Junyi CHEN

CS5487

2017/10/04

CS5487 Programming Assignment 1: regression

# PART 1 pOLYNOMINAL FUNCTION

## implementation

Implementation written in python is attached in the source code files.

## REGRESSION PLOTS AND HYPERPARAMETERS TUNNING

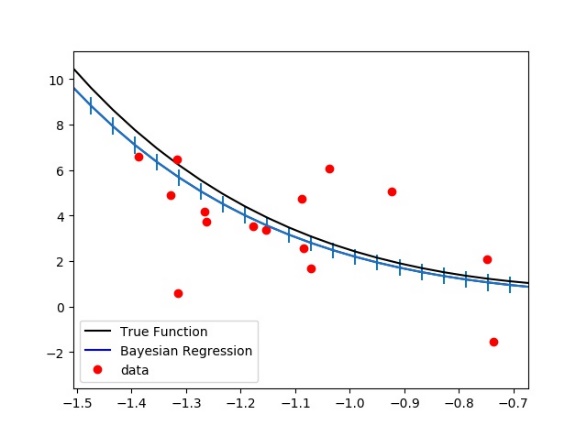
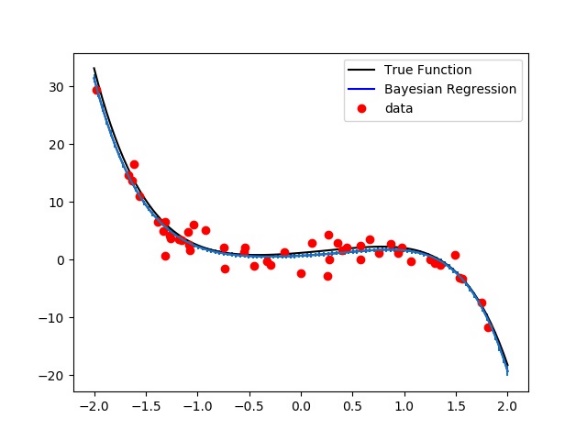
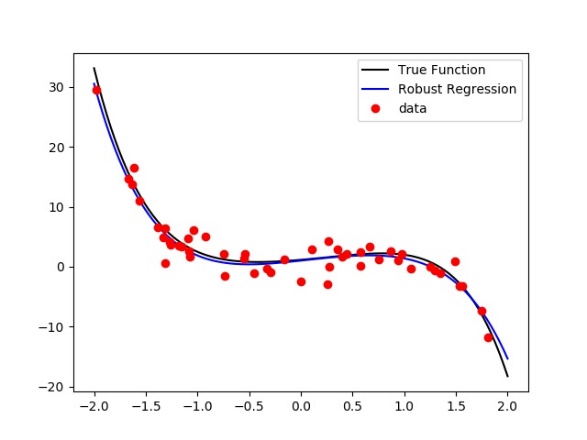
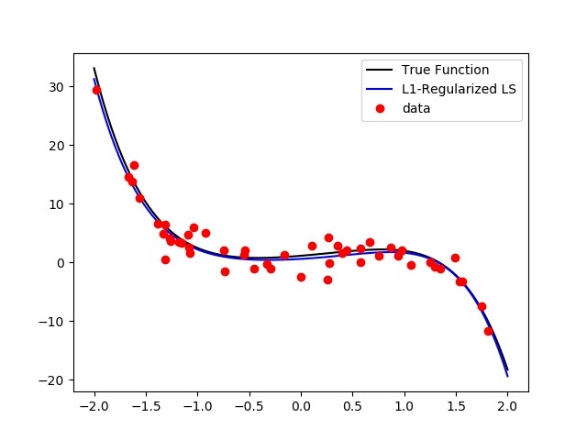
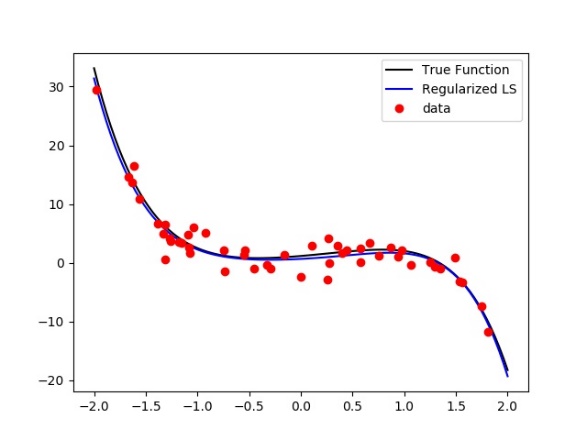
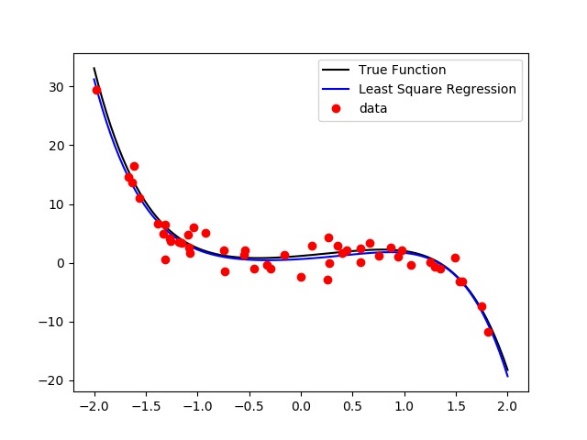


