# Implementation Assignment 1

Deqing Qu (qud@oregonstate.edu) (50%)

Kejo Hsieh (hsiehke@oregonstate.edu) (50%)

## Part 0 Preprocessing and simple analysis

**a) Remove the ID feature. Why do you think it is a bad idea to use this feature in learning?**

The id is not a factor that affects the price of a real estate, hence it does not include useful information towards generating a prediction. Using this set of data as a feature would cause adverse effects to our model, eventually generate bad predictions which increases the loss.

**b) Split the date feature into three separate numerical features: month, day, and year. Can you think of better ways of using this date feature?**

Splitting the date feature into 3 parts would let us make predictions based on the time cycle within a year or a month. It might include certain trends if there are certain periods in a year or month where you can sell your house at a higher or lower price. If we only use the date feature as a whole, it only provides information based on a one-dimensional timeline, which would hardly be useful since time could not be re-winded.

**c) Build a table that reports the statistics for each feature.**

Training Data:

(Please see next page)

| Feature | Mean | Standard De | Range | Min | Max |
|---|---|---|---|---|---|
| month | 6.5924 | 3.11127984 | 11 | 1 | 12 |
| day | 15.8021 | 8.62133027 | 30 | 1 | 31 |
| year | 2014.3185 | 0.46589457 | 1 | 2014 | 2015 |
| bedrooms | 3.3752 | 0.94319932 | 32 | 1 | 33 |
| bathrooms | 2.118875 | 0.76508985 | 7.25 | 0.5 | 7.75 |
| sqft_living | 2080.2232 | 911.28879 | 9520 | 370 | 9890 |
| sqft_lot | 15089.2014 | 41201.8347 | 1650787 | 572 | 1651359 |
| floors | 1.5037 | 0.54261986 | 2.5 | 1 | 3.5 |
| waterfront | 0.007 | 0.08337266 | 1 | 0 | 1 |
| view | 0.2294 | 0.75589393 | 4 | 0 | 4 |
| condition | 3.4091 | 0.65355733 | 4 | 1 | 5 |
| grade | 7.6732 | 1.18000075 | 9 | 4 | 13 |
| sqft_above | 1793.0993 | 830.82389 | 8490 | 370 | 8860 |
| sqft_baseme | 287.1239 | 434.983513 | 2720 | 0 | 2720 |
| yr_built | 1971.1249 | 29.4791197 | 115 | 1900 | 2015 |
| yr_renovatec | 1973.4386 | 28.8820261 | 115 | 1900 | 2015 |
| zipcode | 0.4146 | 0.49265286 | 1 | 0 | 1 |
| lat | 47.5598142 | 0.13864366 | 0.6217 | 47.1559 | 47.7776 |
| long | -122.21329 | 0.14139761 | 1.195 | -122.514 | -121.319 |
| sqft_living15 | 1994.3261 | 691.865705 | 5650 | 460 | 6110 |
| sqft_lot15 | 12746.3234 | 28239.8309 | 870540 | 660 | 871200 |

| Feature | Category | % |
|---|---|---|
| Waterfront | 1 | 0.007 |
| | 0 | 0.993 |
| Condition | 1 | 0.001 |
| | 2 | 0.008 |
| | 3 | 0.653 |
| | 4 | 0.257 |
| | 5 | 0.081 |
| Grade | 1 | 0 |
| | 2 | 0 |
| | 3 | 0 |
| | 4 | 0.001 |
| | 5 | 0.011 |
| | 6 | 0.093 |
| | 7 | 0.413 |
| | 8 | 0.284 |
| | 9 | 0.118 |
| | 10 | 0.055 |
| | 11 | 0.021 |
| | 12 | 0.004 |
| | 13 | 0.001 |
| ZIP Code | 980 | 0.585 |
| | 981 | 0.415 |

