CS534 Implementation Assignment 1

October 2019

Introduction

In this assignment, we used linear regression to predict the housing prices, based on features of the house and the timing of the sale. The full list of features is the number of bedrooms, the number of bathrooms, the square footage of the living area, the square footage of the lot, the number of floors, whether the house is on the waterfront, how many times the house has been viewed, the condition of the house, the grade of the house, the square footage of the basement, the square footage above ground, the year the house was built, the year the house was renovated, the zip code, the latitude, the longitude, the square footage of the living room in 2015, the square footage of the lot in 2015, and the date of the sale.

We trained our linear regression model using the following loss function:

$$\frac{1}{2N}\sum_{i=1}^{N}(y_i - \mathbf{w}^T\mathbf{x}_i)^2 + \lambda||\mathbf{w}||^2$$

We found that this loss function converges for more of the learning rates being tested than the non-averaged version. We expected this to be the case, because there are 10,000 samples, so using a learning rate of l with the MSE is equivalent to using a learning rate of $10^{-4}l$ with the normal SSE. We report the MSE in all of our data. Weights for each regression are initialized to be randomly between -1 and 1.

This report is organized as follows. Part 0 contains the responses to questions about data preprocessing and our predictions about which features of the data are most important. Part 1 compares models trained with different learning rates and without regularization. Part 2 compares regularization with different λ values for a fixed learning rate. Part 3 explores the effects of using non-normalized data.

Part 0: Preprocessing and Simple Analysis

This section analyzes the data prior to training and details our predictions about the data.

(a) The numeric value of ID is not a meaningful feature of the data since it solely serves as an identifier for each sale. Meaningless features make training more difficult, and as such we should not include ID in the regression.

(b) We do not expect the month and day features to be useful, because they are cyclic rather than linear. For example, a month of 1 and 12, i.e. January and December, are far apart numerically, but they are adjacent to each other on the calendar. Year, however, could potentially be useful as a proxy for inflation.

(c) The mean, standard deviation, and range for each numerical feature is reported in Table 1 for. For the categorical features, the following percentage of samples are in each category:

- Waterfront: 99.3% 0, 0.7% 1
- View: 90.3% 0, 1.6% 1, 4.2% 2, 2.5% 3, 1.3% 4
- Condition: 0.1% 1, 0.8% 2, 65.3% 3, 25.7% 4, 8.1% 5
- Grade: 0.1% 4, 1.1% 5, 9.3% 6, 41.3% 7, 28.4% 8, 11.8% 9, 5.5% 10, 2.1% 11, 0.4% 12, 0.1% 13

Stat	Mean	Stdev	Range
dummy	1.00	0.00	0.00
bedrooms	3.38	0.94	32.00
bathrooms	2.12	0.77	7.25
sqft_living	2080.22	911.33	9520.00
sqft_lot	15089.20	41203.89	1650787.00
floors	1.50	0.54	2.50
waterfront	0.01	0.08	1.00
view	0.23	0.76	4.00
condition	3.41	0.65	4.00
grade	7.67	1.18	9.00
sqft_above	1793.10	830.87	8490.00
sqft_basement	287.12	435.01	2720.00
yr_built	1971.12	29.48	115.00
yr_renovated	81.23	394.38	2015.00
zipcode	98078.29	53.52	198.00
lat	47.56	0.14	0.62
long	-122.21	0.14	1.19
sqft_living15	1994.33	691.90	5650.00
sqft_lot15	12746.32	28241.24	870540.00
price	5.39	3.57	68.08
year	2014.32	0.47	1.00
month	6.59	3.11	11.00
day	15.80	8.62	30.00

Table 1: Stat summary for part 0c.

(d) Most of the features could theoretically be useful based on our limited knowledge of selling houses, so we list below the features we believe will not be useful or will be questionably useful:

- month, day, longitude: These features are cyclic, rather than linear.
- year: As noted, year could theoretically be a proxy for inflation, but the data only has information on houses from 2 years, which is too short for any real inflationary effect to have occurred.
- latitude: If climate affects housing prices we note that this would not be linear with respect to latitude, because the endpoints are coldest, while the midpoint (0) is hottest. Additionally, the range of latitudes is only 0.62, so there is little difference between the latitude of the different houses in the data set.
- yr_renovated: This feature could be useful, but the fact that houses that have never been renovated receive a score of 0 makes this sort of a binary indicator of "has the house been renovated at all".
- zipcode: The numeric value of a zip code is difficult to interpret, though increasing zip codes do tend to correlate with moving west in the US. That being said we expect this trend to be fairly noisy.

