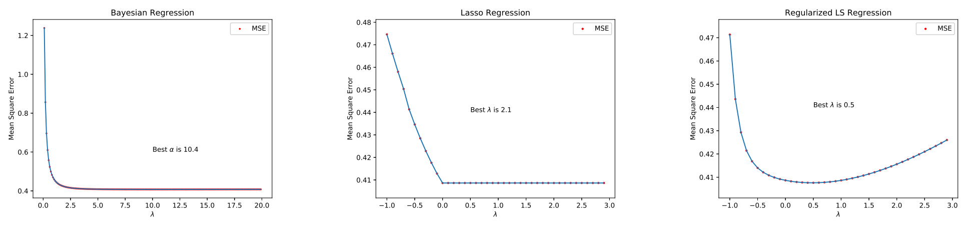
CS5487 Programming Assignment 1 Report

LIN Jiecong (54938833)

October 10, 2017

1 Polynomial function

1.1 Well-working Hyperparameter Selection



(a) Bayesian Regression (b) Lasso Regression (c) Regularized LS Regression

Figure 1: Mean Squres Error Based on Different Hyperparameters

For each regression method, experiments used the sample data (*samplx, sampley*) to estimate the parameters of a 5th order polynomial function. Firstly, I have done several experiments to select best hyperparameters for BR, RLS and LASSO.

For both RLS and LASSO, I used $\lambda \in [-1.0, 3.0]$ for regression based on sample data (*samplx, sampley*), then using *polyx* to predict the output and calculating the mean-squared error. As for Bayesian Regression, testing α was generated from $[0.1, 20.0]$. The experiment results (see figure 1)show that the best hyperparameters of three regression model are $\lambda = 0.5$(Regularized LS Regression), $\lambda = 2.1$(Lasso Regression) and $\alpha = 10.4$(Bayesian Regression). In the following experiment, these three best hyperparameters will be used for regression.

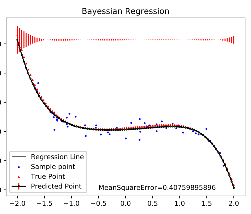
1.2 Regression and Prediction on Original Samples

In this part, I implemented 5 regression algorithm for the 5th order polynomial. For each regression method, use the sample data (*sampx, sampy*) to estimate the parameters of a 5th order polynomial function. Figure 2 shows the estimated function using polyx as inputs, along with the sample data. For BR, I also plot the standard deviation around the mean (see Figure 2a).

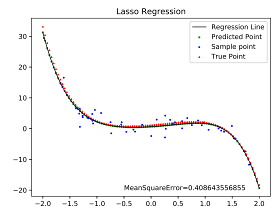
Table 1: Mean Square Error

Algorithm	MSE
Least Square	0.4086
Regularized LS	0.4076
Lasso	0.4086
Robust	0.7680
Bayesian	0.4075

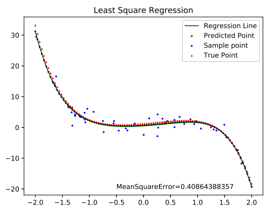
From the experiment we can see that the regression lines generated by 5 algorithms can well fit the sample data, among them Bayesian Regression gains the least MSE Score (MSE=0.4075) and Robust Regression gains the larger one (MSE=0.768). LS, RLS and BR achieved almost the same performance while Robust Regression has the worst performance on sample data apparently.



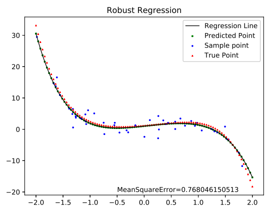
(a) Bayesian Regression



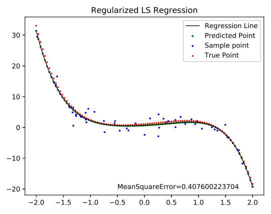
(b) Lasso Regression



(c) Least Square Regression



(d) Robust Regression

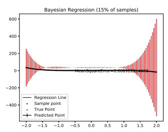


(e) Regularized LS Regression

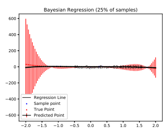
Figure 2: Regression and Prediction of 5 algorithms

1.3 Regression and Prediction on Subset of Samples

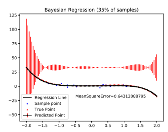
In this part, my experiment reduces the amount of training data available by selecting some subsets of the sample. I generated 9 kinds of subsets whose size ranges from 15% to 95%, then I ran experiment 2000 times on every single subset and calculated the average MSE.



