Group Members
Derek Jackson
Meredith Leung
Dylan Sanderson

Implementation 1

Introduction

This report describes the findings of experimentation with learning rates and regularization parameters in developing an efficient linear machine learning algorithm to predict housing prices based on the houses' features. The algorithm was trained on the data from 10,000 house sales, validated on 5,597, and tested on 6,000. Our results suggest that for this data set, a learning rate of 10e-6 and a regularization parameter of $\lambda = 0$ optimize our price predictions.

Part 0: Pre- Processing

Pre-processing our data ensures that the weights in our machine learning algorithm are not skewed or weighted more heavily for some features than others. In this stage, it's also important to remove data that is not relevant for learning because this could result in an algorithm that is trained on information that doesn't actually help predict realistic outputs. For example, in this implementation, we removed the ID feature. The majority of house IDs are missing, and furthermore may be arbitrary or labeled based on factors irrelevant to housing prices. Training on data that is incomplete and potentially has no bearing on our predictand, would undermine the learning done based on other features.

During pre-processing we also separated the date string of sell date into month, day, and year arrays. This allows our ML algorithm to make predictions on housing price swings based on yearly, monthly, and daily cycles. Another potential way to have handled the date data would have been to create a datetime array that contains the year, month, and day information all in one place. This may have kept information from getting separated, but our algorithm may have struggled when trying to make predictions for combinations of dates it hadn't seen before, as it would not have been able to 'remix' what it had learned from the day, month, year cycles separately.

We input 20 distinct features into our ML implementation that describe different aspects of the homes which supposedly will be helpful in predicting housing prices. A table of these features and their means, standard deviations, and ranges can be seen below. When features were categorical or numeric they were reported in percentages of occurrence.

Feature	Mean	Standard Deviation	Range
Bedrooms	3.4	0.943	[1, 33]
Bathrooms	2.1	0.765	[0.5, 7.75]
Sqft_living	2080.2	911.3	[370, 9890]
Sqft_lot	15089.2	41203.9	[572, 1651359]
Floors	1.5	0.543	[1.0, 3.5]
Sqft_above	1793.1	830.9	[370, 8860]
Sqft_basement	287.1	435.0	[0, 2720]

Feature	Mean	Standard Deviation	Range
Yr_renovated	81.2	394.4	[0, 2015]
Zip code	98078.3	53.5	[98001, 98199]
Lat	47.6	0.139	[47.16, 47.78]
Long	-122.2	0.141	[-122.514, -121.319]
Sqft_living15	1994.3	691.9	[460, 6110]
Sqft_lot15	12746.3	28241.2	[660, 871200]
Price	5.39	3.57	[0.82, 68.9]

Yr_built	1971.1	29.5	[1900, 2015]

Table 1. List of numerical features and their means, standard deviations, and ranges.

Feature	#	Pct of homes with this number
Waterfront	0	99.3
	1	0.70
Condition	1	0.13
	2	0.76
	3	65.3
	4	25.7
	5	8.12
View	0	90.3
	1	1.62
	2	4.25
	3	2.50
	4	1.33
Grade	4	0.110
	5	1.05
	6	9.33
	7	41.3
	8	28.3
	9	11.8
	10	5.47
	11	2.10
	12	0.39
	13	0.05
Year	2014	68.15
	2015	31.85
Month	1	4.54
	2	5.75
	3	8.38
	4	10.08
	5	11.23
	6	10.30
	7	10.37
	8	8.74
	9	8.38
	10	8.87
	11	6.44
	12	6.84
Day	1	2.5
	2	3.5
	3	3.03
	4	3.2
	5	3.5
	6	3.1
	7	3.2
	8	3.4
	9	3.7
Day	10	3.3
	11	2.95
	12	3.42
	13	3.4
	14	3.1
	15	2.88
	16	3.5
	17	3.5
	18	3.4
	19	3.04
	20	3.8
	21	3.2
	22	3.27
	23	4.4
	24	3.9
	25	3.3
	26	3.2
	27	3.3
	28	2.87
	29	2.98
	30	3.03
	31	1.19

Table 2. Percentages of categorical and ordinal features

Based on our intuition of the features we expect to have the biggest impact on predicting price are the number of bedrooms, number of bathrooms, zip code, and square footage of the home (both living and lot-- particularly the 2015 values more so than at the time of being built). Categorical and numerical features like waterfront, view, and year of sale may also have a large impact. Based on the statistics presented above, the number of bedrooms, number of bathrooms, and square footage will still have large impacts because their means are small compared to the max range of the feature. So houses that deviate far from the mean will be effected more by the weight. Similarly, a smaller standard deviation (relative to the features range) will cause the feature to have a larger impact.