Part 1: Learning Rate

In this section, we train our linear regression model with different learning rates and no regularization. We stop training when the gradient is less than 0.5.

(a) Our experiments found that best learning rates for this dataset are 0.1, 0.01, and 0.001 which converged in 35, 336, and 1505 iterations, respectively. As can be seen in the first graph of Figure 2, a higher learning rate resulted in non-convergence behavior. Lower learning rates substantially increased the number of iterations until convergence. We expected to see this behavior. High learning rates demonstrate non-convergence behavior, because they overshoot in their corrections, causing the MSE to increase with more iterations. Lower learning rates take longer to converge, because they make only small corrections to the weights in each iteration.

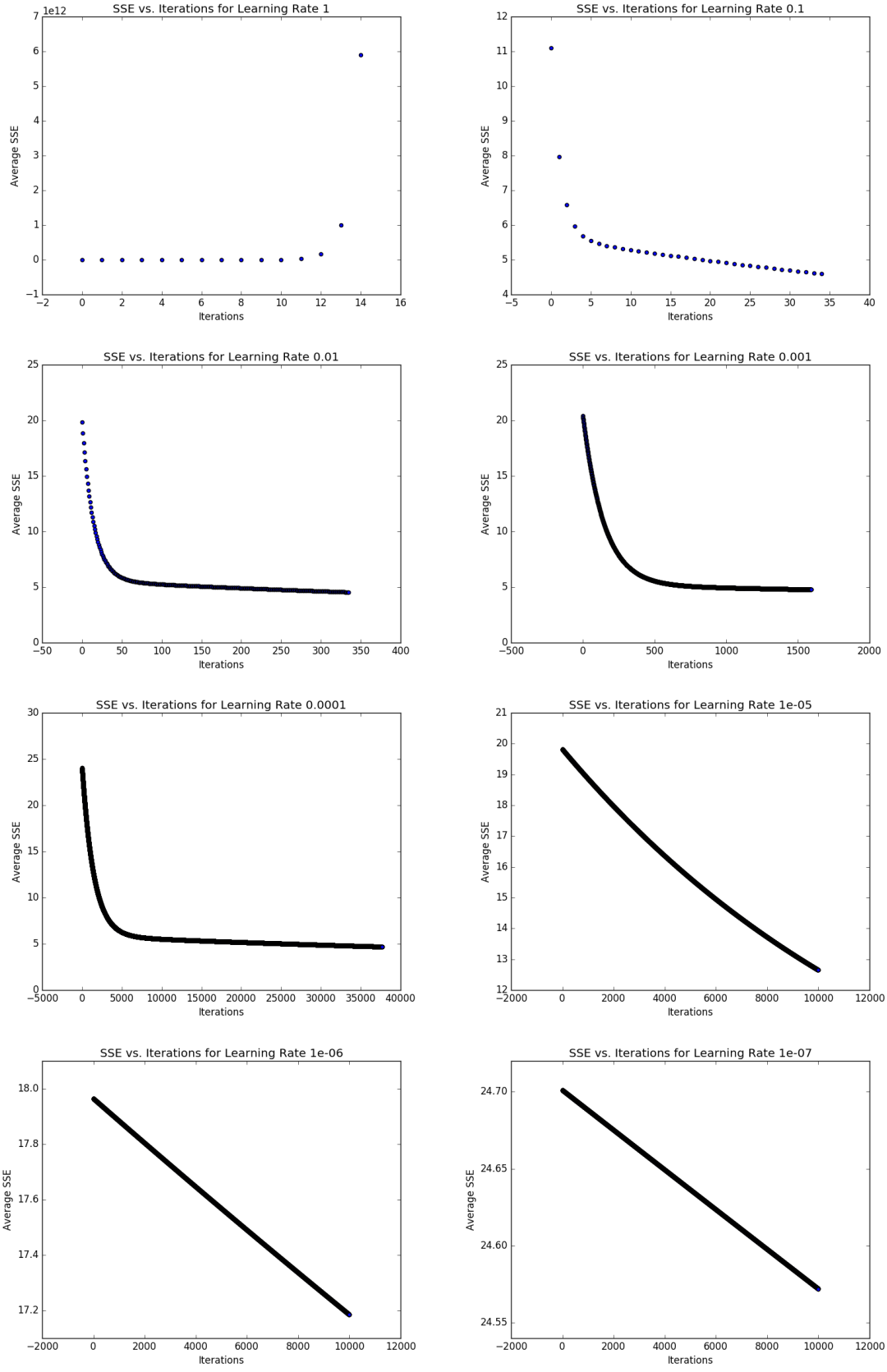

Figure 1: Problem 1a. MSE (multiplied by 1/2) per Iteration by Learning Rate

(b) Table 2 shows the number of iterations until convergence, the MSE on the training data, and the MSE on the validation data for all of the models that we trained until convergence. We do not present learning rates lower than 0.0001, because we set our maximum number of iterations as 40000, and they did not converge within this time. The MSE is similar for the training and validation data.

(c) The lowest MSE for the validation data is 4.53, with a learning rate of 0.01. Training with this learning rate produced the following weights.

- dummy: 1.553392
- bedrooms: -0.359355
- bathrooms: 0.165510
- sqft_living: 1.570926
- sqft_lot: -0.867374
- floors: 1.339114
- waterfront: -0.351074
- view: 0.714037
- condition: 0.914791
- grade: 1.074820
- sqft_above: 0.369707
- sqft_basement: 0.094059
- yr_built: -0.047752
- yr_renovated: 0.408161
- zipcode: 0.087186
- lat: 2.063663
- long: -0.542815
- sqft_living15: 1.759743
- sqft_lot15: 0.165986
- year: 0.419806