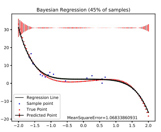
(a) 15%



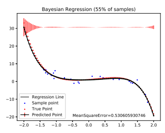
(b) 25%



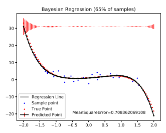
(c) 35%



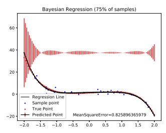
(d) 45%



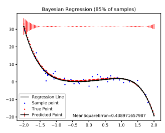
(e) 45%



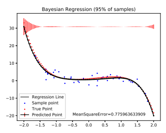
(f) 65%



(g) 75%

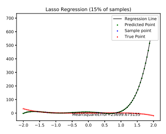


(h) 85%

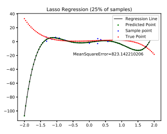


(i) 95%

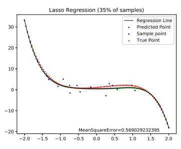
Figure 3: Bayesian Regression



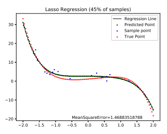
(a) 15%



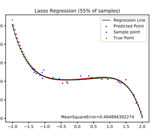
(b) 25%



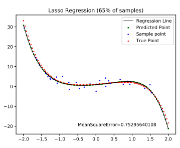
(c) 35%



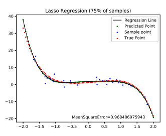
(d) 45%



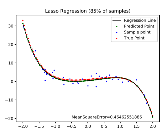
(e) 45%s



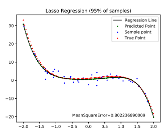
(f) 65%



(g) 75%

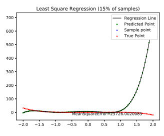


(h) 85%

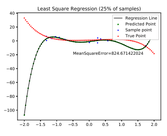


(i) 95%

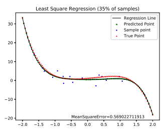
Figure 4: Lasso Regression



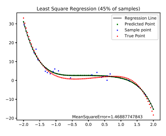
(a) 15%



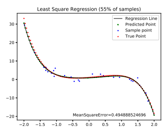
(b) 25%



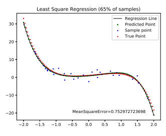
(c) 35%



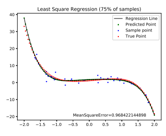
(d) 45%



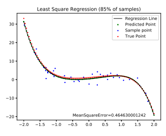
(e) 55%



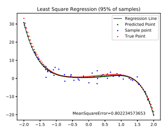
(f) 65%



(g) 75%

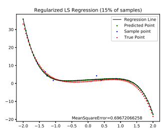


(h) 85%

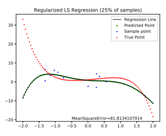


(i) 95%

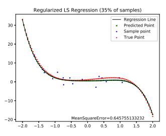
Figure 5: Least Square Regression



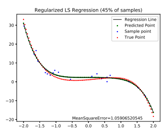
(a) 15%



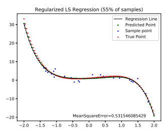
(b) 25%



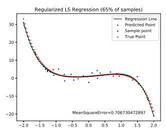
(c) 35%



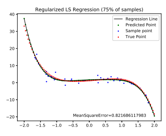
(d) 45%



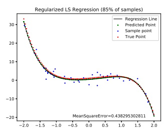
(e) 55%



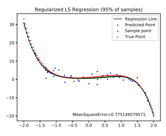
(f) 65%



(g) 75%

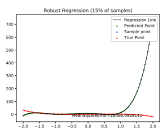


(h) 85%

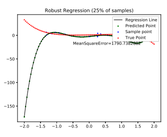


(i) 95%

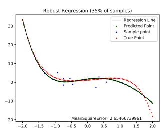
Figure 6: Regularized LS Regression



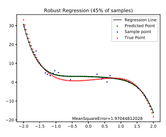
(a) 15%



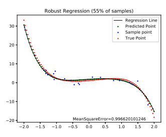
(b) 25%



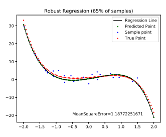
(c) 35%



