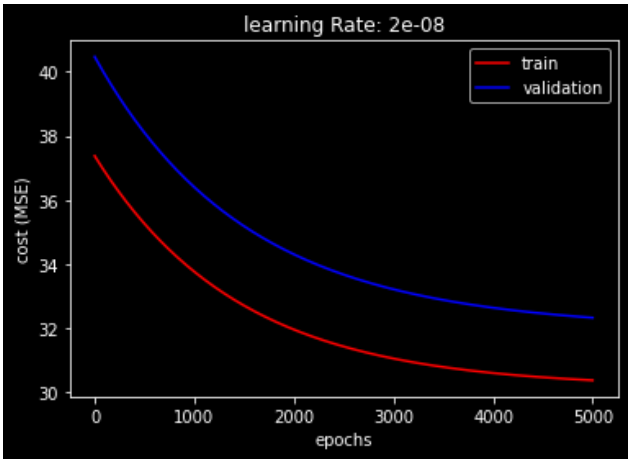
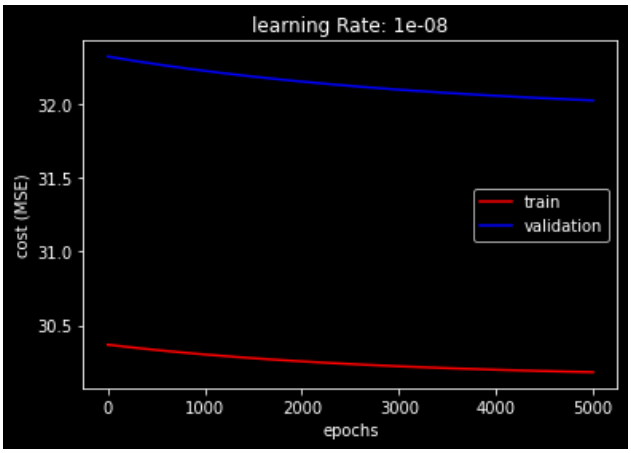
Submitted by: Oluwasegun Somefun, somefuno@oregonstate.edu

AI534 — IA1 Homework Report Due Oct 15st 11:59pm, 2021

1 Part 1: Implement batch gradient descent and explore different learning rates.





a.1) Which learning rate or learning rates did you observe to be good for this particular dataset?

Typically, learning rates lesser than or equal to 10^{-6} converged on this particular dataset.

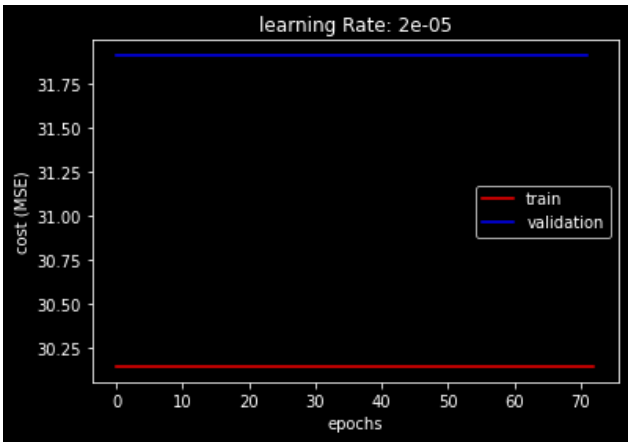
a.2) What learning rates (if any) make gradient descent diverge?

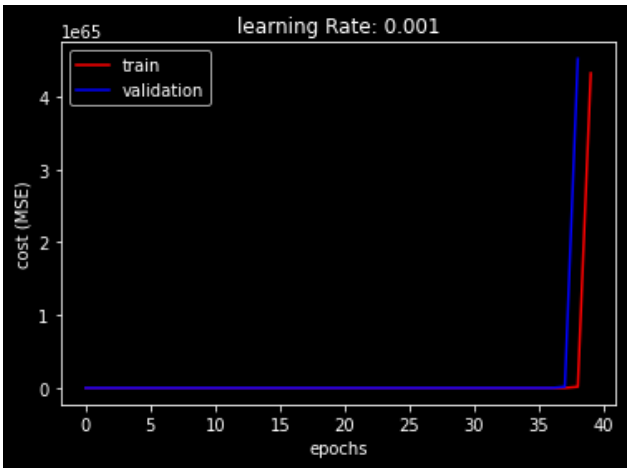
Typically, learning rates greater than 10^{-3} made the gradient descent to diverge greatly.