Part 1: Learning Rates for Batch Gradient Descent

The learning rates that worked for this dataset with a λ of 0 were 10e-5, 10e-6, and 10e-7. Larger learning rates, such as 10e0, 10e-1, 10e-2, 10e-3, and 10e-4, cause the gradient descent to explode. Example curves of SSE as a function of training iterations and convergence behavior can be found in figure 1. The successful learning rates and their SSEs are reported in Table 3.

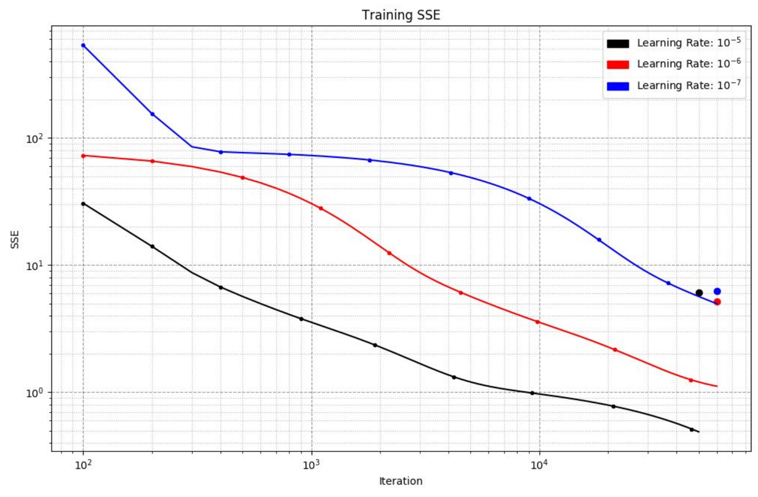

Figure 1(above) shows SSEs for the three different learning rates that converged. The points represent the SSEs of the validation data sets with colors corresponding to learning rate.

Learning Rate	Num. Iterations for Convergence in Training	SSE Training	SSE Validation
10e -5	50001	0.48922304124286	6.05970749112
10e -6	60001	1.11652048732040	5.17038427609
10e -7	60001	4.97753433074505	6.26343259188

Table 3. Learning rates that converged, the number of iterations needed to reach convergence, and their respective SSEs for training and validation.

From the table above, it becomes clear that the smaller the learning rate, the longer it took to converge. While it takes longer, the smaller learning rates were also able to limit overstepping of the minimum, and so it would converge to a lower SSE. However, we limited the number of iterations for training, stopping before reaching the convergence criteria of 0.5 to limit the amount of time needed to run.

Based on the SSE of the validation data, the best converged solution occurred with a learning rate of 10e-6. Interestingly, the best converged solution for the validation data is not the same as the best solution for the training data.

Feature	Weight
Bedrooms	-0.05867546
Bathrooms	-0.00687314
Sqft_living	0.10129459
Sqft_lot	-0.0172629
Floors	0.00374841
Sqft_above	0.06849054
Sqft_basement	0.03598982
Yr_built	0.04240111
Yr_renovated	0.20258751
Zip code	0.10129675
Lat	0.03835085
Long	-0.13144094
Sqft_living15	0.01651717
Sqft_lot15	-0.00814925
Waterfront	0.03509558
Condition	-0.00276398
View	0.01947609
Grade	-0.04373772
Year	-0.01020217
Month	-0.00359234
Day	-0.00382314

Table 4. Weights of the features produced with a learning rate of 10^{-6}, which generated the best SSE of our validation data. Most influential weights are highlighted.

The weights learned from this iteration can be found in Table 4. Based on these weights the most important features for predicting housing prices are the year renovated, zip code, and longitude. We were surprised to see the number of bedrooms and bathrooms were less significant than we predicted based on the statistics of the features and our intuition, while the zip code proved to be a significant feature. Our incorrect prediction of which features will have a large impact could be due to analyzing the statistics of non-normalized data, or goes to show how the weight of a feature is not heavily dependent on their statistics.