Dev:

| Feature | Mean | Standard De | Range | Min | Max |
|---|---|---|---|---|---|
| dummy | 1 | 0 | 0 | 1 | 1 |
| month | 6.53082008 | 3.13310043 | 11 | 1 | 12 |
| day | 15.6751831 | 8.64836758 | 30 | 1 | 31 |
| year | 2014.33357 | 0.47148868 | 1 | 2014 | 2015 |
| bedrooms | 3.36591031 | 0.90524998 | 8 | 1 | 9 |
| bathrooms | 2.1113543 | 0.76355723 | 7.25 | 0.75 | 8 |
| sqft_living | 2073.00125 | 906.762023 | 13150 | 390 | 13540 |
| sqft_lot | 14601.3763 | 38419.1154 | 1023459 | 609 | 1024068 |
| floors | 1.48731463 | 0.53684993 | 2.5 | 1 | 3.5 |
| waterfront | 0.00786135 | 0.08831508 | 1 | 0 | 1 |
| view | 0.22994461 | 0.76634839 | 4 | 0 | 4 |
| condition | 3.40414508 | 0.65124828 | 4 | 1 | 5 |
| grade | 7.6482044 | 1.15327984 | 10 | 3 | 13 |
| sqft_above | 1784.97231 | 817.001899 | 9020 | 390 | 9410 |
| sqft_baseme | 288.028944 | 441.919858 | 4820 | 0 | 4820 |
| yr_built | 1971.06754 | 29.1700469 | 115 | 1900 | 2015 |
| yr_renovatec | 1973.52635 | 28.6124579 | 115 | 1900 | 2015 |
| zipcode | 0.41468644 | 0.49266784 | 1 | 0 | 1 |
| lat | 47.5605231 | 0.13902324 | 0.6154 | 47.1622 | 47.7776 |
| long | -122.21362 | 0.1409749 | 1.196 | -122.511 | -121.315 |
| sqft_living15 | 1977.85939 | 669.858563 | 5540 | 670 | 6210 |
| sqft_lot15 | 12812.61 | 27159.8441 | 434077 | 651 | 434728 |

| Feature | Category | % |
|---|---|---|
| Waterfront | 1 | 0.008 |
| | 0 | 0.992 |
| Condition | 1 | 0.001 |
| | 2 | 0.009 |
| | 3 | 0.651 |
| | 4 | 0.26 |
| | 5 | 0.078 |
| Grade | 1 | 0 |
| | 2 | 0 |
| | 3 | 0 |
| | 4 | 0.002 |
| | 5 | 0.013 |
| | 6 | 0.091 |
| | 7 | 0.412 |
| | 8 | 0.288 |
| | 9 | 0.125 |
| | 10 | 0.05 |
| | 11 | 0.014 |
| | 12 | 0.004 |
| | 13 | 0 |
| ZIP Code | 980 | 0.585 |
| | 981 | 0.415 |

Test:

| Feature | Mean | Standard Dev | Range | Min | Max |
|---|---|---|---|---|---|
| month | 6.5835 | 3.10381933 | 11 | 1 | 12 |
| day | 15.5078333 | 8.64102262 | 30 | 1 | 31 |
| year | 2014.3205 | 0.46666878 | 1 | 2014 | 2015 |
| bedrooms | 3.37666667 | 0.91712716 | 9 | 1 | 10 |
| bathrooms | 2.11491667 | 0.78033272 | 7.5 | 0.5 | 8 |
| sqft_living | 2087.31517 | 939.561862 | 11670 | 380 | 12050 |
| sqft_lot | 15581.0023 | 44341.9598 | 1164274 | 520 | 1164794 |
| floors | 1.48441667 | 0.53708984 | 2.5 | 1 | 3.5 |
| waterfront | 0.00816667 | 0.08999985 | 1 | 0 | 1 |
| view | 0.2465 | 0.78341416 | 4 | 0 | 4 |
| condition | 3.41633333 | 0.64472208 | 4 | 1 | 5 |
| grade | 7.6415 | 1.17982107 | 9 | 4 | 13 |
| sqft_above | 1784.47383 | 832.492917 | 8190 | 380 | 8570 |
| sqft_baseme | 302.841333 | 455.640113 | 3500 | 0 | 3500 |
| yr_built | 1970.72767 | 29.3882942 | 115 | 1900 | 2015 |
| yr_renovated | 1973.15417 | 28.8648702 | 115 | 1900 | 2015 |
| zipcode | 0.41766667 | 0.49317464 | 1 | 0 | 1 |
| lat | 47.5601564 | 0.13794403 | 0.6182 | 47.1593 | 47.7775 |
| long | -122.21548 | 0.13933273 | 1.204 | -122.519 | -121.315 |
| sqft_living15 | 1981.94983 | 688.09334 | 5391 | 399 | 5790 |
| sqft_lot15 | 12727.5395 | 25695.6199 | 411212 | 750 | 411962 |

