CS 534 Machine Learning

Implementation #1
Fall 2019

Oregon State of University

Instructor:

Fern, Xiaoli

Team Member:

Zhengqiang Yue (933-284-058) yuez@oregonstate.edu (33%)
Yufei Bai (933-315-539) baiyuf@oregonstate.edu (34%)
YuWen Tseng (933-652-910) tsengyuw@oregonstate.edu (33%)

Part 0

(a)Remove the ID feature. Why do you think it is a bad idea to use this feature in learning?
Because ID is just a number, it has no effect on house prices. Using ID as a feature will have a bad impact on the prediction of house prices.

(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?
We can combine month, day, and year to one feature 'time' so that we have less feature which improves the efficiency of the code. Date feature could also be changed into the day of a week, the day of a month, or seasons. In this way, if forms a cyclical trend rather than a One-dimensional timeline.

(c) Build a table that reports the statistics for each feature. For numerical features, please report the mean, the standard deviation, and the range. Several of the features (waterfront, grade, condition (the latter two are ordinal)) that are marked numeric are in fact categorical. For such features, please report the percentage of examples for each category.

[PA1_train]

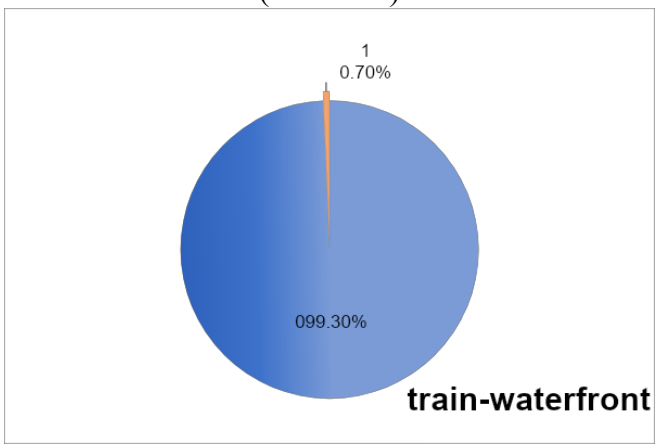
(Range/Mean/Standard Deviation)

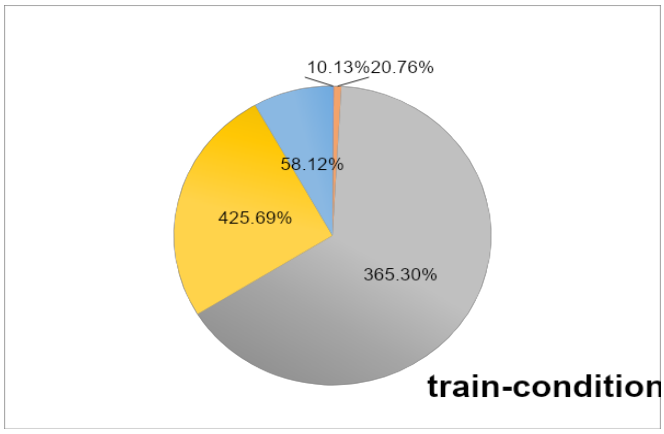
Feature	Max_value	Min_value	Mean_value	Std.dev._value
"dummy"	1	1	1	0
"id"	9900000190	1000102	4586741916	2884509902
'bedrooms"	33	1	3.3752	0.943246485
'bathrooms"	7.75	0.5	2.118875	0.765128111
'sqft_living"	9890	370	2080.2232	911.3343583
'sqft_lot"	1651359	572	15089.2014	41203.89492
'floors"	3.5	1	1.5037	0.542646991
'waterfront"	1	0	0.007	0.083376826
'view"	4	0	0.2294	0.755931732
'condition"	5	1	3.4091	0.653590015
'grade"	13	4	7.6732	1.18005975
'sqft_above"	8860	370	1793.0993	830.8654345
'sqft_basement	2720	0	287.1239	435.0052635
'yr_built"	2015	1900	1971.1249	29.4805938
'yr_renovated"	2015	0	81.2267	394.3798041
'zipcode"	98199	98001	98078.2931	53.51839136
'lat"	47.7776	47.1559	47.5598142	0.13865059
'long"	-121.319	-122.514	-122.213287	0.141404685
'sqft_living15"	6110	460	1994.3261	691.900301
'sqft_lot15"	871200	660	12746.3234	28241.24304
'price"	68.9	0.82	5.38529682	3.57390111

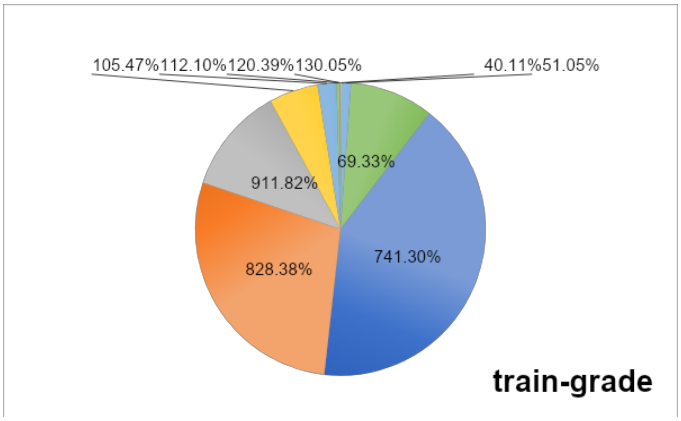
(Percentage for waterfront/condition/grade)