- month: 0.157791
- day: 0.723500

The most significant features (which we are defining as ones with a magnitude ≥ 0.5), in descending order, are latitude, the living room area in 2015, the square footage of the home, the dummy variable, the grade, the condition of the house, the square footage of the lot, the day the house was sold, the whether the house has been viewed, and the longitude.

We predicted that the month, day, latitude, longitude, year, year renovated, and zip code would not be useful features. We were correct about the zip code and fairly correct about the month, year, year renovated. The latitude ended up being the most useful feature. The latitude is over a really small region (it looks like somewhere near Seattle), so this feature is likely a proxy for some other geographic feature, such as distance to a major city. The same reasoning applies to the longitude, to a lesser extent. The other feature that we were wrong about was the day of the sale. We expected that this feature would not be useful because it is cyclic rather than linear. However, it may not be treated cyclically because of factors like when people are generally payed.

Some of the features that we thought might be useful that were actually less impactful are the number of bedrooms, the number of bathrooms, the waterfront view, the square footage of the basement, and the square footage of the lot in 2015. There are various reasons that these features may have had lower weights, including that they were simply less important than other features. For example, it makes sense that the square footage of the basement is not a key feature that people look at when pricing homes.

Learning Rate	Iterations Until Convergence	MSE on Training Data	MSE on Validation Data
0.1	35	9.20	9.20
0.01	336	9.08	9.06
0.001	1505	9.56	9.48
0.0001	37706	9.38	9.34

Table 2: Part 1b. Iterations until convergence and MSE by learning rate, no normalization.