| Feature | Category | % |
|---|---|---|
| Waterfront | 1 | 0.80% |
| | 0 | 99.20% |
| Condition | 1 | 0.10% |
| | 2 | 0.70% |
| | 3 | 64.10% |
| | 4 | 27.50% |
| | 5 | 7.50% |
| Grade | 1 | 0.00% |
| | 2 | 0.00% |
| | 3 | 0.00% |
| | 4 | 0.10% |
| | 5 | 1.10% |
| | 6 | 9.90% |
| | 7 | 42.30% |
| | 8 | 26.90% |
| | 9 | 12.20% |
| | 10 | 5.10% |
| | 11 | 1.80% |
| | 12 | 0.40% |
| | 13 | 0.10% |
| ZIP Code | 980 | 58.20% |
| | 981 | 41.80% |

**d ) Based on the meaning of the features as well as the statistics, which set of features do you expect to be useful for this task? Why?**

All of the features should be useful for the task. Yet we imagine the house area would be most important of them all since how big of a house should be the primary key towards its price. The year built should also be important since a house's price will depreciate with time. Overall, the grade that has received for the house should give us a comprehensive view of the price.

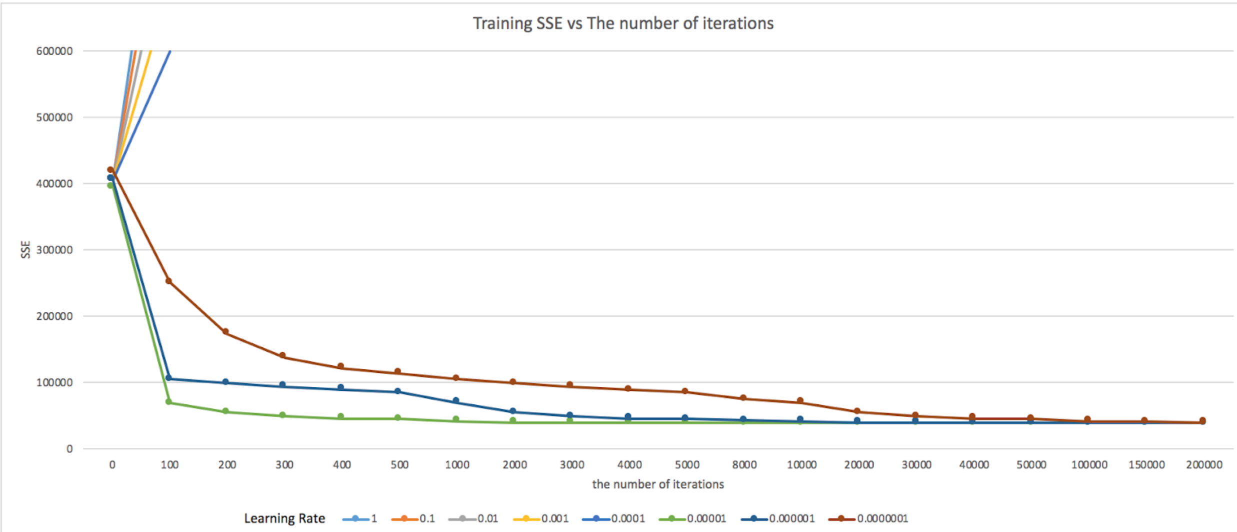
**e ) Normalize all features to the range between 0 and 1 using the training data.**

For each column (feature) in the data, we normalize them by deducting the smaller value in the column, then divided by the range of the column. Code implementation could be found in the normalize_matrix() function from our source code.