Feature	Category	Percentage
waterfront	0	99.30%
	1	0.70%
condition	1	0.13%
	2	0.76%
	3	65.30%
	4	25.69%
	5	8.12%
grade	1	0.00%
	2	0.00%
	3	0.00%
	4	0.11%
	5	1.05%
	6	9.33%
	7	41.30%
	8	28.38%
	9	11.82%
	10	5.47%
	11	2.10%
	12	0.39%
	13	0.05%

(Pie Chart)







[PA1_test]

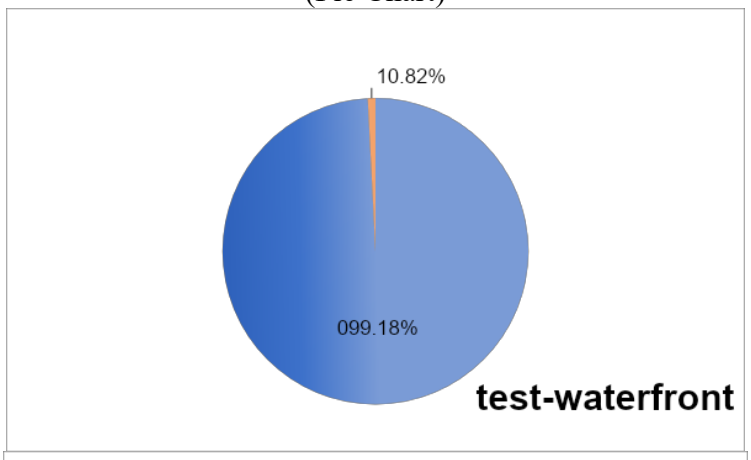
(Range/Mean/Standard Deviation)

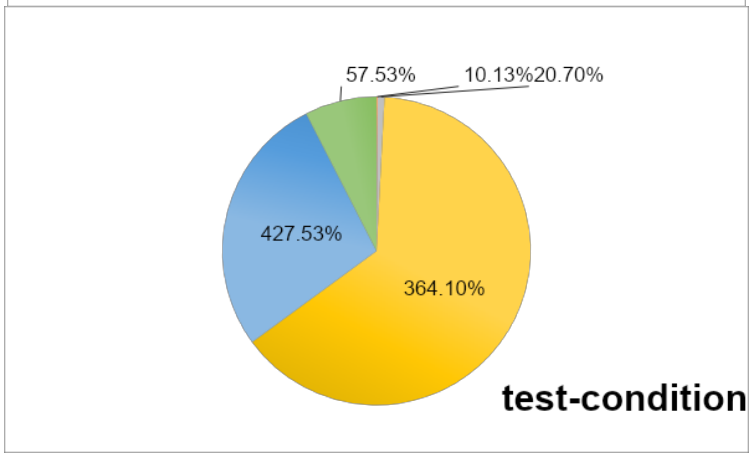
Feature	Max_value	Min_value	Mean_value	Std.dev._value
'dummy"	1	1	1	0
'id"	9835800840	1200021	4558237915	2862593979
'bedrooms"	10	1	3.376666667	0.917203594
'bathroom"	8	0.5	2.114916667	0.78039776
'sqft_living"	12050	380	2087.315167	939.6401686
'sqft_lot"	1164794	520	15581.00233	44345.65547
'floors"	3.5	1	1.484416667	0.5371346
'waterfront"	1	0	0.008166667	0.090007347
'view"	4	0	0.2465	0.783479455
'condition"	5	1	3.416333333	0.64477581
'grade"	13	4	13	1.179919398
'sqft_above"	8570	380	1784.473833	832.5623004
'sqft_basement"	3500	0	302.8413333	455.678088
'yr_built"	2015	1900	1970.727667	29.39074351
'yr_renovated"	2015	0	86.485	406.4041123
'zipcode"	98199	98001	98077.97983	54.00869606
'lat"	47.7775	47.1593	47.56015643	0.137955527
'long"	-121.315	-122.519	-122.215485	0.139344347
'sqft_living15"	5790	399	1981.949833	688.1506881
'sqft_living15"	411962	750	12727.5395	25697.76147

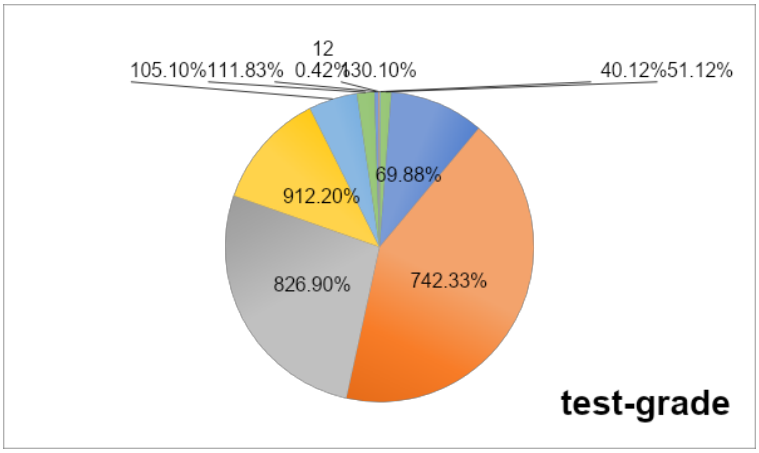
(Percentage for waterfront/condition/grade)

