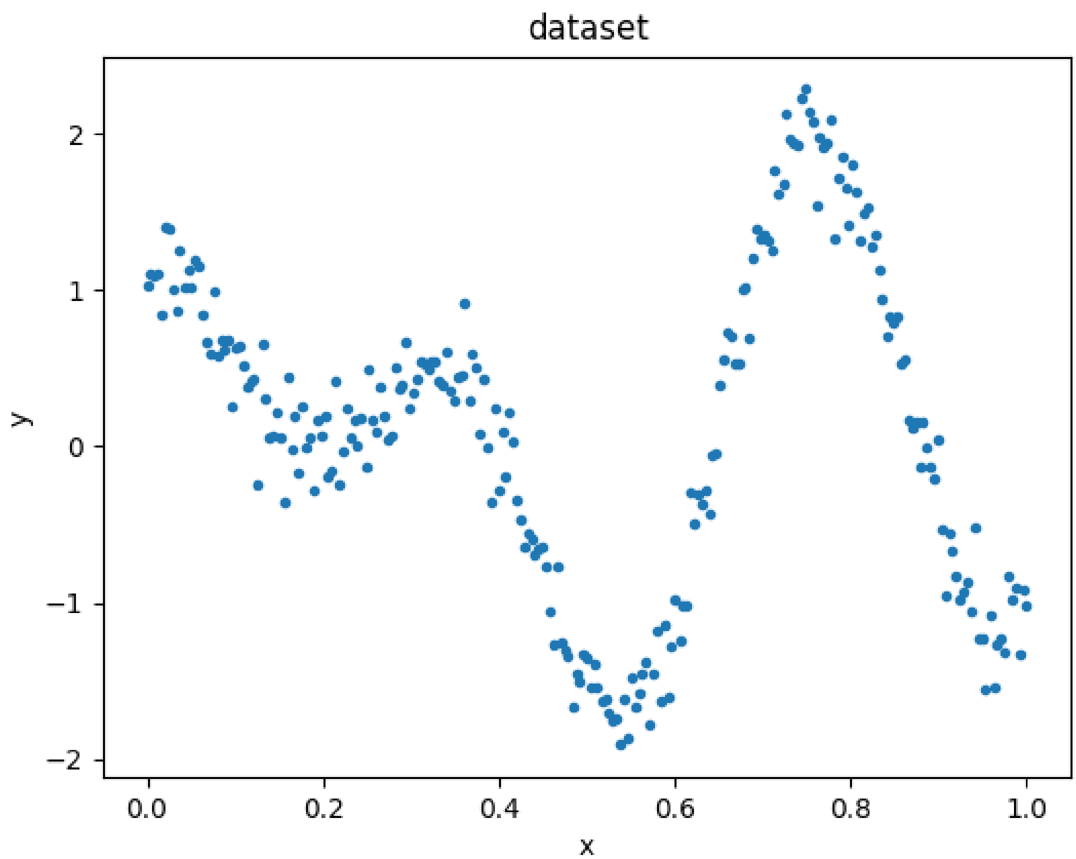
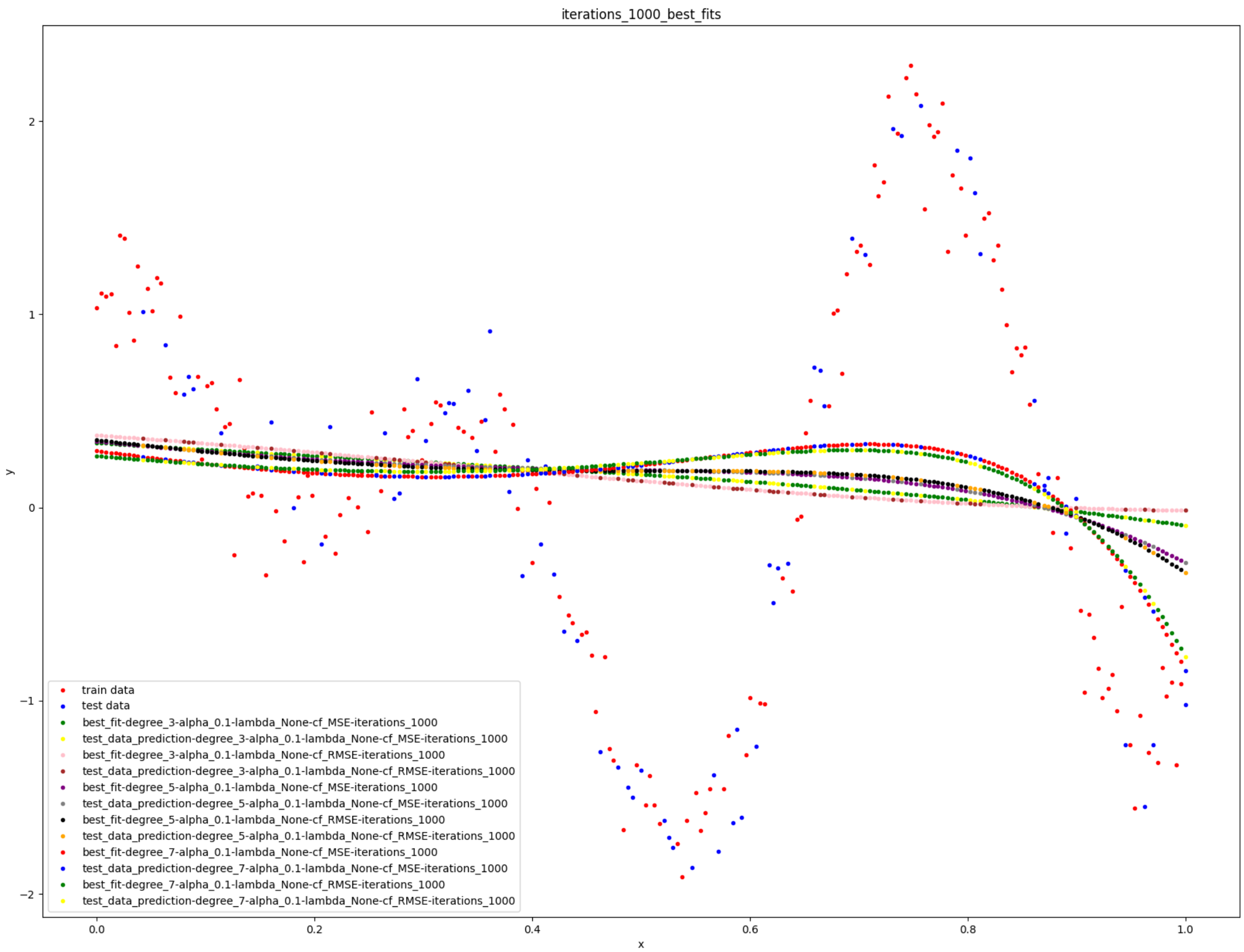
Q1
problem: performing linear regression on a signal

a- plot the data

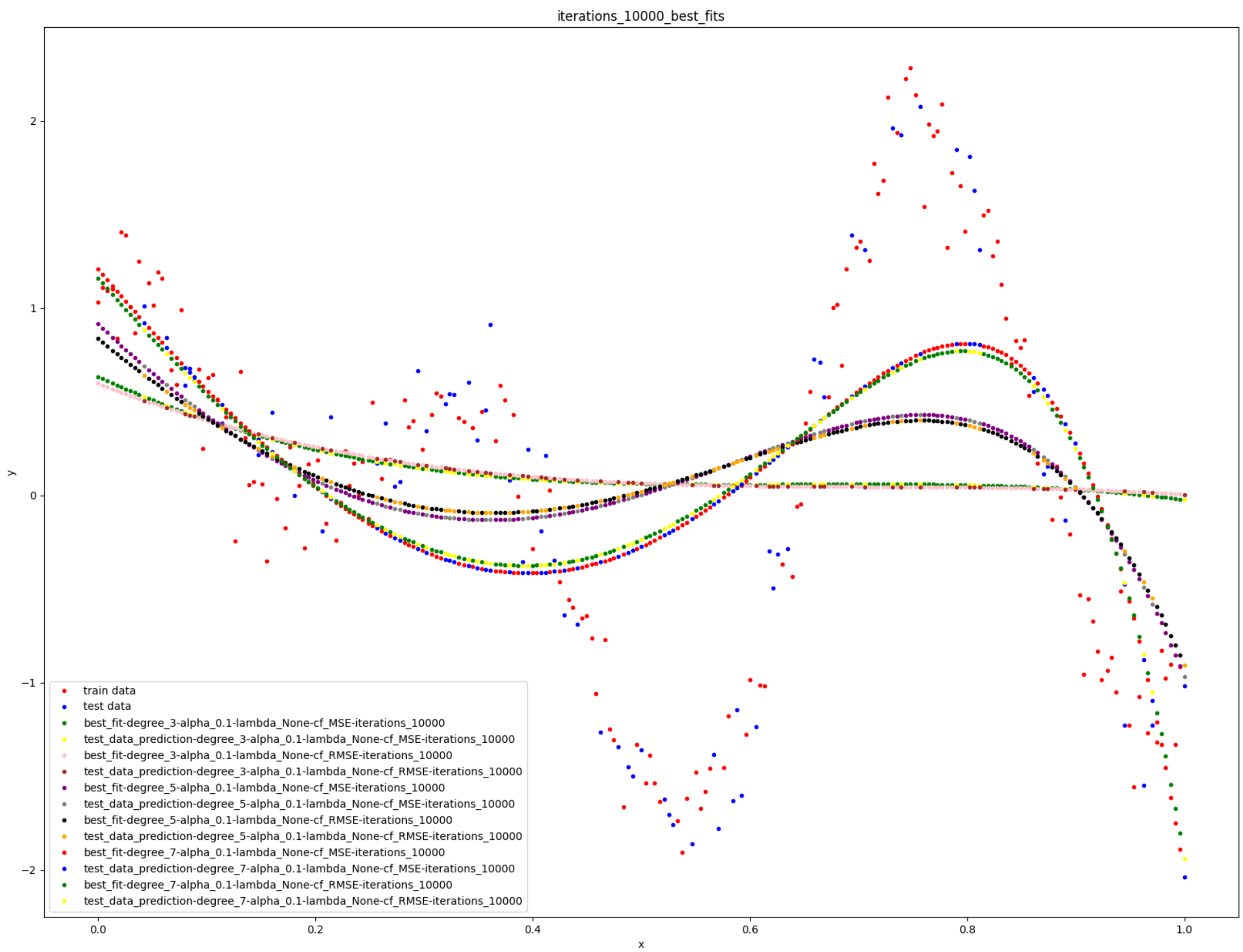


b- use gradient descent algorithm to fit a model on data by adding polynomial features with degrees 3, 5, and 7. report the training and testing costs. repeat the algorithm for 1000 and 10000 iterations. does overfitting occur? in which cases? use cost functions MSE and RMSE to evaluate the models and compare the results.