$$normalized\ x_i = \frac{x_i - x_{min}}{x_{max} - x_{min}}$$

# Part 1 Explore different learning rate

**a) Which learning rate or learning rates did you observe to be good for this particular dataset? What learning rates make the gradient descent explode?**




According to the result, the gradient descent will only work starting from the learning rate = $10^{-5}$. Although the learning rate has to be small enough to make our gradient descent functional, it does not provide a better result when the learning rate is too small. Once it is smaller than $10^{-5}$, it takes a lot more iterations to get a lower norm of the gradient, hence making it harder to converge. This indicates that we are taking too small of a step towards the ideal loss range, and it would take forever to reach the goal if we don't make bigger steps.

**b) For each learning rate worked for you, Report the SSE on the training data and the validation data respectively and the number of iterations needed to achieve the convergence condition for training. What do you observe?**

| Learning Rate | 0.00001 | 0.000001 | 0.0000001 |
|---|---|---|---|
| SSE on the training set | 39,370.91 | 39,374.84 | 39,491.47 |
| SSE on the validation set | 22,939.45 | 23,109.93 | 23,167.24 |
| The number of iterations to convergence | 148,433 (convergence, norm of gradient = 0.5) | 500,000 (no convergence, norm of gradient = 2.90) | 500,000 (no convergence, norm of gradient = 51.16) |

The validation SSE is nearly half the training SSE, due to the validation dataset itself is half the size of the training dataset. If we divide the SSE for the training set in half, the SSE would be smaller than the validation set. Also, SSE increases in the training set while learning rate decreases due to the reason mentioned in 1a.

**c) Which features are the most important in deciding the house prices according to the learned weights?**

| Feature | Weight | Feature | Weight |
|---|---|---|---|
| Bias | -2.80085068 | Condition | 1.42377189 |
| Month | 0.19954502 | Grade | 8.9102138 |
| Day | -0.16887045 | Sqft_above | 7.91890032 |
| Year | 0.38426361 | Sqft_basement | 1.29948111 |
| Bedrooms | -2.92872561 | Year_built | -3.273957 |
| Bathrooms | 3.52862237 | Year_Renovated | 0.50622065 |
| Sqft_living | 7.20665111 | Zip code | 0.03393697 |
| Sqft_lot | 1.36819341 | Latitude | 3.51936519 |
| Floors | -0.04122866 | Longitude | -1.55944545 |
| Waterfront | 4.72380737 | Sqft_living15 | 1.62366308 |
| View | 2.20282349 | Sqft_lot15 | -2.448195 |

(Top 3 weights are labeled in red)

According to the data, the most important feature among all the the grade that the house received, following by the house area excluding the basement, and the living room's area. These are totally understandable and matches our own assumptions. What turns out to be a surprise is that the year built was not as important as we thought. This could indicate that no matter how old the house is, its value could remain given proper management.

## Part 2 Experiments with different λ values.

| $\lambda$ | SSE on the training set | SSE on the validation set |
|---|---|---|
| $\lambda$ =0 | 39370.26 | 23202.59 |
| $\lambda$ =0.001 | 39370.27 | 23194.98 |
| $\lambda$ =0.01 | 39370.29 | 23177.12 |
| $\lambda$ =0.1 | 39370.58 | 23168.81 |
| $\lambda$ =1 | 39378.37 | 23122.98 |
| $\lambda$ =10 | 39612.20 | 23218.55 |
| $\lambda$ =100 | 44149.37 | 26532.42 |

**Weights ( $\lambda$ =0)**

[-2.80744024 0.20010598 -0.16915518 0.38436488 -2.93083008 3.52879205
7.45364593 2.4598755 -0.04402851 4.72516218 2.19790808 1.42664734
8.90594427 7.69136399 1.22871644 -3.27261545 0.50914146 0.03576403
3.5212088 -1.5537145 1.63921798 -3.23349637]

**Weights ( $\lambda$ =0.001)**

[-2.80744281 0.20010386 -0.16915492 0.38436305 -2.93070049 3.52879608
7.40334111 2.45822904 -0.04398909 4.72509264 2.19793329 1.42663844
8.9059034 7.73594512 1.24304808 -3.27258213 0.50911311 0.03576635
3.52120154 -1.55370628 1.63932176 -3.23212022]