Figure 1: Plots of Different Regression Methods (figure in lower-right corner is the zoomed plot to show deviation of Bayesian Regression)

Figure 1 are plots of predictive functions of 5 different regression methods. Quadratic programming and linear programming solvers comes from the python package ‘cvxopt’. Hyperparameters (if any) are chosen by examining mean-square errors shown in Table 1. Values in bold type corresponds to ‘optimal hyperparameters’ in our experiments (noted that LS and RR required no hyperparameters). From the table, we can see that RLS (Regularized LS) and BR (Bayesian Regression) have the smallest MSE around 0.4076, a bit smaller than LS (Least-squares) and LASSO (L1-regularized LS) with 0.4086 and 0.4128 respectively while RR (Robust Regression) has a relatively larger number around 0.7680.

Table 1: Experiment mean-square errors of different hyperparameters

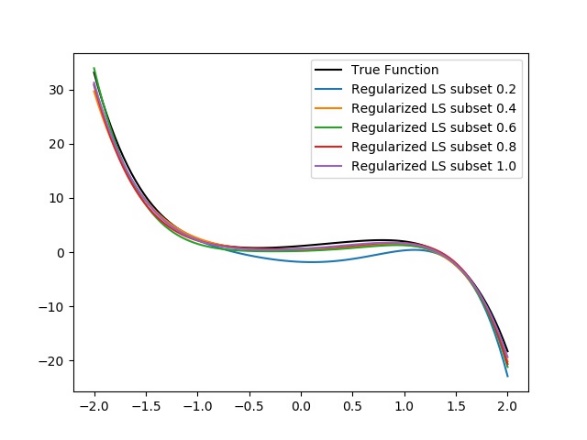
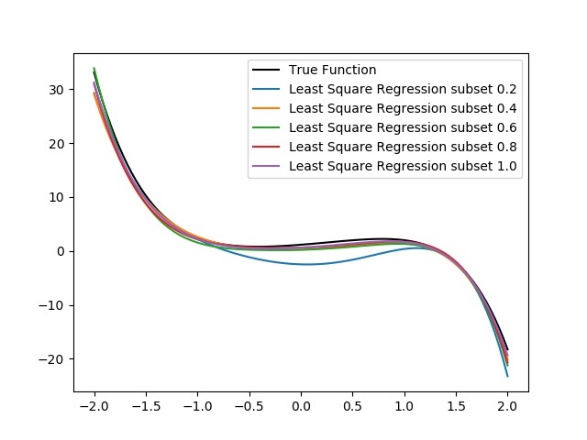
|  |  |  |
| --- | --- | --- |
|  | MSE of LS | MSE of RR |
| NA | 0.408644 | 0.768046 |

|  |  |
| --- | --- |
| Alpha and sigma | MSE of BR |
| 'alpha': 0.1, 'sigma': 0.1 | 0.408237 |
| 'alpha': 0.1, 'sigma': 0.5 | 0.420991 |
| 'alpha': 0.1, 'sigma': 1 | 0.557904 |
| 'alpha': 0.1, 'sigma': 5 | 2.895878 |
| 'alpha': 0.5, 'sigma': 0.1 | 0.408553 |
| 'alpha': 0.5, 'sigma': 0.5 | **0.4076** |
| 'alpha': 0.5, 'sigma': 1 | 0.415603 |
| 'alpha': 0.5, 'sigma': 5 | 1.238121 |
| 'alpha': 1, 'sigma': 0.1 | 0.408598 |
| 'alpha': 1, 'sigma': 0.5 | 0.407827 |
| 'alpha': 1, 'sigma': 1 | 0.408633 |
| 'alpha': 1, 'sigma': 5 | 0.856287 |
| 'alpha': 5, 'sigma': 0.1 | 0.408635 |
| 'alpha': 5, 'sigma': 0.5 | 0.408425 |
| 'alpha': 5, 'sigma': 1 | 0.407939 |
| 'alpha': 5, 'sigma': 5 | 0.459158 |

