Alaa Ayach A20317680

Problem Statement

In this assignment I am trying to implement parametric regression for both Single variable and Multivariate regression. I changed parameters such as data, type of the regression from linear to polynomial, amount of training and testing data, trying higher dimensions and trying iterative and kernel method.
I analyzed the differences comparing the different results

Proposed Solution

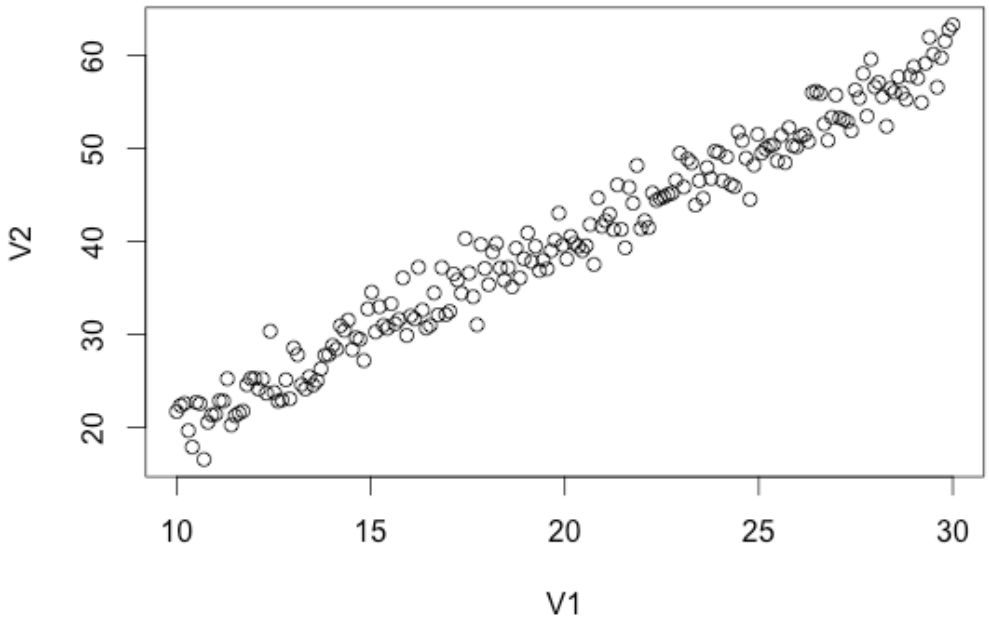
I implemented all functions from scratch using only matrix manipulation given by R. I used gradient descent as a way to implement the iterative solution.
I used RSE as an error measurement (for the first experiment) that could work and be understand for all datasets

Implementation Details

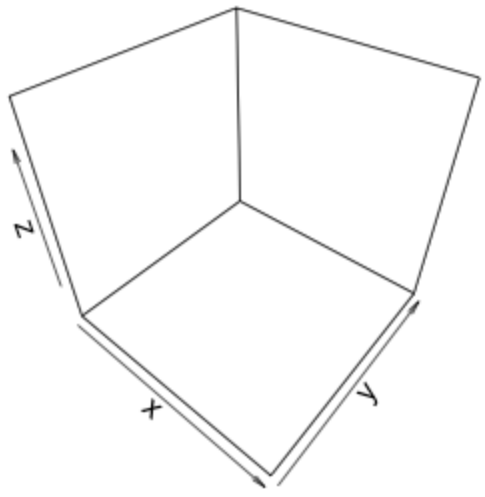
I encapsulate all the work in a modular way using files and parameters. Those files took the parameters that I talked about above.
There were some problem in computing G matrix so I used gausskernel function provided by the package "KRLS" which only can generate G for pair-wise Gaussian matrix by passing features array and Sigma.

