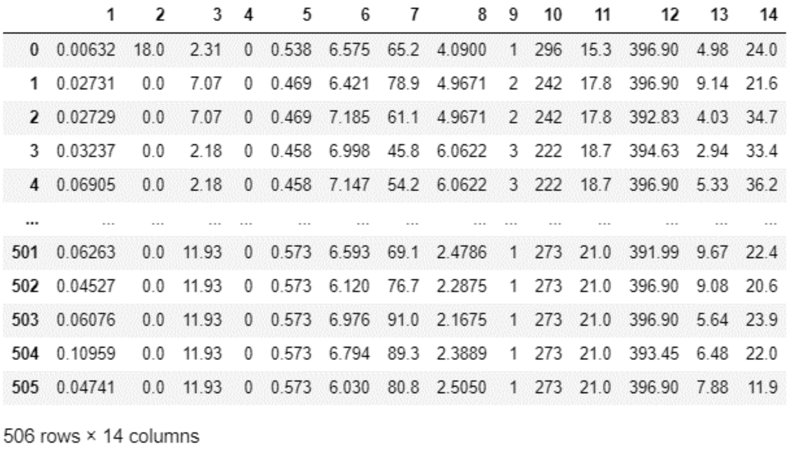
1. Solution

1.1 Housing

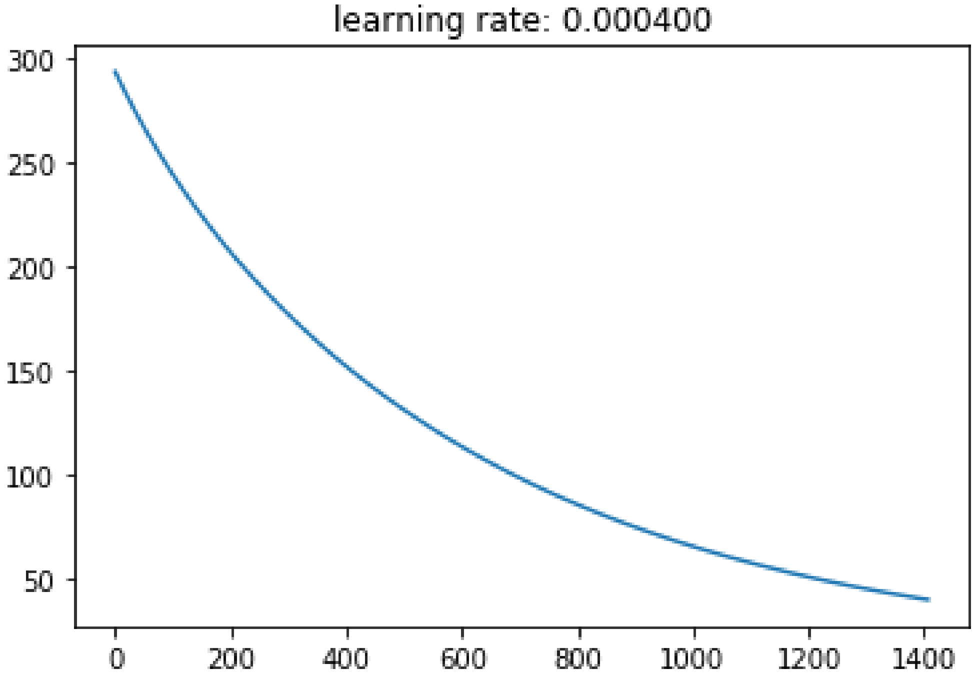


	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	36.2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
501	0.06263	0.0	11.93	0	0.573	6.593	69.1	2.4786	1	273	21.0	391.99	9.67	22.4
502	0.04527	0.0	11.93	0	0.573	6.120	76.7	2.2875	1	273	21.0	396.90	9.08	20.6
503	0.06076	0.0	11.93	0	0.573	6.976	91.0	2.1675	1	273	21.0	396.90	5.64	23.9
504	0.10959	0.0	11.93	0	0.573	6.794	89.3	2.3889	1	273	21.0	393.45	6.48	22.0
505	0.04741	0.0	11.93	0	0.573	6.030	80.8	2.5050	1	273	21.0	396.90	7.88	11.9