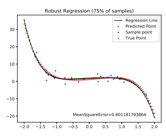
(d) 45%



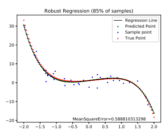
(e) 55%



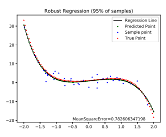
(f) 65%



(g) 75%



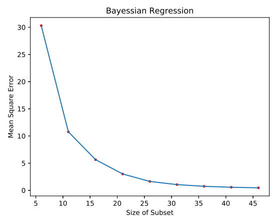
(h) 85%



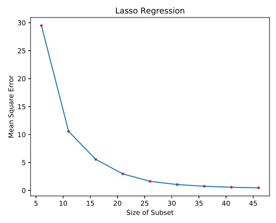
(i) 95%

Figure 7: Robust Regression

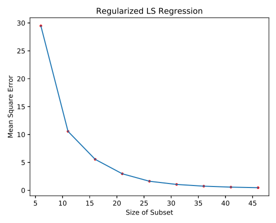
Figure 8 on the above shows the average error in 2000 times versus training size; as we can see from the figure, Least-squares Regression and Robust Regression more robust with less data (only around 20% of the sample) while the other tend to overfit badly when the size of sample is less 50%.



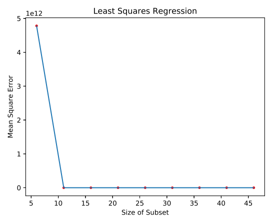
(a) Bayesian Regression



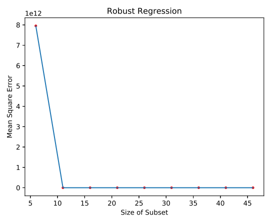
(b) Lasso Regression



(c) Regularized LS Regression



(d) Least Square Regression



(e) Robust Regression

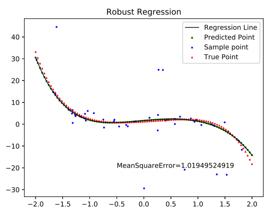
Figure 8: Mean-squared Error versus Training Size

1.4 Regression and Prediction on Samples with Outliers

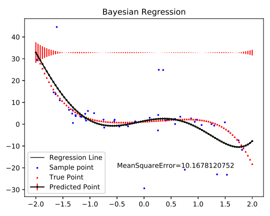
In this experiment, I added a random integer (can be negative or positive) with range from 20 to 30 to 15% of training samples. Figure 9 shows that Robust Regression are robust to the presence of outliers because whose MSE value is merely 1 while the MSEs of the other 4 algorithms reach more than 10. Among 5 algorithm, Least Square Regression whose MSE are the largest are most sensitive to the outliers.

Table 2: Mean Square Error on Sample with Outliers

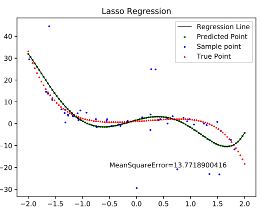
Algorithm	MSE
Least Square	13.7723
Regularized LS	10.0730
Lasso	13.7718
Robust	1.01940
Bayesian	10.1678



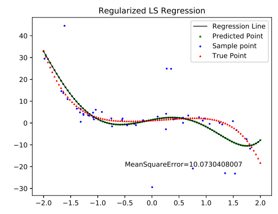
(a) Robust Regression



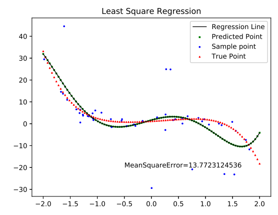
(b) Bayesian Regression



(c) Lasso Regression



(d) Regularized LS Regression



(e) Least Square Regression

Figure 9: Regression and Prediction on Sample with Outliers

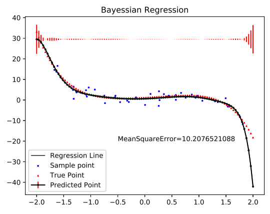
1.5 Higher-order Polynomial Estimation

In this section, I used the sample data (sampx, sampy) to estimate the parameters of a 10th order polynomial function for each regression method.