Results and Discussion

Loading and plotting data
Example from single variable dataset

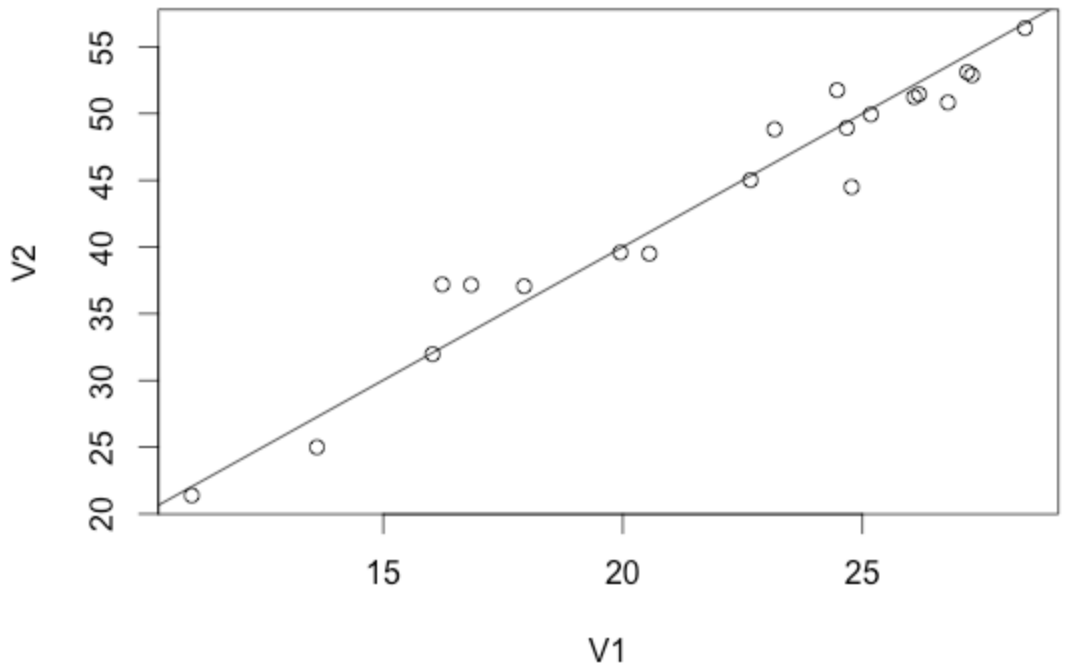


Example from the multivariate dataset



As a first look it seems that this single variable dataset can be fitted into a linear regression. However, the second one can't and probably need some higher level polynomial regression.

Linear Model Results



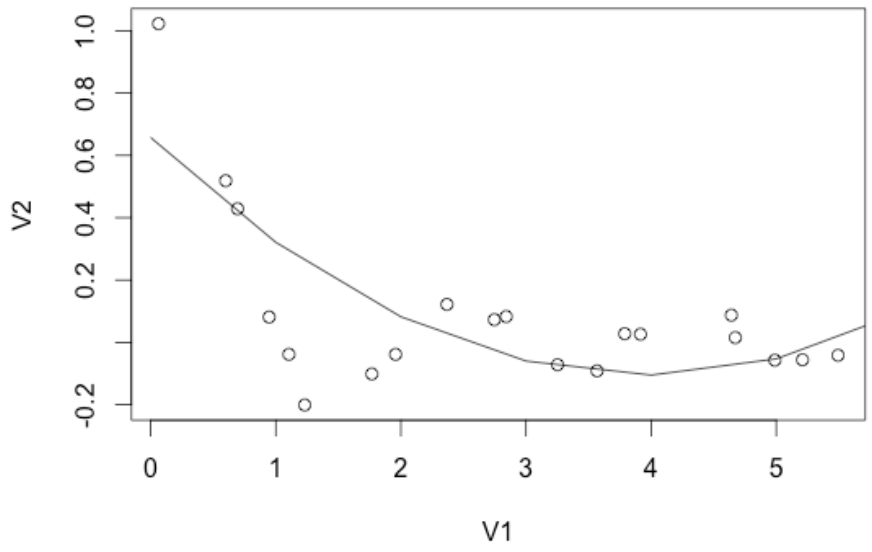
#Training error
[1] 0.02964582
> #Testing error
[1] 0.05389491

Here we see that we got a training error greater that the testing error as the model fits the training set.