λ	0	0.001	0.01	0.1	1	10	100
dummy	0.570886	2.044755	1.627204	2.100648	3.122430	4.709268	4.875048
bedrooms	1.133336	-0.022040	0.399920	-0.269917	0.085325	0.007134	0.000690
bathrooms	1.616822	0.552888	0.620882	0.157462	0.353178	0.031425	0.003086
sqft_living	1.075192	0.513364	1.364489	1.277492	0.255878	0.033043	0.003262
sqft_lot	-0.853811	0.958107	-0.007856	-0.431240	0.049058	0.001268	0.000124
floors	1.358422	1.013115	0.373783	-0.047529	0.304689	0.030940	0.003060
waterfront	1.027406	1.050121	-0.372536	-0.307824	0.088123	0.006972	0.000698
view	0.275423	0.126913	-0.370704	0.630479	0.279971	0.029364	0.002937
condition	1.182407	0.317318	1.823075	0.888167	0.368449	0.035156	0.003296
grade	1.763490	1.383385	1.735584	1.165039	0.514433	0.052898	0.005175
sqft_above	0.430732	1.843469	1.237359	0.968832	0.315087	0.029889	0.002950
sqft_basement	0.377541	1.843469	-0.340236	-0.230598	0.202335	0.022360	0.002210
yr_built	0.502447	0.061451	0.274114	0.465143	0.364840	0.037848	0.003599
yr_renovated	0.094063	-0.153714	0.965420	0.004797	0.114890	0.008794	0.000865
zipcode	0.317425	0.291826	0.224382	0.815824	0.163104	0.016139	0.001471
lat	0.892746	2.155291	1.119058	1.114986	0.547708	0.058823	0.005677
long	0.763955	-0.208942	-0.033197	0.263182	0.189062	0.014429	0.001359
sqft_living15	0.393205	0.436621	1.486939	0.206000	0.339284	0.039982	0.003926
sqft_lot15	-0.254833	-0.884508	-0.356518	0.405398	-0.021682	0.001721	0.000168
year	0.639063	0.568379	-0.030436	0.354261	0.160379	0.017230	0.001612
month	1.321211	0.119727	-0.167463	0.312899	0.311604	0.026237	0.002443
day	0.505472	-0.109593	0.134351	0.336284	0.240713	0.023593	0.002198

Table 3: Part 2 Weights

Part 2: Regularization

In this section, we trained our model using various λ values for regularization. The bias term was excluded from regularization. The following experiments were run using a learning rate of 0.01. The resulting feature weights are shown in Table 3, while MSEs computed on the training and validation models are shown in Table 4.

(a) As we increase λ, we see that the training MSE increases as a result of being discouraged from overfitting to the data. The resulting models are worse at predicting housing values on this data set, but will theoretically be able to better predict prices for new samples. At very high values of λ, the model learns nothing, producing high MSE values.

(b) Unexpectedly, in our experiments, the validation data set also exhibits the same trend of increasing MSE values as the training data when we increase the learning rate. Usually, the validation set MSE will decrease for a while as we increase λ, because overfitting is decreasing, and then increase again when λ gets too high. One possible explanation is that the distribution of the training and validation data sets is extremely similar, since both are drawn from house sales in a relatively small area over the same time period.

(c) The fact that training MSE goes up as we increase λ isn't surprising, as the learning rate is intended to prevent the model from overfitting perfectly to the data. In fact, mathematically, a higher λ must

λ	Training	Validation
0.0	9.16	9.06
0.001	9.23	9.19
0.01	9.50	9.45
0.1	10.28	10.18
1.0	11.52	11.46
10.0	12.85	12.75
100.0	13.00	12.90

Table 4: Part 2 MSEs. Note that these values are not divided by 2, unlike in part 1.

result in a higher MSE because it's equivalent to a constraint on the solution space which tightens as λ increases.

Regarding the validation set, based off the lectures in class, we were expecting to see a local minimum somewhere indicating that the training overfitting was being mitigated, but this wasn't the case, indicating that either overfitting isn't a significant issue for the model or that the validation dataset has very similar characteristics to the training dataset. The fact that we have 10000 data points and "only" 23 regressors probably prevents the training data set from overfitting too much.

(d) We consider any weight with magnitude lower than 0.1 one that has been "turned off".

- At $\lambda = 0$, most coefficients are non-zero, though yr_renovated has a fairly low impact, which we mentioned as one of the variables likely to be of questionable use.
- At $\lambda = 10^{-2}$, most of the weight magnitudes are lower than at $\lambda = 0$. The inactive coefficients are: sqft_lot, long, and year.
- At $\lambda = 10$, all of the non-intercept coefficients have been turned off. Essentially, the constraint is so strong here that the optimizer is only really able to optimize the intercept term, which represents a weighted average of all the prices regardless of the features.

Part 3: Non-normalized Data

In this section, we conducted regression using non-normalized data. For each value, we ran up to 10,000 iterations.