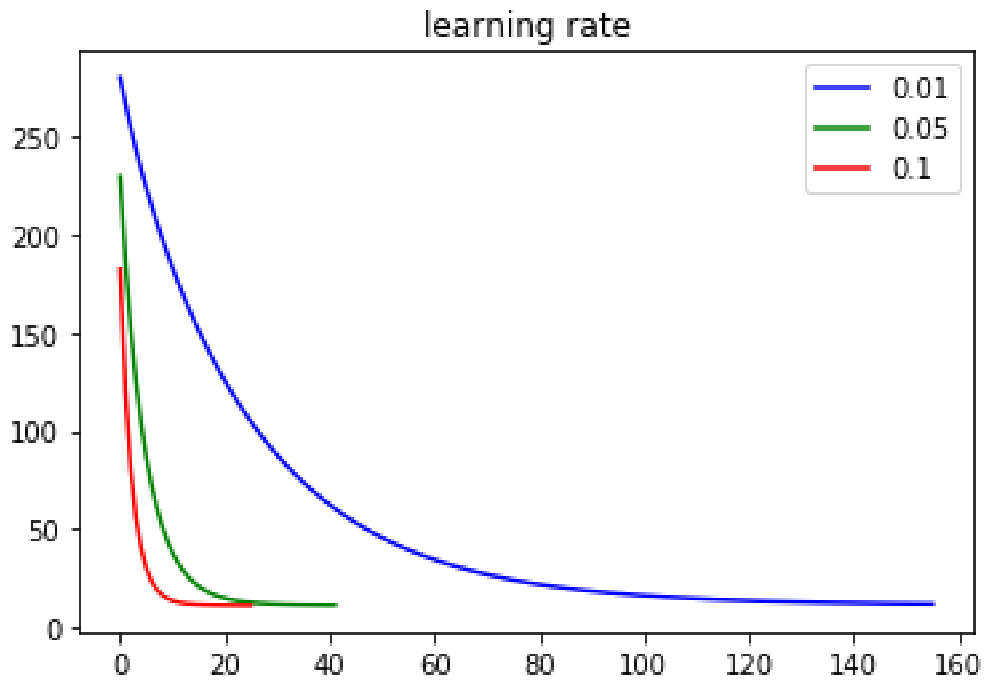
506 rows × 14 columns

Through normalizing features, constructing loss function, selecting suitable learning rate and calculating the weights through Gradient Descent, we get weights, value of loss function for each iteration and iteration number.

Firstly, we use the given parameters to find the results (learning rate=0.4×10−3, tolerance=0.5×10−2, maximum iterations = 50000). The process of convergence for gradient descent is shown as below:



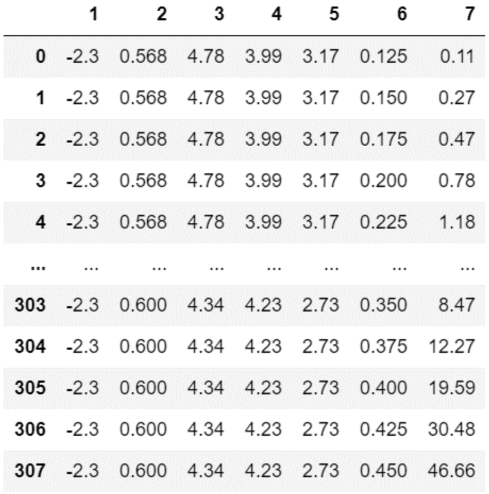
Then we experiment with different learning rates such as 0.1,0.01,0.05 and plot the process as well to compare the results.



Thirdly, we use the theta returned by given parameters (learning rate=0.4×10−3, tolerance=0.5×10−2, maximum iterations = 50000) to predict the y for test dataset. The RMSE is shown as 9.06.

To compare the performance of Gradient Descent and Normal Equation, we also use the function of weights by normal equation to find the regression weights. The RMSE is shown as 4.09.