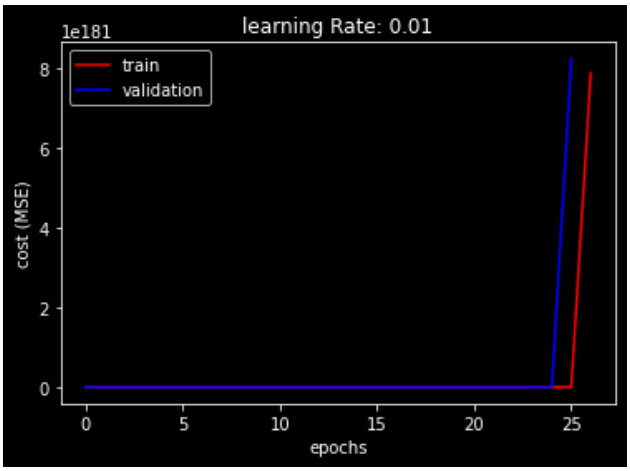
b.1) Which learning rate leads to the best validation MSE?

Among converged learning rates, 10^{-6} gave a Train MSE: 30.142 and the best Validation MSE: 31.91

b.2) Between different convergent learning rates, how should we choose one if the validation MSE is nearly identical?







We could choose the one with the smallest training MSE of 'convergence' to give the best of both approximation and estimaton error.

c.1) What features are the most important in deciding the house prices according to the learned weights?

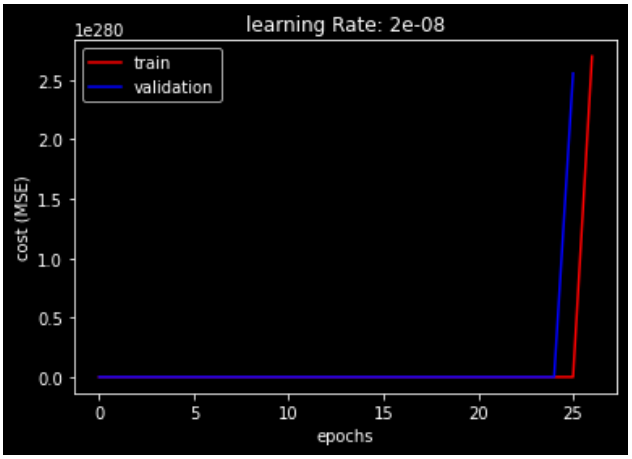
The learned weights in the features according to how they are arranged in the dataset are:

$$[[0.4852, 0.45, 0.3998, 0.4375, 0.4045, 0.4523, 0.4991, 0.5222, 0.545, 10.4624,$$

$0.4178, 0.4397, 0.4101, 0.4189, 0.3883, 0.4745, 0.4615, 0.552, 0.4653, 0.47560.4165]]$

Based on the learned weights, the following features according to the learned weights were most important in deciding or estimating the house prices.

2 Part 2. Training with non-normalized data



a.1) What learning rates work for the un-normalized data?

None of the learning rates worked for the un-normalized data

a.2) Compare between using the normalized and the non-normalized versions of the data. Which one is easier to train and why?

Comparing both versions of the trained data, the normalized version was easier to train. One reason, may be because after normalization, the data now resembles a gaussian distribution compared to the un-normalized data, with a lot of skewed features.

b.1)

Compare the weights to those of 1(c), what difference do you observe? What is the explanation for such difference? How does this impact the interpretation of the weights as the measure of feature importance.

Compared to the weights of 1.c, the weights of the un-normalized training data, were very large, approaching infinity in some cases, so led to a very poor model, which underfitted the data. This is because linear regression expects a dataset which with an underlying guassian distribution, which is lacking in the un-normalized dataset.

This means to prevent underfitting and overfitting, learned weights alone cannot be used to indicate the importance of a feature.