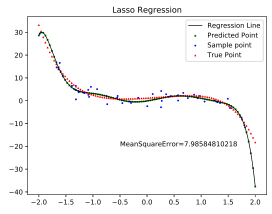
Table 3: Mean Square Error on 10th order

Algorithm	MSE
Least Square	13.7723
Regularized LS	10.0730
Lasso	13.7718
Robust	1.01940
Bayesian	10.1678

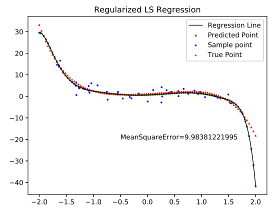
The Table 3 shows that Robust Regression has the least mean-squared error, however in the figure it seems that RR is well-fitting than the other models. The reason why RR has the best MSE is that the other 4 models overfitted the tail of samples (polyx > 1.5) badly which cause the average bad performance on the whole sample. Also figure shows the prediction of Bayesian Regression is the most overfitting with the highest mean-squared error that results from the worst overfitting on the tail of the sample.



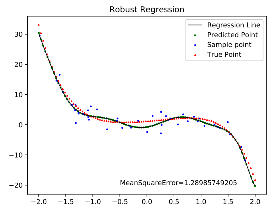
(a) Bayesian Regression



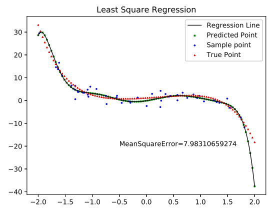
(b) Lasso Regression



(c) Regularized LS Regression



(d) Robust Regression



(e) Least Square Regression

Figure 10: Regression and Prediction on Sample with Outliers

1.6 Conclusion

After the above experiments, I found that among 5 algorithm Robust Regression had the worst performance in the texting though, it achieved the best performance in the experiments with less training data or adding outliers, which means that Robust Regression is indeed more "Robust" than other method and has less chance to be overfitting. On the other hand, I found that the performance of Lasso and Least-squares Regression are easy to be disturbed by outliers and less training data, which means they have the high risk to be overfitting in bad training dataset.

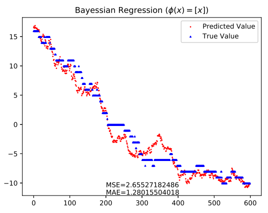
2 A real world regression problem - counting people

2.1 Number of People Prediction on Original Feature

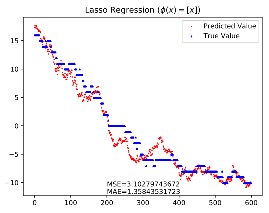
In this section, I use the feature directly (set $\phi(x) = x$). Table 4 and Figure 11 show that Regularized LS Regression works the best with the lowest mean-absolute error (MAE=1.2794) and mean-squared error (MSE=2.6503).

Table 4: Error on People Counting Prediction using Original Features

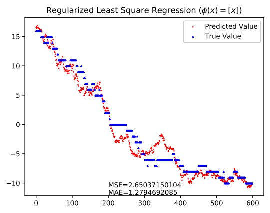
Algorithm	MSE	MAE
Least Square	3.1028	1.3584
Regularized LS	2.6503	1.2794
Lasso	3.1027	1.3584
Robust	3.1189	1.3645
Bayesian	2.6552	1.2801



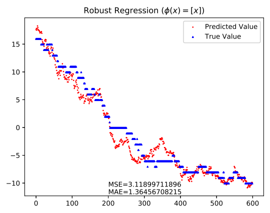
(a) Bayesian Regression



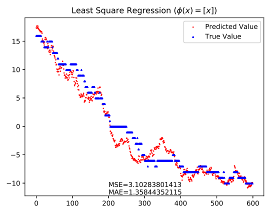
(b) Lasso Regression



(c) Regularized LS Regression



(d) Robust Regression



(e) Least Square Regression

Figure 11: People Counting Prediction on Original Features

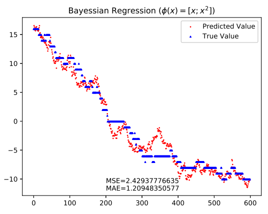
2.2 Number of People Prediction on Transformed Feature

