UiO : Department of Physics
University of Oslo

Regression analysis and resampling methods

Erik Skaar
Sondre Torp
Mikael Kiste

Contents

1 Introduction **2**

2 Theory **2**
2.1 Standard . . . 2
2.2 Ridge . . . 2
2.3 Lasso . . . 2
2.4 k-fold and bootstrap . . . 2

3 Method **2**

4 Implementation **3**
4.1 Scikit vs. manually implementation . . . 3
4.2 Time evolution . . . 3
4.3 Noise - MSE evolution . . . 3

5 Result & Discussion **4**

6 Conclusion **4**

7 Appendix **4**

Abstract

[1]

1 Introduction

2 Theory

2.1 Standard

$$\beta = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y}$$

2.2 Ridge

$$\beta = \left(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{y}$$

2.3 Lasso

$$\beta = \text{argmin}_\beta \left\{ \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j|^q \right\}$$

2.4 k-fold and bootstrap

3 Method

4 Implementation

The three different algorithms discussed in section xxx was implemented in our script. It is a few different versions, but the ëversion contains all you need. All the scripts discussed in this report can be found at our github.

The program was tested on the Frank-function, see equation 1. With an known solution we did a k-fold test and an degree and λ/α test. Both tested was done with the script descriped earlier. The tables below shows the different results.

$$f(x,y) = \frac{3}{4}e^{\left(-\frac{(9x-2)^2}{4}-\frac{(9y-2)^2}{4}\right)} + \frac{3}{4}e^{\left(-\frac{(9x+1)^2}{49}-\frac{(9y+1)}{10}\right)} + \frac{1}{2}e^{\left(-\frac{(9x-7)^2}{4}-\frac{(9y-3)^2}{4}\right)} - \frac{1}{5}e^{\left(-(9x-4)^2-(9y-7)^2\right)} \quad (1)$$

4.1 Scikit vs. manually implementation

4.2 Time evolution

Table 1: This tables shows how the MSE evoles for different degrees. Scikit OLS is to confirm that our implementation is not retarded. For lasso and ridge the λ/α was set to 1e-5. Also, if we go beyond fifth order the OLS solutions starts to crumble.

degree ↓	method →	OLS	SCIKIT	RIDGE	SCIKIT LASSO
2		0.01517	0.25830	0.00516	0.00543
$2_{relative}$		1.00	1.00	1.00	1.00
$3_{relative}$		2.42	1.58	2.45	2.38
$4_{relative}$		3.63	2.45	5.11	4.88
$5_{relative}$		4.98	3.61	8.77	8.31

4.3 Noise - MSE evolution

Table 2: This tables shows how the MSE evoles for different degrees. Scikit OLS is to confirm that our implementation is not retarded. For lasso and ridge the λ/α was set to 1e-5. Also, if we go beyond fifth order the OLS solutions starts to crumble.

degree ↓	method →	OLS	SCIKIT	RIDGE	SCIKIT LASSO
0		0.00127	0.00127	0.00514	0.00127
$0_{relative}$		1.00	1.00	1.00	1.00
$0.01_{relative}$		1.03	1.03	1.00	1.03
$0.2_{relative}$		12.84	12.84	3.68	12.84
$0.5_{relative}$		42.04	42.04	10.84	42.04

Table 3: This tables shows how the MSE evoles for different degrees. Scikit OLS is to confirm that our implementation is not retarded. For lasso and ridge the λ/α was set to 1e-5. Also, if we go beyond fifth order the OLS solutions starts to crumble.

degree ↓	method →	OLS	SCIKIT	RIDGE	SCIKIT LASSO
0		0.98	0.98	0.91	0.98
0.01		0.98	0.98	0.91	0.98
0.2		0.68	0.68	0.62	0.68
0.5		0.28	0.28	0.25	0.28

5 Result & Discussion

6 Conclusion

References

[1] Morten Hjorth-Jensen. *Computational Physics*. Lecture notes. 2015. URL: https://github.com/CompPhysics/ComputationalPhysics/blob/master/doc/Lectures/lectures2015.pdf.

7 Appendix