|  |  |  |
| --- | --- | --- |
| Lambda | MSE of RLS | MSE of LASSO |
| 0.1 | 0.408236518 | **0.412832** |
| 0.25 | 0.407826865 | 0.420168 |
| 0.5 | **0.407600224** | 0.434636 |
| 1 | 0.408632571 | 0.474607 |
| 2 | 0.415602545 | 0.519128 |
| 5 | 0.459157939 | 0.569842 |

## sample subsets and learning curves

In our experiment, subset sizes of samples are 20%, 40%, 60%, 80%. For each size of subset, we run 5 trials of different random subsets and take the average error. The sampling function comes from python package ‘scikit-learn’. On the other hand, we only plot the first-round function inside 10 same size trials. In this section, figure 2 include plots range from 20% to 80% of the full dataset using the hypermeters from the previous section, full set figure is not provided for it is identity to plots in part (b). Curves in different colors are prediction functions leanrned form different sizes of subset. Figure 3 are leraing curves indicate relations between mean square errors and size of subset. Range of y aixs is fixed, the part of curve with y value larger than 50 will not be displayed.



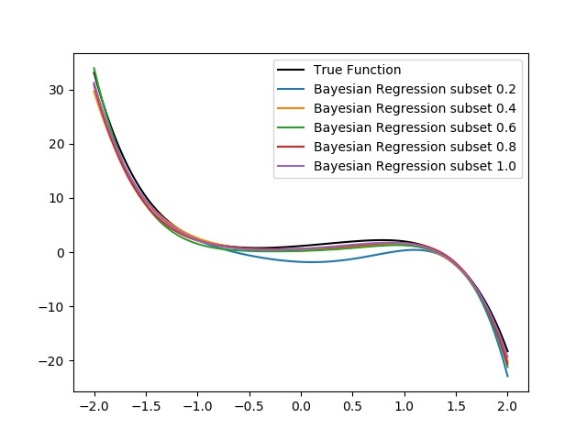
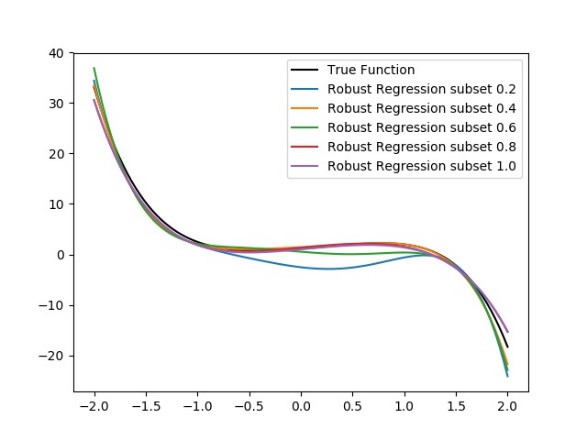
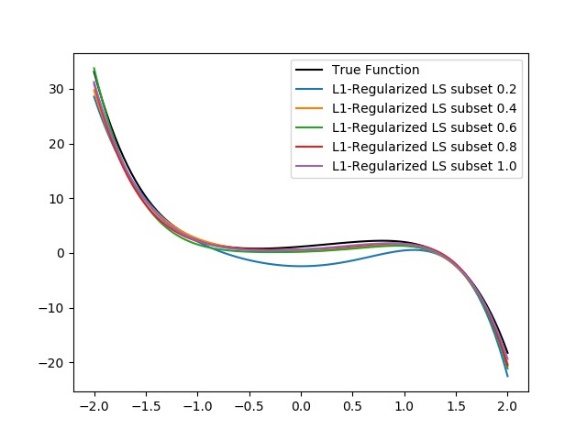


Figure Prediction functions and true functions of different regression methods

From two set of figures, we can be inferred that RLS and BR have better regression performances when the dataset is small, for they have intuitively ‘closer’ line in the prediction function plots and much smaller MSE indicated by the learning curves. Besides, though LASSO have an MSE excesses the limit of 50 when we train the model using 20% of data, it is much better than LS and RR whose MSE are around 500 and 600 respectively. This experiment may indicate that RLS, LASSO and BR have better resistances against overfitting