Feature	Category	Percentage
waterfront	0	99.18%
	1	0.82%
condition	0	0.00%
	1	0.13%
	2	0.70%
	3	64.10%
	4	27.53%
	5	7.53%
grade	0	0.00%
	1	0.00%
	2	0.00%
	3	0.00%
	4	0.12%
	5	1.12%
	6	9.88%
	7	42.33%
	8	26.90%
	9	12.20%
	10	5.10%
	11	1.83%
	12	0.42%
	13	0.10%

(Pie Chart)







[PA1_dev]

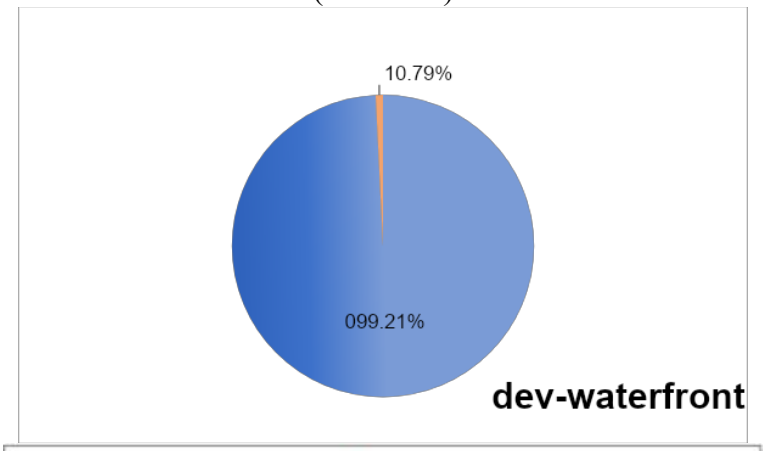
(Range/Mean/Standard Deviation)

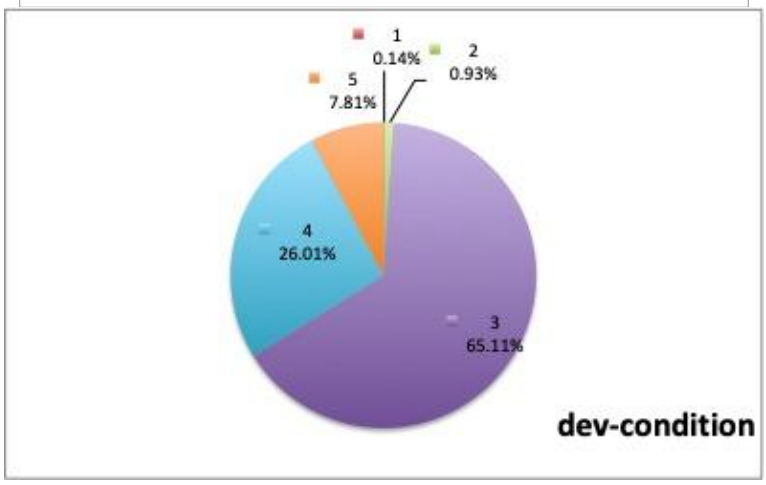
Feature	Max_value	Min_value	Mean_value	Std.dev._value
'dummy"	1	1	1	0
'id"	9842300095	2800031	4593113550	2878330348
'bedrooms"	9	1	3.36591031	0.905330857
'bathrooms"	8	0.75	2.1113543	0.763625449
'sqft_living"	13540	390	2073.00125	906.8430379
'sqft_lot"	1024068	609	14601.3763	38422.54796
'floors"	3.5	1	1.48731463	0.536897898
'waterfront"	1	0	0.00786135	0.088322971
'view"	4	0	0.22994461	0.766416856
'condition"	5	1	3.40414508	0.651306467
'grade"	13	3	7.6482044	1.153382883
'sqft_above"	9410	390	1784.97231	817.0748945
'sqft_basemer	4820	0	288.028944	441.9593419
'yr_built"	2015	1900	1971.06754	29.17265306
'yr_renovated	2015	0	88.0845096	409.9979469
'zipcode"	98199	98001	98077.3121	52.97082981
'lat"	47.7776	47.1622	47.5605231	0.139035664
'long"	-121.315	-122.511	-122.213616	0.140987495
'sqft_living15	6210	670	1977.85939	669.9184123
'sqft_lot15"	434728	651	12812.61	27162.27069
'price"	55.7	0.78	5.37517475	3.561417637

