Ordinary least squares regression of Franke's function

Anders Eriksen

Contents

1 introduction 1

2 Methods 2

3 Results 4

4 Conclusion 4

5 Appendix 4

List of Figures

List of Tables

Abstract

The main summary of the work

1 introduction

aims and rationale of the physics case, what you've done as well as a brief summary of the report
o Motivate the reader
o What done
o Structure of report

2 Methods

Theoretical models and technicalities.
o describe methods and algorithms
o explain implementation of methods and say something about the structure of algorithm and present parts of code
o Plug in some calculations to test code, such as selected runs used to validate and verify results. Latter extremely important. Reader needs to know that your code reproduces selected benchmarks and reproduces previous results, either numerical and/or well-known closed-form expressions.

The aim is to study regression models on data. These create a continuous function with which the input data is fitted. The first, and most basic is the *Ordinary Least Squares method* (*OLS*)

To test and validate the algorithms, a closed form function is an advantage. We wish to predict terrain. there is a 2-dimensional function called *Franke's function*[] which can transform a set of coordinate vectors with values between $[0, 1]$ into height values through a sum of weighted exponents.

$$\begin{aligned} f(x, y) &= \frac{3}{4}\exp\left\{-\frac{(9x-2)^2}{4} - \frac{(9y-2)^2}{4}\right\} \\ &+\frac{3}{4}\exp\left\{-\frac{(9x+1)^2}{49} - \frac{(9y+1)}{10}\right\} \\ &+\frac{1}{2}\exp\left\{-\frac{(9x-7)^2}{4} - \frac{(9y-3)^2}{4}\right\} \\ &-\frac{1}{5}\exp\left\{-(9x-4)^2 - (9y-7)^2\right\} \end{aligned} \tag{1}$$

To generate the data, one can create a uniformly distributed set of initial values ordered from low to high. These initial arrays are then combined into coordinate data through the numpy meshgrid function before being passed to Franke's function. These are matrices, and they are therefore flattened to 1D arrays through numpy's *ravel()* function.

Next comes setting up the model. Firstly, constructing the model design matrix. We want a design matrix for variable levels of model complexity, so we construct it from a polynomial combination of the input parameters x and y.

Next is to fit the model. The linear model, taken from Hastie et al[], predicts the value Y as

$$Y = X^T \beta. \tag{2}$$

Algorithm 1 make design matrix X given input $\vec{x}, \vec{y}$ and dimension n

```
create X as a matrix of ones with dimension $length(x)$ and $(int)i(i+1)/2$
for $i = 1$ to $n + 1$ do
    $q = (int)i(i+1)/2$
    for $k = 0$ to $i + 1$ do
        $X[:, q + k] = x^{i-k} \cdot y^k$
    end for
end for
```

Y is given through the inner product of the transpose of X and β. X being the design matrix mentioned above.

One method to approximate this, is with the *Residual Sum of Squares*

$$RSS(\beta) \; = \; \sum_{i=1}^{N} (y_i - x_i^T \beta)^2 .[] \tag{3}$$

As the sum of squares, there is a guaranteed minimum, though not necessarily a unique one. If we write this in vector notation, differentiate w.r.t β we get the so-called *normal equations*.

$$RSS(\beta) \; = \; (\vec{y} \, - \, \mathbf{X}\beta)^T \, (\vec{y} - \mathbf{X}\beta)$$

differentiating w.r.t β gives:

$$\mathbf{X}^T(\vec{y} \, - \, \mathbf{X}\beta) = 0$$

given non-singular $\mathbf{X}^T\mathbf{X}$:

$$\hat{\beta} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T \, \vec{y}. \tag{4}$$

A prediction can then be made of the values Y given our β and $\mathbf{X}$ with

$$\hat{Y} \; = \; \mathbf{X} \, \hat{\beta}. \tag{5}$$

In a naive way, this prediction can be compared to the reference output values, to see how well the model predicts the data. Our wish, however, is to predict data of the same stochastic "source" as our known data. As such, we could get arbitrarily like the known data by increasing the complexity of the polynomial we use to fit the input. This will, however cost us generality. Any data outside the data we use to specifically fit the model is unlikely to fall within the model's prediction. This bias is something we need to avoid.

To this effect, we can implement a sectioning of the available data so that we can use parts of the set to "train" our model and a separate part as a "test" or "validation". The meethod chosen is called *k-fold Cross Validation* (k-fold CV)

3 Results

The results and discussion of such
o Present results
o Give critical discussion of you work & place it in correct context
o Relate work to other calculations/studies
o Reader should be able to reproduce calculations should they wish to do so. All input variables should be properly explained.
o Make sure figures and tables contain enough information in their captions. Axis labels, etc. A reader should be able to get a first impression of the work by purely studying the figures and tables.

4 Conclusion

Conclusions and perspectives
o State main findings and interpretations
o Try as far as possible to present perspectives for future work.
o Try to discuss the pros and cons of the methods and possible improvements.

5 Appendix

any extra material
o Additional calculations used to validate code.
o Selected calculations. Can be listed with few comments.
o Listing of code if necessary.
Consider moving parts from methods to appendix. A webpage is also an appropriate place for a lot of this type of info.

o Always reference material you base your work on, either scientific articles/reports or books.
o Refer to articles as: name(s) of author(s), journal, volume(Bold), page and year in parenthesis.
o Refer to bookds as: name(s) of author(s), title of book, publisher, place and year, eventual page numbers.