3 Part 3. Redundancy in features.

Questions

a.1) How does this new model compare to the one in part 1 (c)? What do you observe when comparing the weight for sqft_living in both versions? Consider the situation in general when two features x1 and x2 are redundant, what do you expect to happen to the weights (w1 and w2) when learning with both features, in comparison with w1 which is learned with just x1? Why?

Compared to the 1.c, this new model, for a learning-rate of 10^{-6} was able to converge slightly to a lesser MSE, for both training and validation. Train MSE: 29.52, Validation MSE: 31.49

Learned Weights:

$$[[0.6068, 0.4864, 0.4668, 0.5704, 0.6068, 0.5003, 0.5851, 0.4958, 0.4878, 0.5567,$$

$$0.6024, 0.5813, 0.5613, 0.4684, 0.5227, 0.4685, 0.6333, 0.6309, 0.4522, 0.4956]]$$

It can be observed that the weight of sqft_living slightly increased in this new model, placing more importance on it.

Models based on two highly correlated features will need more iterations and may take more time to train (converge), and would have a hard time, balancing the right weights between the two features. This would not be the case if redundant features are eliminated from the data, allowing the model to place more emphasis on the right weights.

4 Part 4 Explore feature-engineering and Participate in-class Kaggle competition.

Questions

a.1) Come up/experiment with at least three different ways of modifying the feature set (similar to 0(c) or part 3) to improve the performance. Your grade will depend on the sensibility and creativity of your explored modifications.

Some of the ways the feature set was modified in `preprocess.py` for this task are as follows:

1. Making the time feature (month and day) sets cyclical with respect to quarters, to improve its correlation to the price.

2. Normalizing the data and Fixing outliers by replacing with the extrema tail values via the inter-quartile range proximity rule, in-order to make the feature sets more normally distributed.

3. Removing the features that strongly correlated with the price, and also showed high correlation with each other, inorder to reduce multi-colinearity among the features.

4.In general, observing the correlation matrix of the data, we see that the 5 most correlated features with price are: 'sqft_living,sqft_living15,grade,sqft_above,bathrooms'. Interestingly, most of these fetaures are also highly correlated with each other.

5. Also, the grade, condition, view and waterfront feature sets can be renormalized between 0 and 1 and then combined to form one integrated feature, since they all indicate a form of aesthetics.

6. Logarithmic transformations was also applied to select features, to attempt to reduce their skewness and enforce normal distribution.

Further, see the comments in `preprocess.py` in the submitted code for this assignment.

Generally after doing this, both the training and validation MSE reduced to: Train MSE: 22.02, Validation MSE: 22.93

The final feature set used and their correlation with price are shown in Table 1:

The corresponding learned weights in order are:

$$[[0.9023, 0.8808, 0.9255, 0.8225, 0.8696, 0.9206, 0.9334, 0.8622]]$$

It should be noted that since the results and initial weights were uniformly randomised, the weights in the implementation may slightly differ. (A random.seed function was not used to ensure repeatability, since different initial random starts could lead to better results.)

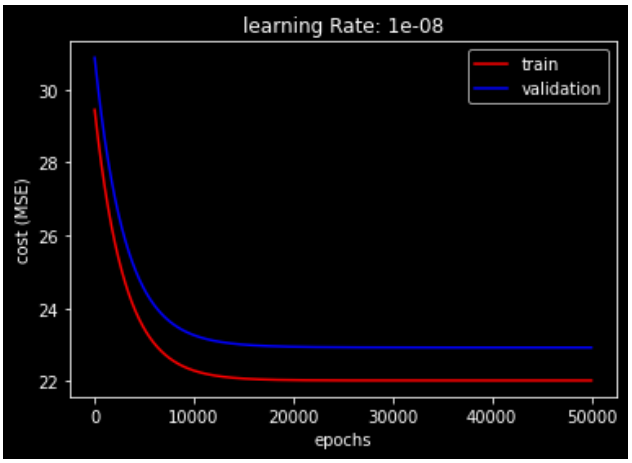


Table 1: Final feature set and correlation with price

Price	Correlation with Price
bedrooms_log	0.350
bathrooms_log	0.497
sqft_living	0.698
floors	0.302
wvgc_n	0.487
age_since_renovated	-0.108
lat_log	0.397