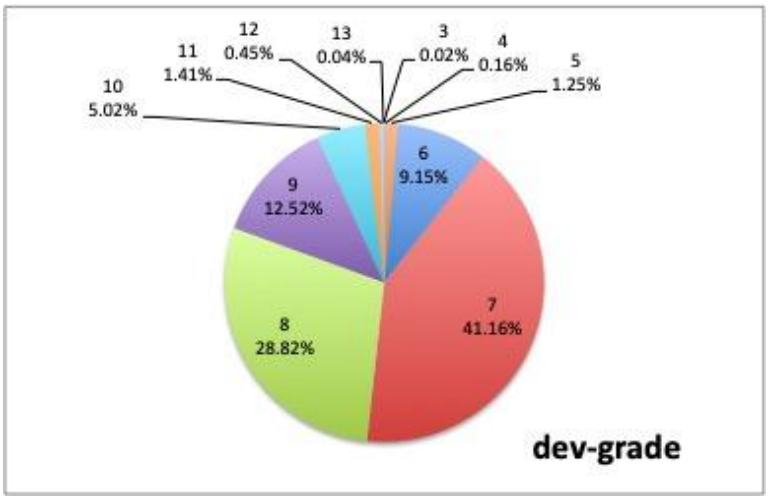
(Percentage for waterfront/condition/grade)

Feature	Category	Percentage
waterfront	0	99.21%
	1	0.79%
condition	0	0.00%
	1	0.14%
	2	0.93%
	3	65.11%
	4	26.01%
	5	7.81%
grade	0	0.00%
	1	0.00%
	2	0.00%
	3	0.02%
	4	0.16%
	5	1.25%
	6	9.15%
	7	41.16%
	8	28.82%
	9	12.52%
	10	5.02%
	11	1.41%
	12	0.45%
	13	0.04%

(Pie Chart)





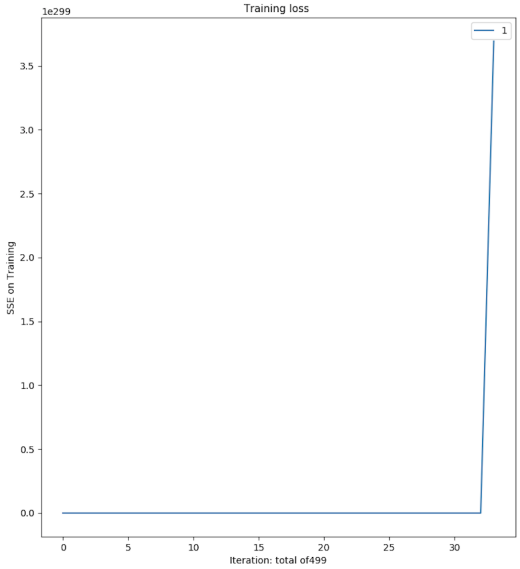


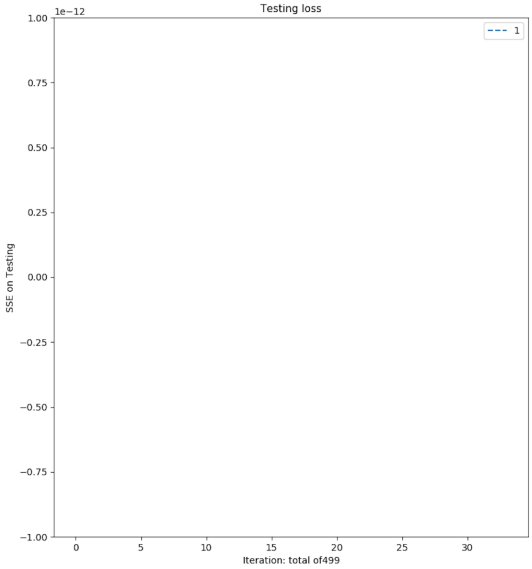
(d) Based on the meaning of the features as well as the statistics, which set of features do you expect to be useful for this task? Why?
All of the features will be useful for this task. Among these features, housing area like sqft_above and bedrooms will be the most important factor affecting housing price. Besides, year built is also important because the house prices will fall as the age of the house increase.

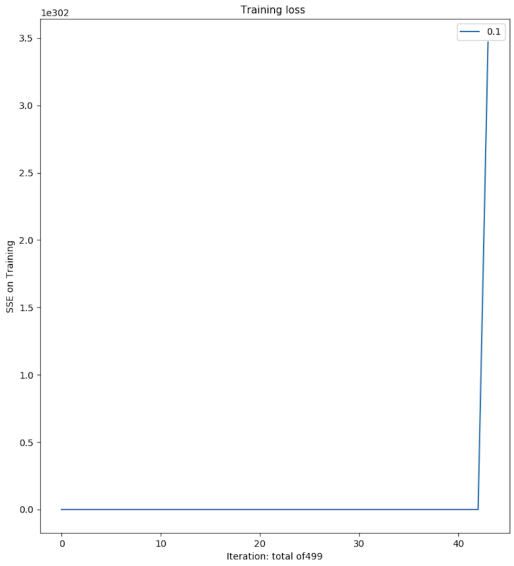
(e) Normalize all features to the range between 0 and 1 using the training data. Note that when you apply the learned model from the normalized data to test data, you should make sure that you are using the same normalizing procedure as used in training.
Normalize each feature according to the formula:
$Z_i = \frac{x_i - min\ (x)}{(x) - min\ (x)}$
Where x = (x_1,…,x_n) and z_i is i^{th} normalized data.