**Weights ( $\lambda$ =0.01)**

[-2.80746651 0.20008478 -0.16915263 0.38434658 -2.92953476 3.52883222
7.2888702 2.44351929 -0.04363462 4.72446705 2.19815974 1.42655866
8.90553509 7.83550184 1.27538371 -3.27228212 0.50885835 0.03578746
3.52113642 -1.55363172 1.64025682 -3.21981389]

**Weights ( $\lambda$ =0.1)**

[-2.80774213 0.19989773 -0.16913186 0.38418244 -2.91792941 3.52917922
7.28849984 2.30446713 -0.04011816 4.71822886 2.20038677 1.42578234
8.90181439 7.81059356 1.2718046 -3.26927543 0.50633684 0.03601053
3.52049944 -1.55284181 1.64967672 -3.10265423]

**Weights ( $\lambda$ =1)**

[-2.81268816 0.198260504 -0.169059267 0.382578128 -2.80644432 3.53121529
7.24737848 1.43170374 -0.0704034182 4.65726577 2.22020923 1.41942656
8.86199888 7.60186343 1.24854715e -3.23890102 0.482960239 0.00389757159
3.51507115 -1.54196192 1.74684342 -2.31540946]

**Weights ( $\lambda$ =10)**

[-2.8447067 0.18453259 -0.17011881 0.36719183 -1.96667913 3.46801241
6.30847794 0.29895814 0.24593572 4.13002635 2.36908871 1.37956592
8.41691333 6.65358987 1.2944505 -2.95797763 0.30663033 0.07125641
3.47837876 -1.38370409 2.50580208 -0.65170268]

**Weights ( $\lambda$ =100)**

[-2.11083342 0.07974424 -0.1772446 0.25931638 0.29782907 2.69297295
4.04056444 0.17046833 1.09241682 2.05389369 2.81345462 1.10854322
5.79047614 4.0138463 1.61390135 -1.67915477 -0.18369711 0.13630947
3.1820594 -0.56663365 3.52228538 0.0984885 ]

**a) What trend do you observe from the training SSE as we change $\lambda$ value?**

When $\lambda$ =0, it gives us the original SSE without the regularization. The SSE is at its lowest point given $\lambda$ =0. Then would gradually increase with the value of $\lambda$ .

**b) What trend do you observe from the validation SSE?**

For the validation set, we can observe that giving $\lambda \neq 0$ does decrease the SSE in a certain range. However, if the $\lambda$ value becomes too large, the SSE will start to increase again. The best $\lambda$ we observed in the test data set is 1.

**c) Provide an explanation for the observed behaviors.**

Since the regularize process is to prevent overfitting, the SSE for the training set will certainly increase once $\lambda \neq 0$, because overfitting is guaranteed to give a lower SSE for the training dataset.

For the validation data, we can see that $\lambda$ has a best value that gives the lowest SSE, any values larger or smaller would increase the SSE. This is because $\lambda$ determines how smooth the function is. Regularization itself is trying to control the gradient direction to be more aligned to the accumulated weight vectors we have generated through huge amounts of iterations, and $\lambda$ determinates the amount of influence of the accumulated w. So if $\lambda$ is too small, it would be easily effected with some rare case values. However, a $\lambda$ that is too large would make the latter part of the data have almost no influence to the gradient descent's direction, hence increases the loss.

**d) What features get turned off for $\lambda$ = 10, 10^(-2) and 0 ?**

$\lambda$ =0 nothing is turned off

$\lambda$ =0.01 floors

$\lambda$ =10 sqft_lot & sqft_lot15

If a feature gets turned off (|w| drops dramatically), it could indicate that these features are not very important on a whole scale. When a weight drops as the $\lambda$ increase, it could indicate that these features were originally biased to seem like it's important due to some rare-case data.