Part 2: Experiment with different λ values

From part (1), 10^{-5}, 10^{-6}, and 10^{-7} were the only learning rates to converge. Although the learning rate of 10^{-6} resulted in the best validation SSE, we chose to vary the regularization parameters (λ) with a learning rate of 10^{-5} in order to optimize run times. Here, we experimented with regularization parameters of 0, 10^{-3}, 10^{-2}., 10^{-1}, 1, 10, and 100. Small values of the regularization parameter penalize complex models by encouraging small weight values and creating a better fit to the data (decrease SSE), while larger lambda values encourage a simpler model by driving more elements of the weights to zero. We are using an L-2 regularization, which curbs overfitting but does not always produce sparse solutions.

Results shown in Figure 2, indicate that as the regularization parameter, λ, decreases from 100 to 0, the SSE decreases (**2a**). The SSE is defined as:

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

The SSE equation shows that a large regularization parameter will result in larger SSE values.

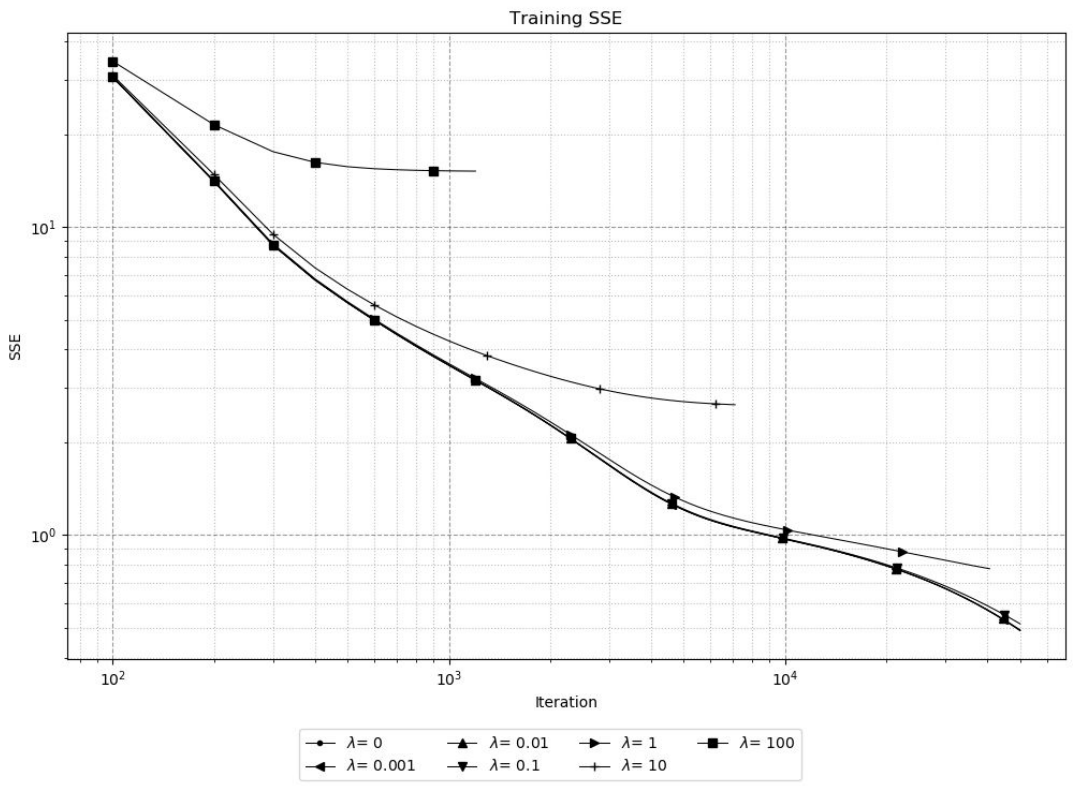

Figure 2: Training SSE convergence for a learning rates of 10^{-5}.The different marker styles correspond to regularization parameters ranging between 0 and 100.

Table 5 summarizes both the training and validation SSE values. Whereas the training SSE decreases with a decreasing regularization parameter, the validation SSE increases (**2b**). The regularization parameter of 100 results in an outlier for the validation SSE (10.394). Small regularization parameters result in overfitting to the training data (resulting in small errors). The overfit models do not perform well on validation data, ultimately resulting in larger errors (**2c**).