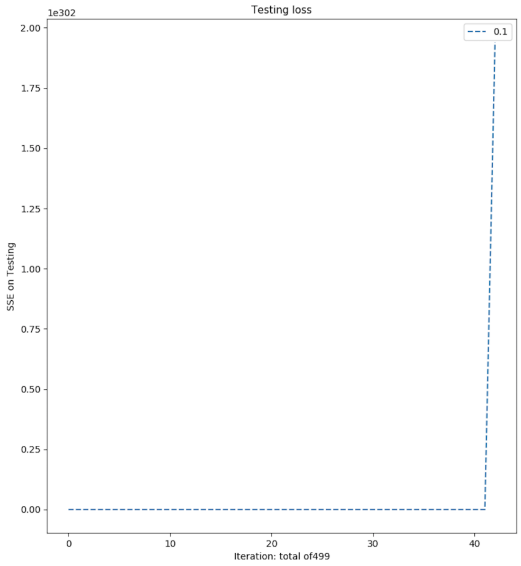
Part 1

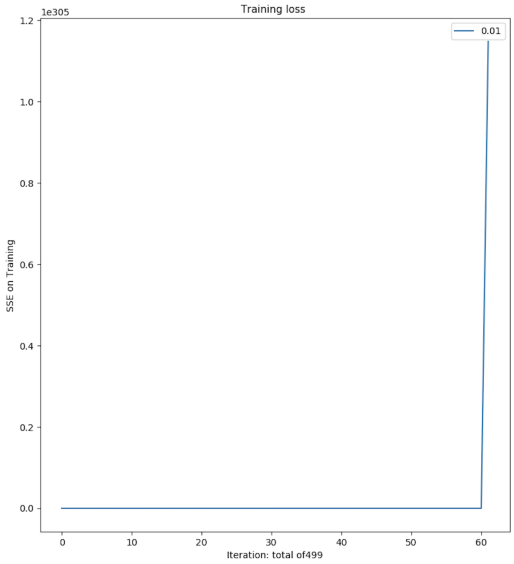
(a) Which learning rate or learning rates did you observe to be good for this particular dataset? What learning do rates make the gradient descent explode? Report your observations together with some example curves showing the training SSE as a function of training iterations and its convergence or non-convergence behaviors.