Linear polynomial models for single data

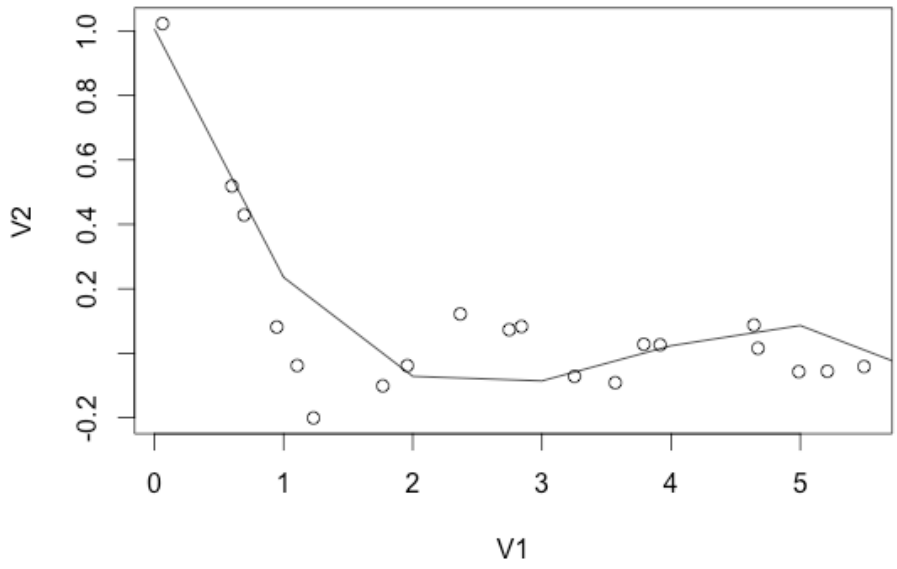
For other datasets, a linear model won't be efficient so we can use polynomial model as shown here:

3-dimensions



#Training error
[1] 5.434276
> #Testing error
[1] 5.777405

4-dimensions



#Training error
[1] 12.36575
> #Testing error
[1] 13.17785

We see here that the higher dimension overfits so it gives a greater error.

Reducing the amount of training data

When we reduce the amount of training data, the training error rate usually increase as the model became more poor. Here is reducing training data from 90% to 50%:
[1] 0.7266364
[1] 0.7330945
[1] 0.7565154
[1] 0.7556635
[1] 0.7876142

Multivariate polynomial higher dimension mapping then fitting the model

With extending dimentions from 2 to 6 I got the following error rate:
Error
[1] 3.670004

We noticed that the testing error differs a lot because the model is too overfitted

Iterative solution

I used gradient descent as a way to implement the iterative solution.
The error I got here is [1] 0.5945696
We noticed that it is way more smaller and that's because in the gradient descent we can reach a global optimal if we choose the appropriate parameters

Kernel method

There were some problem in computing G matrix so I used gausskernel function provided by the package "KRLS" which only can generate G for pair-wise Gaussian matrix by passing features array and Sigma.

I noticed here that the time performance for kernel was way too faster than the previous solutions (it was instant while it took about a minute for the iterative solution) that is because when we are using kernel we are reducing the unknown we have, however, the accuracy won't be good as the iterative one.

Existing implementation

It gave error rate [1] 1.509246 better than mine [1] 3.670004

Cross validation for linear single data

Using cross validation I got the following errors
[1] 0.6618496
[1] 1.042064
[1] 1.0308
[1] 0.7675229
[1] 0.7015682
[1] 0.7367857
[1] 1.38069
[1] 0.7920848
[1] 0.7014951
[1] 0.6848639

And when taking the average I could be comparing two models regardless what data wasn't seen by the model because the 10-fold will go through all the data as test data