Learning rate	λ	Training SSE	Validation SSE
10^{-5}	0	0.489	6.606
	10^{-3}	0.489	6.059
	10^{-2}	0.492	6.055
	10^{-1}	0.515	6.010
	1	0.777	5.628
	10	2.648	5.540
	100	15.182	10.394

Table 5: Training and validation SSE values

Figure 3 shows the resulting weight values for each variable and regularization parameter (**2d**). While the weights do not vary significantly for many features, it can be seen that there is strong variation for the year built. The smallest weights occur for a regularization parameter of 0, and increase as the regularization parameter increases.

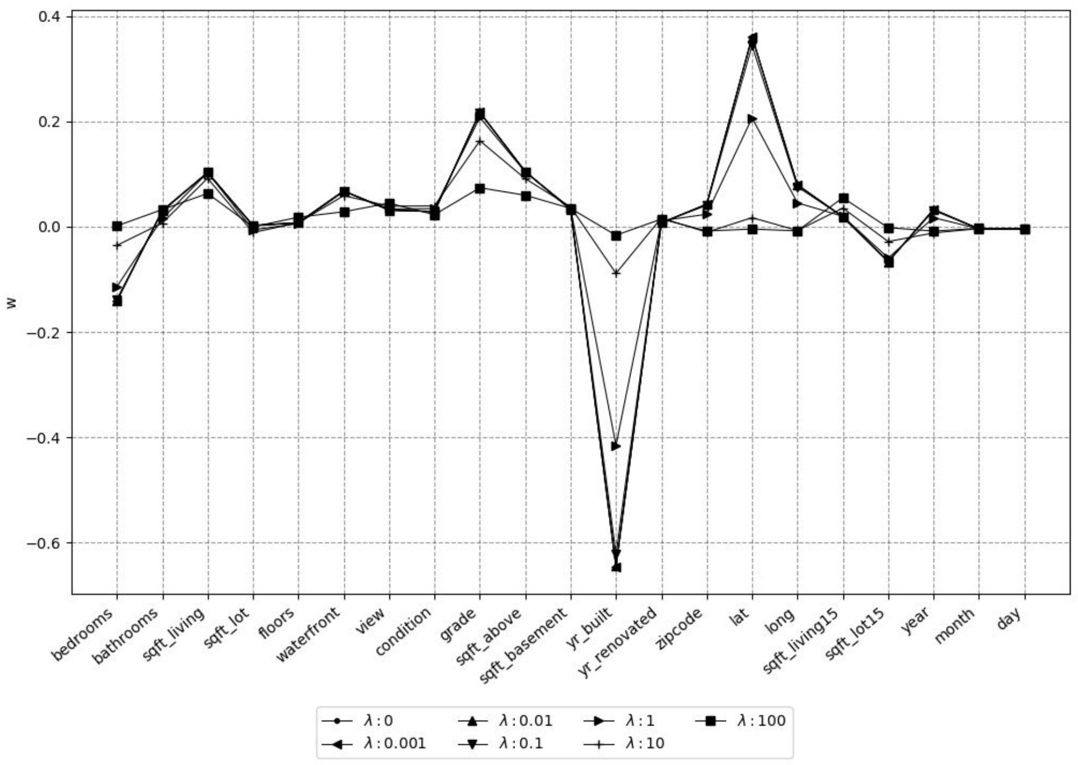

Figure 3: Variations in weight values by regularization parameter.

Part 3: Training with Non-Normalized Data

To compare how well our machine learning algorithm works with normalized vs. non normalized data, we ran our non-normalized training and validation data with different learning rates. Apart from when the learning rate was set to 10e-15 and zero, all other learning rates we terminated early as their loss gradient started to exceed 10^{30} (SSE diverged). When the learning rate was zero, it did not exceed our 10^{30} SSE threshold, but it did not ever converge. SSE remained at 31665228305.71812 until it was terminated for exceeding 10,000 iterations. With a learning rate of 10e-15, the run was also terminated after the 10,000 iterations, however it was showing signs of converging. After the first 1000 iterations it had an SSE of 61418224. After the 10,000 iterations the SSE had decreased to 22721026. It likely would have continued to converge if not terminated early.