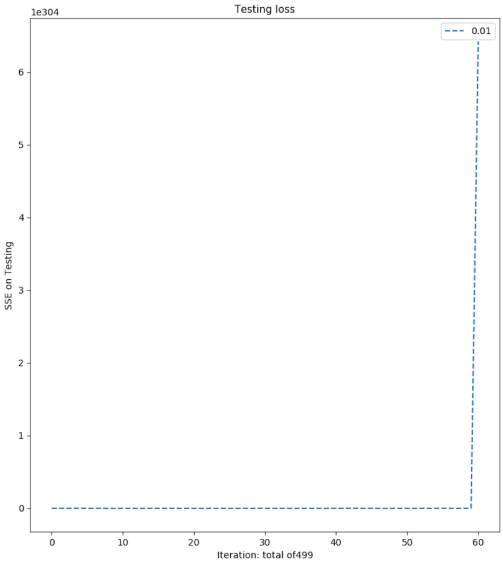


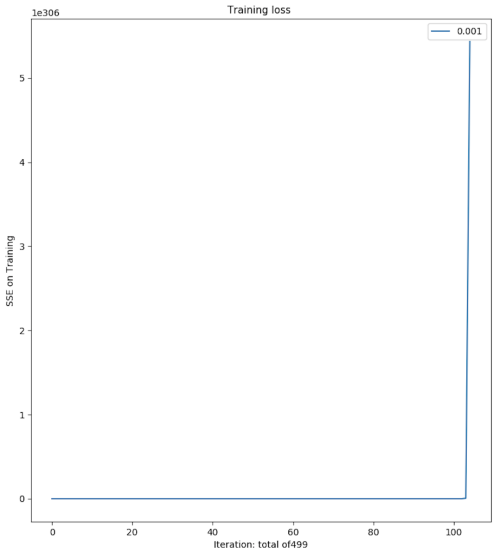


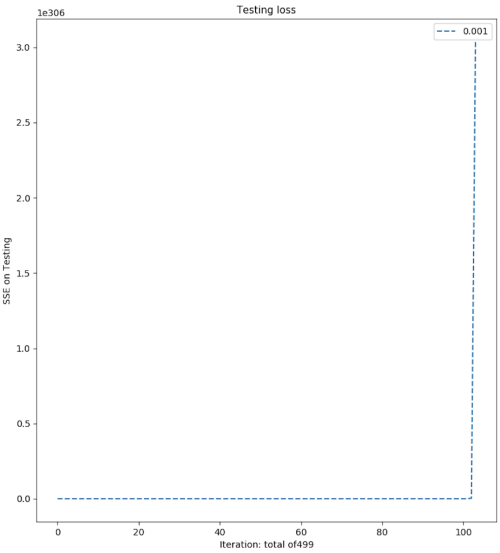


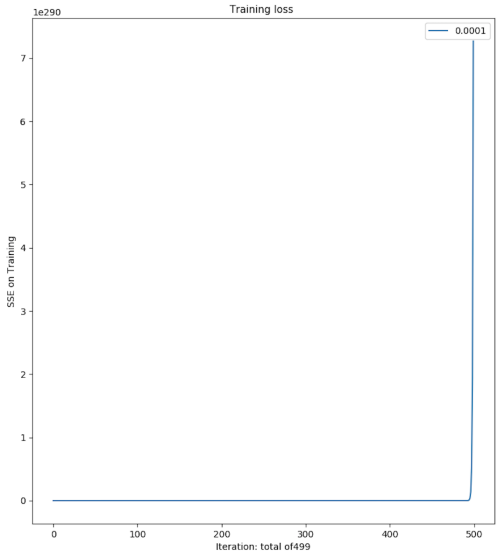


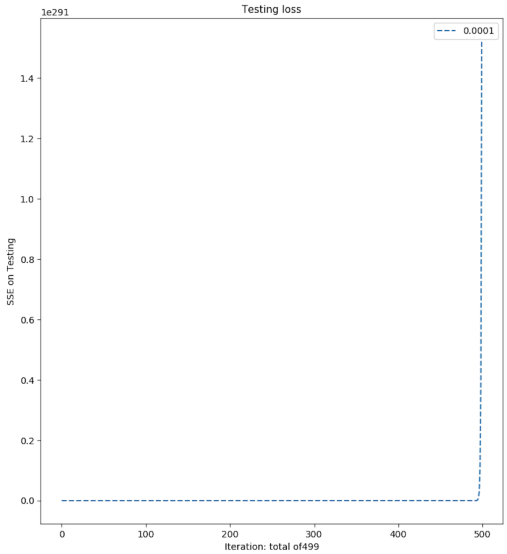

Training loss
1e305
0.01
SSE on Training
Iteration: total of499

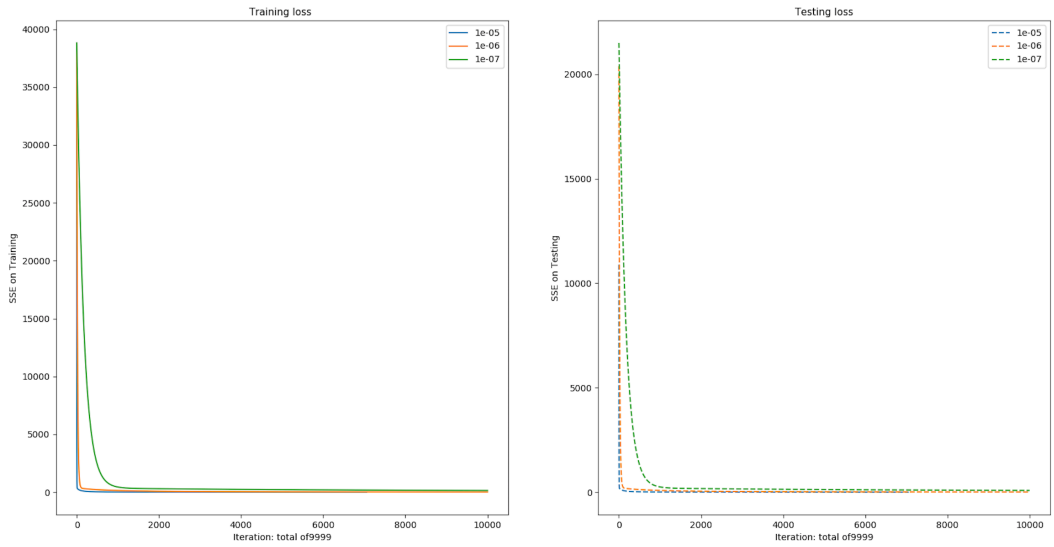

Testing loss
1e304
0.01
SSE on Testing
Iteration: total of499


Training loss
1e306
0.001
SSE on Training
Iteration: total of499


Testing loss
1e306
0.001
SSE on Testing
Iteration: total of499


Training loss
1e290
0.0001
SSE on Training
Iteration: total of499


Testing loss
1e291
0.0001
SSE on Testing
Iteration: total of499



Learning rate 10^{-5} is good for this particular dataset. Learning rates 10^{0}, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4} make the gradient descent explode. Learning rates 10^{-5}, 10^{-6}, 10^{-7} are appropriate for this dataset, but 10^{-6}, 10^{-7} lead to overly slow convergence. We use threshold = 5 as a convergence criterion.

(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	1e-5	1e-6	1e-7
SSE (training data)	4.842	19.281	145.780
SSE (validation data)	2.656	10.659	83.920
Number of Iteration	5068	10000	10000
Convergence	Yes	No	No
Norm	5	41.594	320.923

From de data table, we notice that the SSE on the validation data is about half of the SSE on the training data. This is because the number of samples of the validation data is about half of the number of samples of the training data.

(c) Use the validation data to pick the best converged solution, and report the learned weights for each feature. Which features are the most important in deciding the house prices according to the learned weights? Compare them to your pre-analysis results (Part 0 (d)).