2.2.1 2nd Order Polynomial Transformed Feature

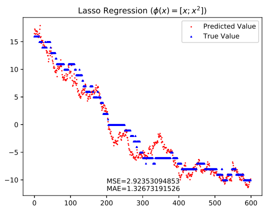
Here I created a simple 2nd order polynomial as $\phi(x) = [x_1, \ldots, x_9, x_1^2, \ldots, x_9^2]$. Table 6 and Figure 13 show that all 5 regression model achieved better performance with the 2nd order polynomial transformed feature and Regularized LS Regression still works the best.

Table 5: Error on People Counting Prediction using 2nd Order Polynomial Features

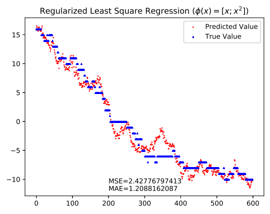
Algorithm	MSE	MAE
Least Square	2.9235	1.3267
Regularized LS	2.4277	1.2088
Lasso	2.9235	1.3267
Robust	2.8981	1.3079
Bayesian	2.4293	1.2094



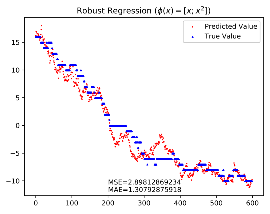
(a) Bayesian Regression



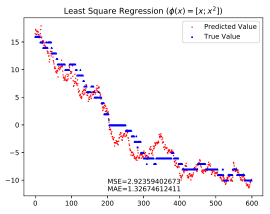
(b) Lasso Regression



(c) Regularized LS Regression



(d) Robust Regression



(e) Least Square Regression

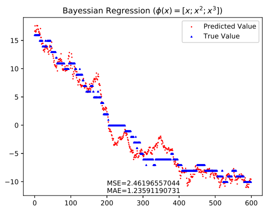
Figure 12: People Counting Prediction on 2nd Order Polynomial Features

2.2.2 2nd & 3rd Order Polynomial Transformed Feature

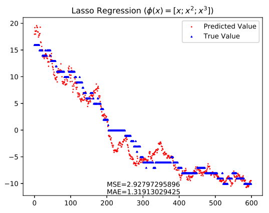
Here I created a combination with 2nd order polynomial and 3rd order polynomial as $\phi(x) = [x_1, \ldots, x_9, x_1^2, \ldots, x_9^2, x_1^3, \ldots, x_9^3]$. Table 6 for people counting prediction. Figure 13 show that only Bayesian Regression achieved obviously better performance. As a result, Bayesian Regression merely works best among 5 methods.

Table 6: Error on People Counting Prediction using 2nd Order Polynomial Features

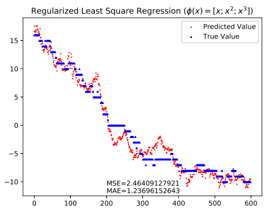
Algorithm	MSE	MAE
Least Square	2.9282	1.3191
Regularized LS	2.4640	1.2369
Lasso	2.9235	1.3267
Robust	3.1116	1.3517
Bayesian	2.4619	1.2359



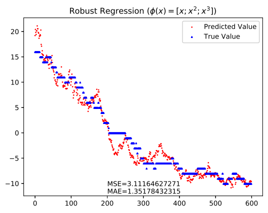
(a) Bayesian Regression



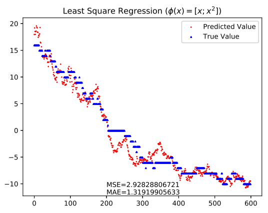
(b) Lasso Regression



(c) Regularized LS Regression



(d) Robust Regression



(e) Least Square Regression

Figure 13: People Counting Prediction on 2nd & 3rd Order Polynomial Features

2.3 Conclusion

Regularized LS Regression achieved the best performance on people counting prediction. With 2nd Order and 2nd Order& 3rd Order Polynomial feature transformation, all 5 regression methods achieved better performance than using original feature. Under 3rd Order Polynomial feature transformation, Bayesian Regression had highest effect enhancement.