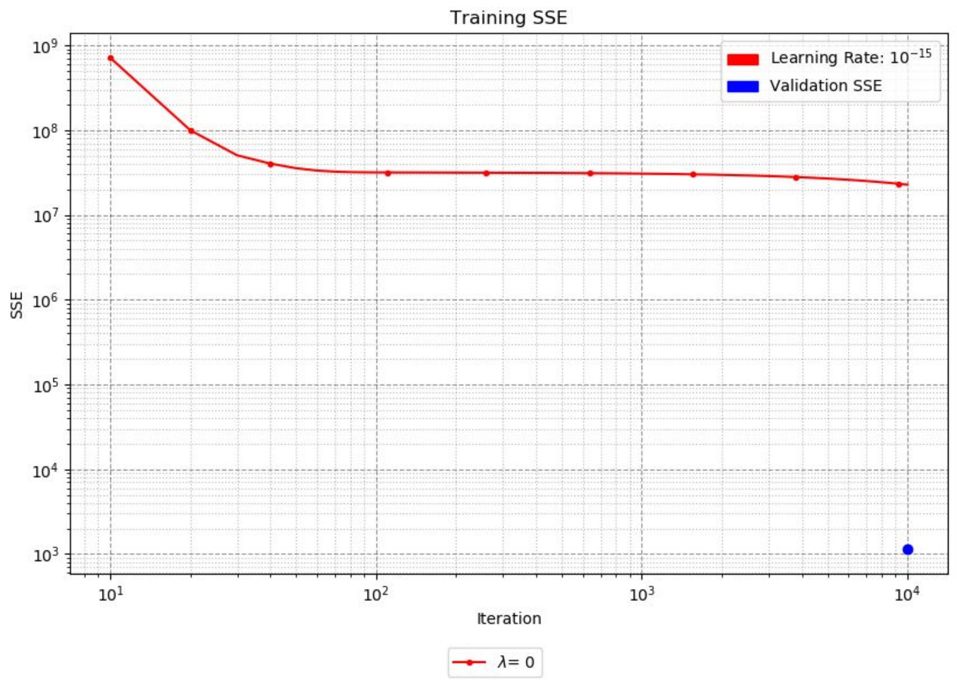

Figure 4: Training SSE convergence. Learning rate (red) 10^{-15}. Validation data SSE (blue).

Figure 4 shows the Training SSE convergence along with the SSE of the validation data. As can be seen from the figure, the validation SSE (1148.37) is much lower than the final training SSE (22721026).

Normalized versions of the data were much easier to train with than the non-normalized versions as evidenced by the speed and overall ability to converge with different learning rates. This is because our non-normalized features are all recorded in different units / have vastly different magnitudes. This means that each of our features have different learning rates that are optimal for them. With different learning rates, it is much easier to overstep our minimum during gradient descent and cause a divergence, therefore a much smaller step size is needed.

Summary / Discussion of Results

After experimenting with learning rate and regularization parameters we determined that a learning rate of 10^{-5} and $\lambda = 0$ were the most efficient options to predict prices based on housing features. While an SSE of 10^{-6} and 10^{-7} may have eventually converged to smaller SSEs, they were computationally expensive and were stopped early.

In completing this Implementation, we were surprised by a few results.

- Part 0: The features we originally expected to drive housing price predictions were not as heavily weighted as expected
  - One possible reason the features that we originally expected to have large weights were not heavily weighted could be that mean, standard deviation, and range don't fully capture the distributions of data.
- Part 1: For learning rate 10^{-6}, the SSE of the validation data was lower than the SSE of the training data, despite not reaching the full convergence criteria of (loss gradient < 0.5). Despite 10^{-7} ending with a convergence criteria of ~10, the validation SSE was almost as good as those for 10^{-6} and 10^{-5}, which ended with convergence criteria of ~2 and 0.9 respectively.
  - The fact that a learning rate of 10^{-6} had a lower SSE for validation without reaching as low a convergence criteria as the learning rate of 10^{-5} could show that learning rate 10^{-5} had begun to overfit the training data. Thus proving to be less accurate for the validation data.
- Part 2: The validation SSE associated with a regularization parameter of 100 resulted in an outlier.
  - This indicates that the model was poorly fit.
- Part 3: For a learning rate of 10^{-15} the SSE of the training data (22721026) was much higher than the SSE of the validation data (1148.37). While still poor, the difference in the magnitudes of the SSE is large.
  - We aren't quite sure why this happened. Perhaps a similar phenomenon as what surprised us in Part 1?