We pick learning rate = 1e-5. The learning weight is
[[-0.05830906] [0.00452398] [-0.03591693] [0.14931411] [-0.02379136] [0.26006589] [0.00312821] [0.10792436] [0.02215048] [0.01837775]

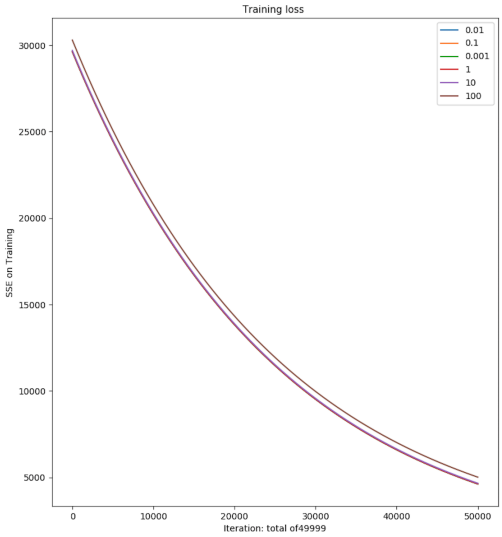
[0.14528191] [0.06346939] [0.01884709] [-0.05528384] [0.00078988] [-0.00835936] [0.05587633] [-0.0293089] [0.06337444] [0.0588668]]

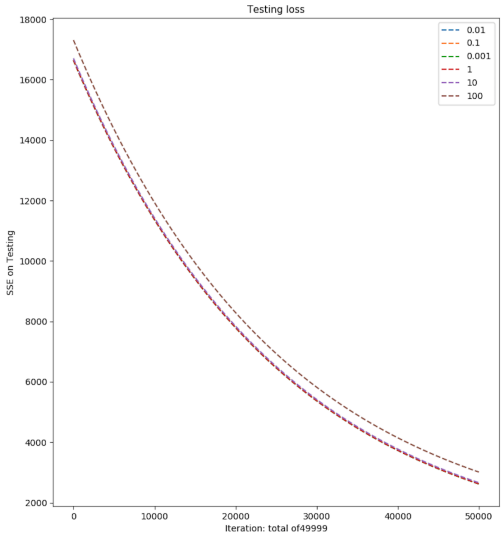
Feature	Weights	Feature	Weights
dummy	-0.05830906	grade	0.14528191
date	0.00452398	sqft_above	0.06346939
bedrooms	-0.03591693	sqft_basement	0.01884709
bathrooms	0.14931411	yr_built	-0.05528384
sqft_living	-0.02379136	yr_renovated	0.00078988
sqft_lot	0.26006589	zipcode	-0.00835936
floors	0.00312821	lat	0.05587633
waterfront	0.10792436	long	-0.0293089
view	0.02215048	sqft_living15	0.06337444
condition	0.01837775	sqft_lot15	0.0588668

According to the learning rate, sqft_lot is the most important feature in deciding the house prices. This is in line with our expectations in Part 0 (d). The housing area is indeed the most important factor in determining housing prices。

Part 2

We have talked about λ=0 in part1.
Experiment with the different λ value as below:





[λ = 0.1]
"0.1": {"weight": "[[0.58019833]\n [0.51172306]\n [0.56874488]\n [0.67943565]\n
[0.36235025]\n [0.63799061]\n [0.77759828]\n [0.00630593]\n [0.00576499]\n
[0.01824464]\n [0.54712639]\n [-0.06181068]\n [-0.13289606]\n [0.66716371]\n
[0.27951185]\n [0.08686863]\n [-0.14798441]\n [0.56467817]\n [0.45055675]]"

[λ = 0.01]
{"0.01": {"weight": "[[0.58019833]\n [0.51172306]\n [0.56874488]\n
[0.67943565]\n [0.36235025]\n [0.63799061]\n [0.77759828]\n [0.00630593]\n
[0.00576499]\n [0.01824464]\n [0.54712639]\n [-0.06181068]\n [-0.13289606]\n
[0.66716371]\n [0.27951185]\n [0.08686863]\n [-0.14798441]\n [0.56467817]\n
[0.45055675]]",

[λ = 0.001]
"0.001": {"weight": "[[0.58019833]\n [0.51172306]\n [0.56874488]\n
[0.67943565]\n [0.36235025]\n [0.63799061]\n [0.77759828]\n [0.00630593]\n
[0.00576499]\n [0.01824464]\n [0.54712639]\n [-0.06181068]\n [-0.13289606]\n
[0.66716371]\n [0.27951185]\n [0.08686863]\n [-0.14798441]\n [0.56467817]\n
[0.45055675]]"

[λ = 1]
"1": {"weight": "[[0.58019833]\n [0.51172306]\n [0.56874488]\n [0.67943565]\n
[0.36235025]\n [0.63799061]\n [0.77759828]\n [0.00630593]\n [0.00576499]\n
[0.01824464]\n [0.54712639]\n [-0.06181068]\n [-0.13289606]\n [0.66716371]\n
[0.27951185]\n [0.08686863]\n [-0.14798441]\n [0.56467817]\n [0.45055675]]"