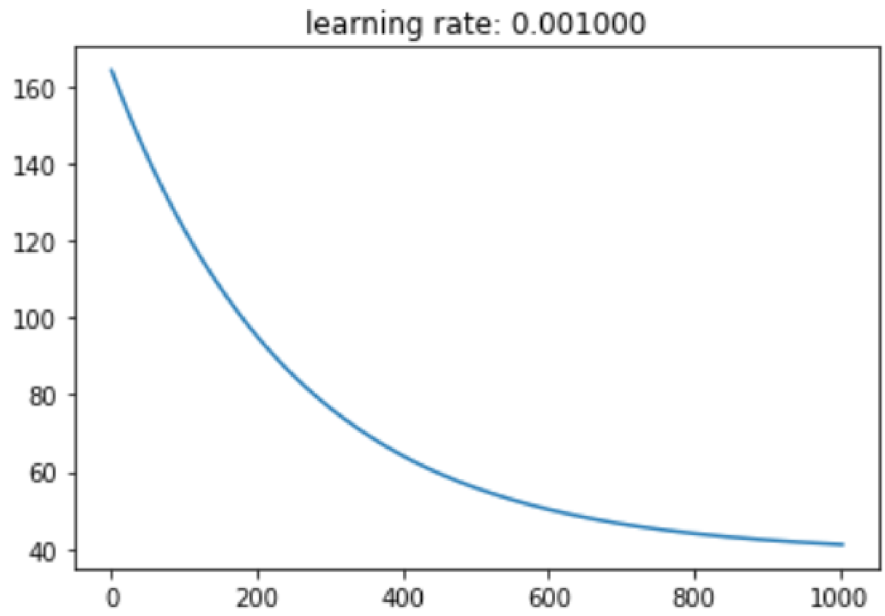
1.2 Yacht



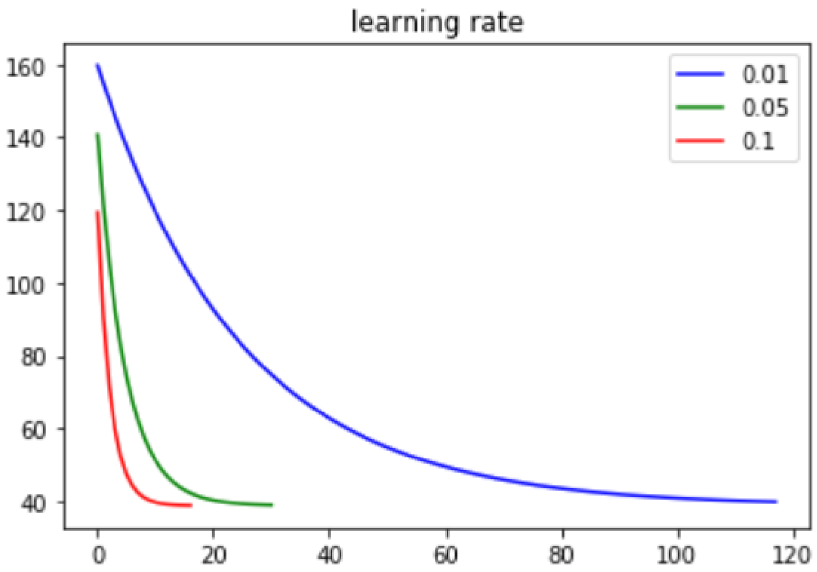
	1	2	3	4	5	6	7
0	-2.3	0.568	4.78	3.99	3.17	0.125	0.11
1	-2.3	0.568	4.78	3.99	3.17	0.150	0.27
2	-2.3	0.568	4.78	3.99	3.17	0.175	0.47
3	-2.3	0.568	4.78	3.99	3.17	0.200	0.78
4	-2.3	0.568	4.78	3.99	3.17	0.225	1.18
...	...	...	...	...	...	...	...
303	-2.3	0.600	4.34	4.23	2.73	0.350	8.47
304	-2.3	0.600	4.34	4.23	2.73	0.375	12.27
305	-2.3	0.600	4.34	4.23	2.73	0.400	19.59
306	-2.3	0.600	4.34	4.23	2.73	0.425	30.48
307	-2.3	0.600	4.34	4.23	2.73	0.450	46.66

we get weights, and value of loss function for each iteration and the iteration number by normalizing features, constructing loss function, selecting suitable learning rate and calculating the weights through Gradient Descent.