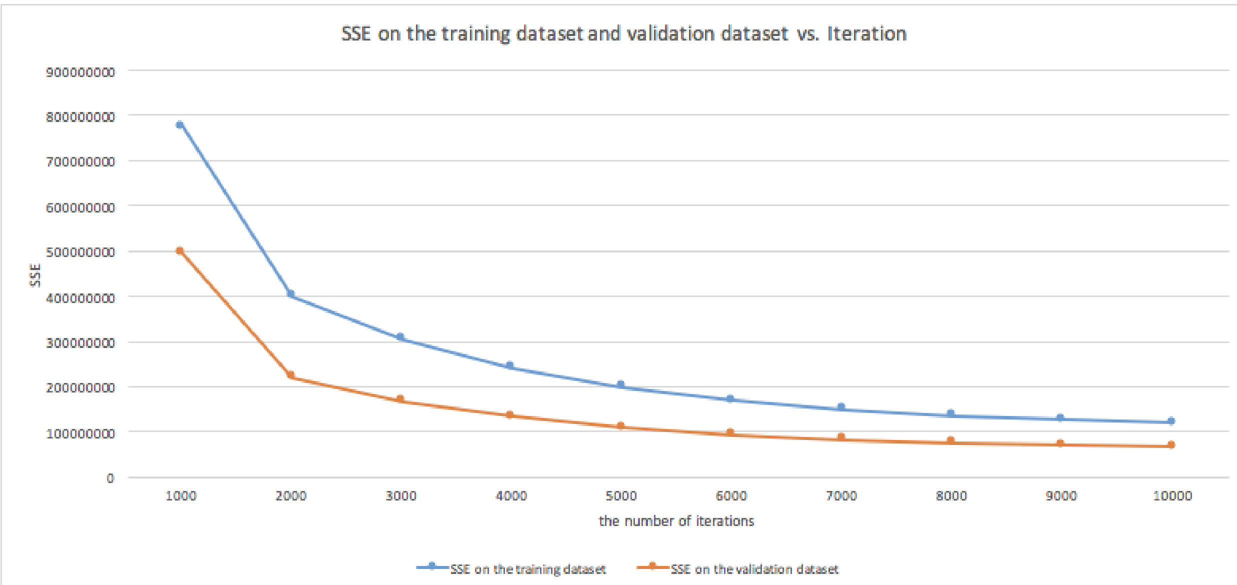
# Part 3

| Learning rate | SSE on the training set | SSE on the validation set |
|---|---|---|
| lr=1 | N/A | N/A |
| lr=0 | N/A | N/A |
| lr=10^-3 | N/A | N/A |
| lr=10^-6 | N/A | N/A |
| lr=10^-9 | N/A | N/A |
| lr=10^-15 | 119580504.11 | 66814097.37 |

The SSE on the training set and validation set for learning rate = 10^-15

| iterations | SSE on the training set | SSE on the validation set |
|---|---|---|
| it=0 | 199375657068.76 | 105549427720.40 |
| it=1000 | 773710507.29 | 494570568.08 |
| it=2000 | 398835389.63 | 220381955.71 |
| it=3000 | 304717979.74 | 167480855.61 |
| it=4000 | 241746811.51 | 133157927.38 |
| it=5000 | 198904631.96 | 109874948.41 |
| it=6000 | 169674711.32 | 93999486.95 |
| it=7000 | 149654589.58 | 83130551.95 |

| | | |
|---|---|---|
| it=8000 | 135866406.14 | 75647889.72 |
| it=9000 | 126295879.97 | 70456007.79 |
| it=10000 | 119580504.11 | 66814097.37 |




**a) What do you observe?**

Non-normalized data explodes very easily, it needs a smaller learning rate than the normalized data to prevent explosion.

**b) Specify the learning rate that prevents the gradient descent from exploding**

The learning rate would have to be lesser than $10^{-15}$ .

**c) Which one is easier to train and why?**

The normalized data is much easier to train. Non-normalized data would easily explode unless the learning rate is small enough, and the weight of each feature would not be very balanced against each other due to the nature that they have very different value ranges. For example, 'year' is normally beyond 2000 but bedrooms will often be a one-digit number. Normalized data do not have this problem and is easier to control.