[λ = 10]
"10": {"weight": "[[0.58019833]\n [0.51172306]\n [0.56874488]\n [0.67943565]\n

[0.36235025]\n [0.63799061]\n [0.77759828]\n [0.00630593]\n [0.00576499]\n
[0.01824464]\n [0.54712639]\n [-0.06181068]\n [-0.13289606]\n [0.66716371]\n
[0.27951185]\n [0.08686863]\n [-0.14798441]\n [0.56467817]\n [0.45055675]]"

[λ = 100]
"100": {"weight": "[[0.58019833]\n [0.51172306]\n [0.56874488]\n
[0.67943565]\n [0.36235025]\n [0.63799061]\n [0.77759828]\n [0.00630593]\n
[0.00576499]\n [0.01824464]\n [0.54712639]\n [-0.06181068]\n [-0.13289606]\n
[0.66716371]\n [0.27951185]\n [0.08686863]\n [-0.14798441]\n [0.56467817]\n
[0.45055675]]"

Regularization Factor (λ)	**SSE for training data set**	**SSE for validation data set**
0.1	4622	2621
0.01	4622	2621
0.001	4622	2621
1	4626	2625
10	4661	2660
100	5014	3013

(a) What trend do you observe from the training SSE as we change λ value?
Based on the data, if focusing on the point of the iteration. The SSE will increase when the λ (=100) is bigger.

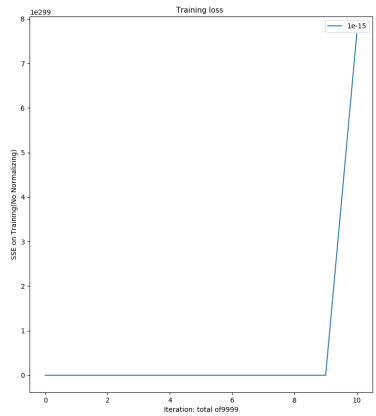
(b) What tread do you observe from the validation SSE?
As for the validation SSE, there is the same situation as the training SSE. It remains same value when the λ is small. However, when we increase the λ to the 100, the SSE will change.

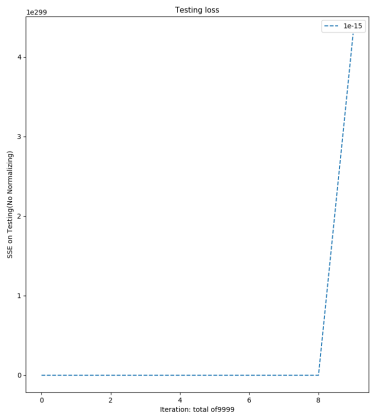
(c) Provide an explanation for the observed behaviors.
The data from training SSE and validation SSE conclude that linear regression has good fit when the value of λ is {0; 0.001; 0.01; 0.1; 1; 10}. If the value of the lambda becomes higher, the model will cause under-fitting.

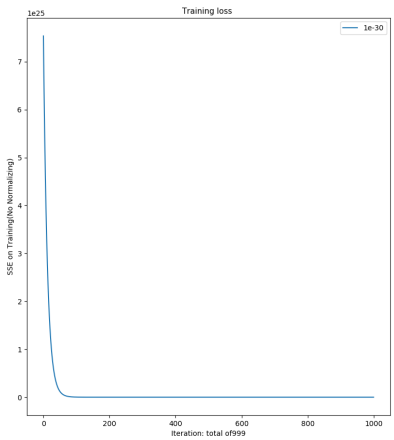
(d) What features get turned o_ for _= 10, 0.01 and 0?
Comparing the weights of features for λ= 0, 10-2 and 10. The yr_built feature is the smallest weight in λ = 0 and 10-2. Thus, the yr_built can be turned off.

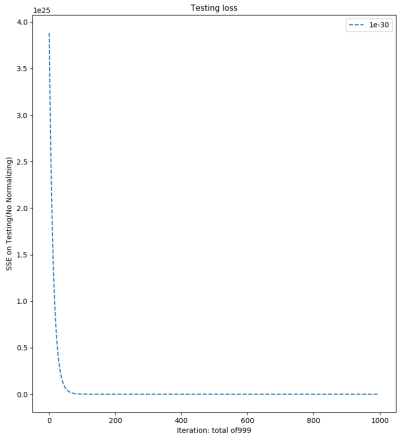
Part 3

Learning Rate (lr)	**SSE for training data set**	**SSE for validation data set**
1	N/A	N/A
0	N/A	N/A
10^-3	N/A	N/A
10^-6	N/A	N/A
10^-9	N/A	N/A
10^-15	nearly explode	nearly explode









(a) What do you observe?
The non-normalized learning rate is easy to explode out, it needs smaller learning rate to prevent.

(b)Specify the learning rate value (if any) that prevents the gradient descent from

exploding?
The learning rate has to be smaller than 10^(-15).

(c) Compare between using the normalized and the non-normalized versions of the data. Which one is easier to train and why?
Non-normalized data is easy to explode like the learning rate we input is 10^(-15), unless the learning rate is small enough (10^(-30)). Thus, we can conclude that it is hard to get a convergence result when we train the non-normalized data. The normalized data is better for us to train.