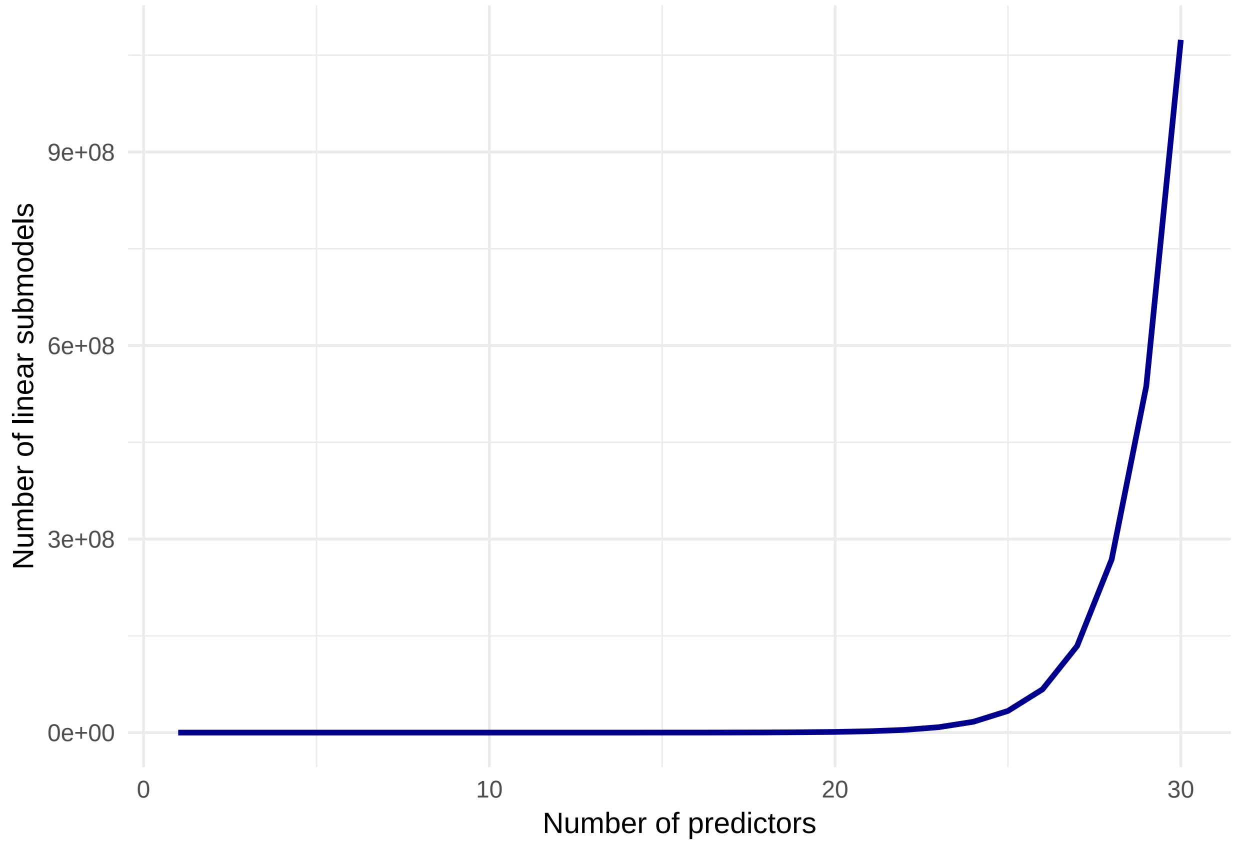
6. **Generate all formulas with as outcome `Salary` and 3 predictors from the `Hitters` data. Assign this to a variable called `formulas`. There should be 969 elements in this vector.**

7. **Use a `for loop` to find the best set of 3 predictors in the `Hitters` dataset based on MSE. Use the `baseball_train` and `baseball_valid` datasets.**

8. **Do the same for 1, 2 and 4 predictors. Now select the best model with 1, 2, 3, or 4 predictors in terms of its out-of-sample MSE**

9. **Calculate the test MSE for this model. Then, create a plot comparing predicted values (mapped to x position) versus observed values (mapped to y position) of `baseball_test`.**

Through enumerating all possibilities, we have selected the best subset of at most 4 non-interacting predictors for the prediction of baseball salaries. This method works well for few predictors, but the computational cost of enumeration increases quickly to the point where it is infeasible to enumerate all combinations of variables:



Regularisation with glmnet

glmnet is a package that implements efficient (quick!) algorithms for LASSO and ridge regression, among other things.

---

10. **Read through the help file of glmnet. We are going to perform a linear regression with normal (gaussian) error terms. What format should our data be in?**

Again, we will try to predict baseball salary, this time using all the available variables and using the LASSO penalty to perform subset selection. For this, we first need to generate an input matrix.

---

11. **First generate the input matrix using (a variation on) the following code. Remember that the "." in a formula means "all available variables". Make sure to check that this x_train looks like what you would expect.**

```r
x_train <- model.matrix(Salary ~ ., data = baseball_train %>% select(-split))
```

The model.matrix() function takes a dataset and a formula and outputs the predictor matrix where the categorical variables have been correctly transformed into dummy variables, and it adds an intercept. It is used internally by the lm() function as well!

---

12. **Using glmnet(), perform a LASSO regression with the generated x_train as the predictor matrix and Salary as the response variable. Set the lambda parameter of the penalty to 15. NB: Remove the intercept column from the x_matrix – glmnet adds an intercept internally.**

---

13. **The coefficients for the variables are in the beta element of the list generated by the glmnet() function. Which variables have been selected? You may use the coef() function.**

---

14. **Create a predicted versus observed plot for the model you generated with the baseball_valid data. Use the predict() function for this! What is the MSE on the validation set?**

Tuning lambda

Like many methods of analysis, regularised regression has a *tuning parameter*. In the previous section, we've set this parameter to 15. The `lambda` parameter changes the strength of the shrinkage in `glmnet()`. Changing the tuning parameter will change the predictions, and thus the MSE. In this section, we will select the tuning parameter based on out-of-sample MSE.

---

15. **Fit a LASSO regression model on the same data as before, but now do not enter a specific `lambda` value. What is different about the object that is generated? Hint: use the `coef()` and `plot()` methods on the resulting object.**

For deciding which value of lambda to choose, we could work similarly to what we have don in the best subset selection section before. However, the `glmnet` package includes another method for this task: cross validation.

---

16. **Use the `cv.glmnet` function to determine the `lambda` value for which the out-of-sample MSE is lowest using 15-fold cross validation. As your dataset, you may use the training and validation sets bound together with bind_rows(). What is the best lambda value?**

---

17. **Try out the plot() method on this object. What do you see? What does this tell you about the bias-variance tradeoff?**

---

18. **Use the `predict()` method directly on the object you just created to predict new salaries for the baseball players in the `baseball_test` dataset using the best lambda value you just created (hint: you need to use the `s` argument, look at `?predict.cv.glmnet` for help). Create another predicted-observed scatter plot.**

Exercise: method comparison

---

19. **Create a bar plot comparing the test set (baseball_test) MSE of (a) linear regression with all variables, (b) the best subset selection regression model we created, (c) LASSO with lambda**

set to 50, and (d) LASSO with cross-validated lambda. As training dataset, use the rows in both the `baseball_train` and `baseball_valid`