Firstly, we use the given parameters to find the results (learning rate=0.1 × 10−2, tolerance=0.1 × 10−2, maximum iterations = 50000). The process of convergence for gradient descent is shown as below:



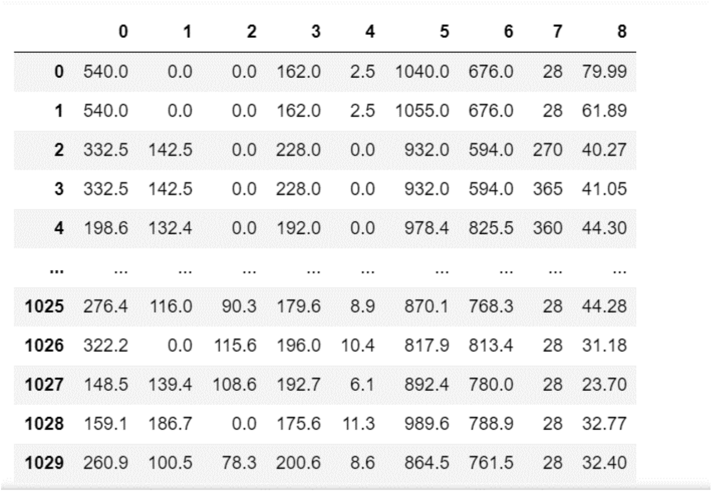
Then we experiment with different learning rates such as 0.1,0.01,0.05 and plot the process as well to compare the results.



Thirdly, we use the Theta returned by given parameters learning rate=0.1 × 10−2, tolerance=0.1 × 10−2, maximum iterations = 50000) to predict the y for test dataset. The RMSE is shown as 9.60.

To compare the performance of Gradient Descent and Normal Equation, we also use the function of weights by normal equation to find the regression weights. The RMSE decrease to 8.72.

