CS5487 Machine Learning - Programming Assignment 1

Zhanghan Ke (55460880)

zhanghake2-c@my.cityu.edu.hk

Abstract. This is the report for my Programming Assignment 1 of CS5487 Machine Learning. The code also available in my github. Link: https://github.com/ZHKKKe/PA1.git.

1 Polynomial Function

1.1 Problem (a)

I implemented all 5 regression algorithms for the K-th order polynomial. The code of them in folder 'PA1/src'.

1.2 Problem (b)

The Fig. 1 (a) shows my results of this problem. The mean-squared error (MSE) is shown in the picture's legend. The hyperparameters setting in Table 1.

1.3 Problem (c)

The Fig. 2 shows my results averaged by 5 runs of each algorithms under 10%, 25%, 50% and 75% data. The Least Squares is most sensitive for the reduce of data and the Bayesian Regression is most robust for small data size. But with the increase of data size, the Least Squares could fit data will but the variance of Bayesian Regression reduce slowly.

1.4 Problem (d)

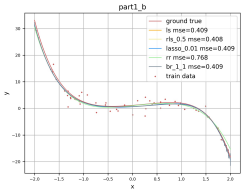
The Fig. 1 (b) shows my results of add one outlier output value $[11, -9851589]$. The MSE value shows that the Regularized LS is most sensitive for the outlier and the L1-Regularized LS is most robust for it. The reason is the L1-Regularized could reduce the influence of outlier by the extra term in it.

1.5 Problem (e)

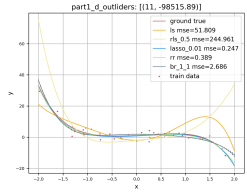
The Fig. 1 (c) shows the results with order=10. The Regularized LS overfits data most and the Robust Regression do not. The Robust Regression use 1-norm so it is not easy to overfit the data when the data is not enough for the algorithm.

Table 1. Hyperparameters settings of polynomial function

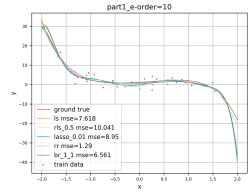
Algorithm	**Hyperparameters**
Least Squares (LS)	-
Regularized LS (RLS)	$\alpha = 0.5$
L1-Regularized LS (LASSO)	$\alpha = 0.01$
Robust Regression (RR)	-
Bayesian Regression (BR)	$\alpha = 1, \sigma = 1$



(a) 1-(b) curves

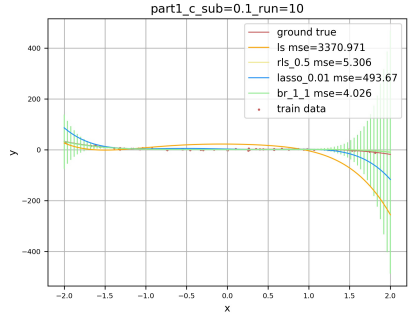


(b) 1-(d) curves

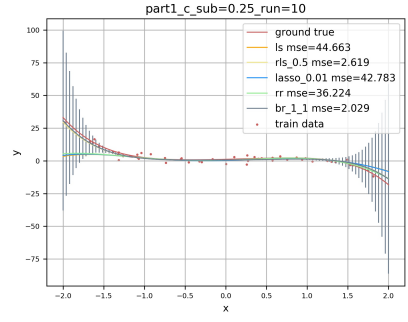


(c) 1-(e) curves

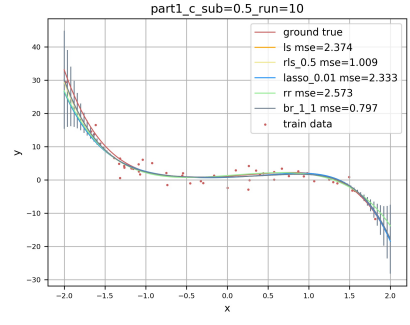
Fig. 1. Result curves of polynomial function (b) (d) (e).



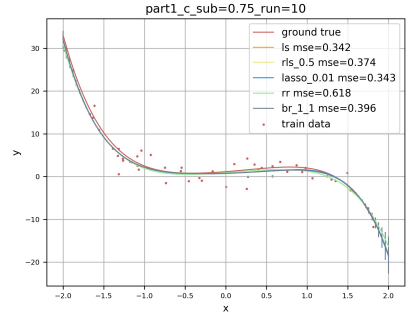
(a) 1-(c) 10% data



(b) 1-(c) 25% data



(c) 1-(c) 50% data



(d) 1-(c) 75% data

Fig. 2. Result curves of polynomial function (b) (d) (e).

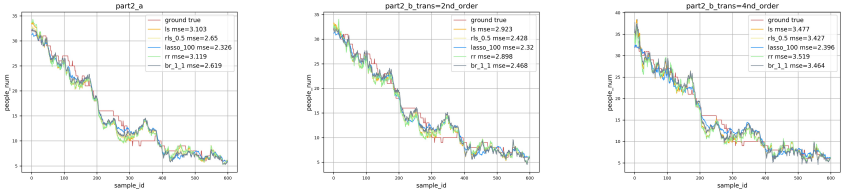
2 A Real World Regression Problem Counting People

2.1 Problem (a)

The Fig. 3 (a) shows my results. From the picture, we can see that the L1-Regularized work best. From the picture I also find that all algorithms output similar results, and all of them cannot fit the ground true well.

2.2 Problem (b)

I try the 2nd order polynomial and 4nd polynomial in this problem. The 2nd order could improve the result of algorithms, but 4nd will make the bad results. I think such bad results caused by overfitting. Fig. 3 (b),(c) shows the results.



(a) 2-(a) curves (b) 2-(b) 2nd curves (c) 2-(b) 4nd curves

Fig. 3. Result curves of counting people.

3 Estimating Hyperparameters

I implemented N-fold cross-validation, and auto-search hyperparameters by it with folder=3. (code in 'src/algorithm.py/n_folder_cross_validation') Fig. 4 compares the results. The MSE shows the decrease of accuracy. The hyperparameters is very different with my hand-selected ones. I think the auto-searched hyperparameters overfit to the training data, so it not works well in test data.

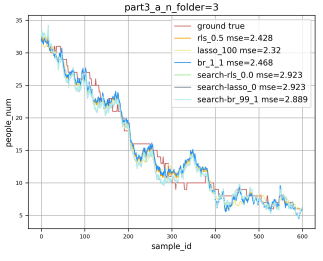


Fig. 4. Comparing auto-search and hand-selected hyperparameters for counting people.