Getting the non-normalized regression to work was difficult, owing to the highly unstable gradient as a result of the large-magnitude values. To make this section interesting (i.e. to not have the gradient instantly blow up for all learning values), we initialize all weights to be between $\pm 10^{-5}$ instead of ± 1. Even with this, all learning rates except 10^{-15} and 0 blew up nearly instantly; learning rates of 1, 10^{-3}, 10^{-6}, and 10^{-9} blew up (i.e. reached a gradient magnitude of over 10^{80}) after 9, 13, 21, and 82 iterations respectively. For the two learning rates that worked, the results showing the MSE are shown in Figure 2. Obviously, a learning rate of 0 means that the model never changes from its initial randomly-seeded weights. The only rate that worked was 10^{-15}, which converges very slowly given just how small it is.

As such, we see that it is much easier to train the normalized version of the weights. The primary reason for this seems to be that having weights of different magnitudes makes the gradient descent unstable; for example, if we have a regressor with a prior weight of 1 and a magnitude of 10^6 (e.g. the zipcode regressor), since the value of the price is typically on the order of 10^1 or 10^2, this causes the resulting error to also be on the order of 10^6, which will cause a huge subsequent weight update, particularly for other regressors which do not share the same scale.

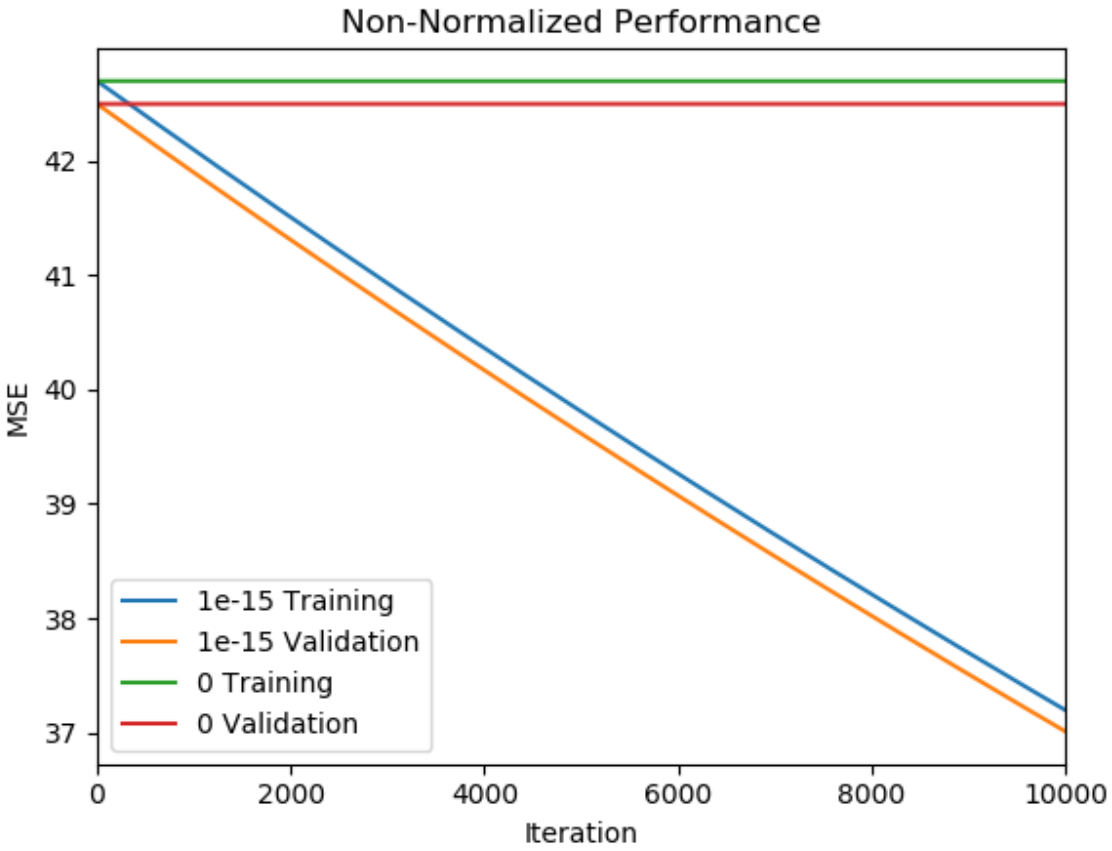

Figure 2: SSEs per iteration when training with non-normalized data. The learning rates which failed (1, 10^{-3}, 10^{-6}, and 10^{-9}) are not plotted.