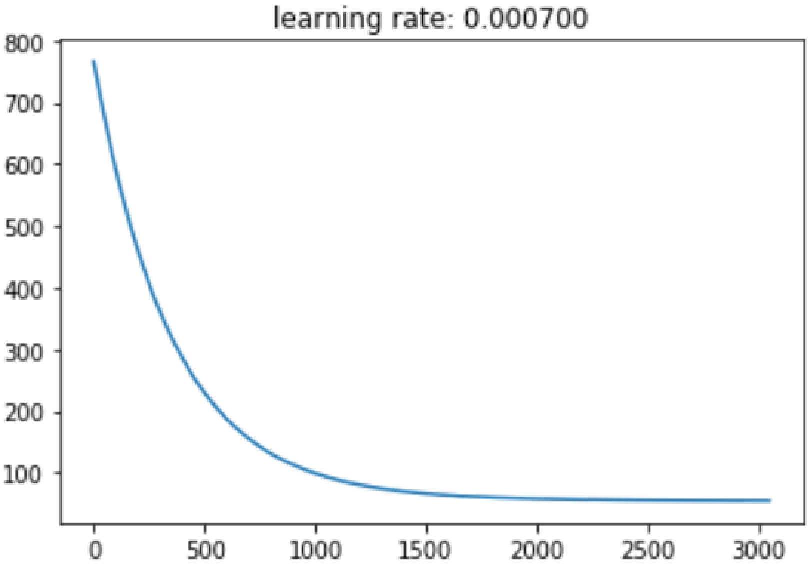
1.3 Concrete



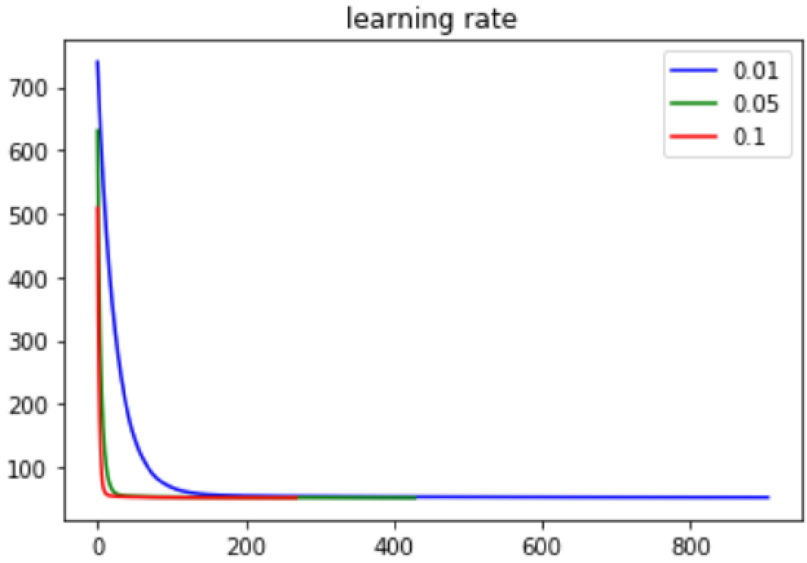
	0	1	2	3	4	5	6	7	8
0	540.0	0.0	0.0	162.0	2.5	1040.0	676.0	28	79.99
1	540.0	0.0	0.0	162.0	2.5	1055.0	676.0	28	61.89
2	332.5	142.5	0.0	228.0	0.0	932.0	594.0	270	40.27
3	332.5	142.5	0.0	228.0	0.0	932.0	594.0	365	41.05
4	198.6	132.4	0.0	192.0	0.0	978.4	825.5	360	44.30
...	...	...	...	...	...	...	...	...	...
1025	276.4	116.0	90.3	179.6	8.9	870.1	768.3	28	44.28
1026	322.2	0.0	115.6	196.0	10.4	817.9	813.4	28	31.18
1027	148.5	139.4	108.6	192.7	6.1	892.4	780.0	28	23.70
1028	159.1	186.7	0.0	175.6	11.3	989.6	788.9	28	32.77
1029	260.9	100.5	78.3	200.6	8.6	864.5	761.5	28	32.40

Through normalizing features, constructing loss function, selecting suitable learning rate and calculating the weights by Gradient Descent, we get weights, value of loss function for each iteration and iteration number.

Firstly, we use the given parameters to find the results (learning rate=0.7 × 10−3 , tolerance=0.1 × 10−3, maximum iterations = 50000). The process of convergence for gradient descent is shown as below:



Then we experiment with different learning rates such as 0.1,0.01,0.05 and plot the process as well to compare the results.



Thirdly, we use the theta returned by given parameters (learning rate=0.7 × 10−3, tolerance=0.1 × 10−3, maximum iterations = 50000) to predict the y for test dataset. The RMSE is shown as 11.32.

Additionally, we also calculate the performance of Normal Equation, we use the function of weights by normal equation to find the regression weights. The RMSE is shown as 10.67.

2. Analysis

2.1 Comparison of learning rate

Based on the comparison of the process of gradient descent with different learning rate and by using different dataset, it can be deducted:

1. By comparing the results of different learning rates, we can see that the larger the learning rate is, the faster the lost function be close to convergence. Only ideal learning rate can keep successful iteration and keep loss function decrease.
2. According to the experience, if the learning rate is sufficiently small， the value of loss function should decrease every iteration and maintain the normal speed.
3. According to the given learning rate and the result of gradient descent, it seems that we should use smaller learning rate when the sample number become larger.

2.2 Comparison of performance by Gradient Descent and Normal Equation

According to the RMSE got in each dataset, it is shown that RMSE by Normal Equation is always smaller than that by Gradient Descent. We deduce that Normal Equation will perform better than Gradient Descent in most cases. It seems that we should choose Normal Equation other than Gradient Descent. However, based on the experiential knowledge, we should use Gradient Descent when the sample number of data is very large due to that we should compute the weights through finding an inverse matrix which will become very slow for a large number sample.

Gradient Descent works well even when number of a dataset is very large. Meanwhile, it can be applied in Non-linear regression. But if the dataset is small and can use linear regression to fit, we can choose normal equation to find regression weights directly.