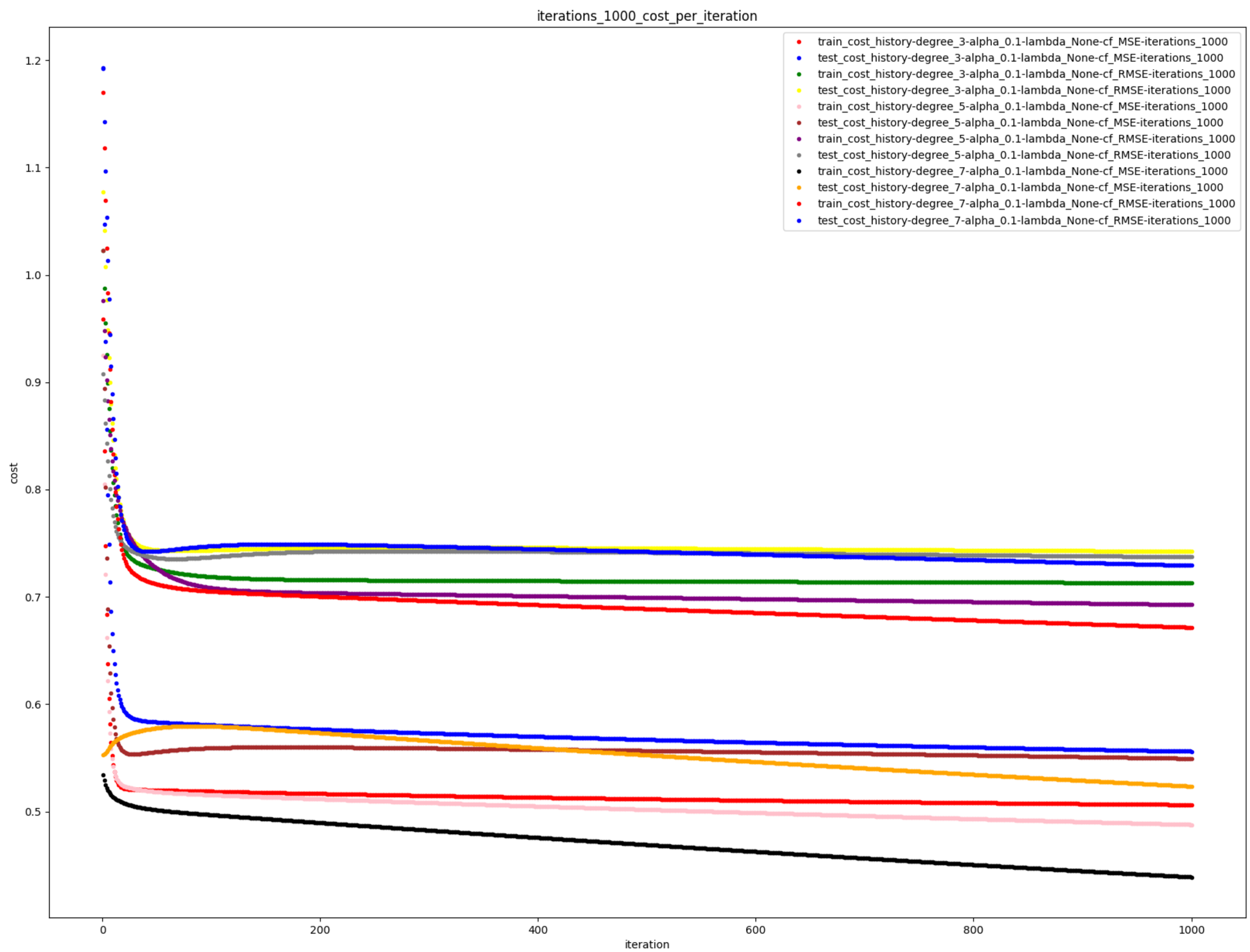
best fits for combinations of degree and cost function:
file names:
iterations_1000_best_fits.png
iterations_10000_best_fits.png

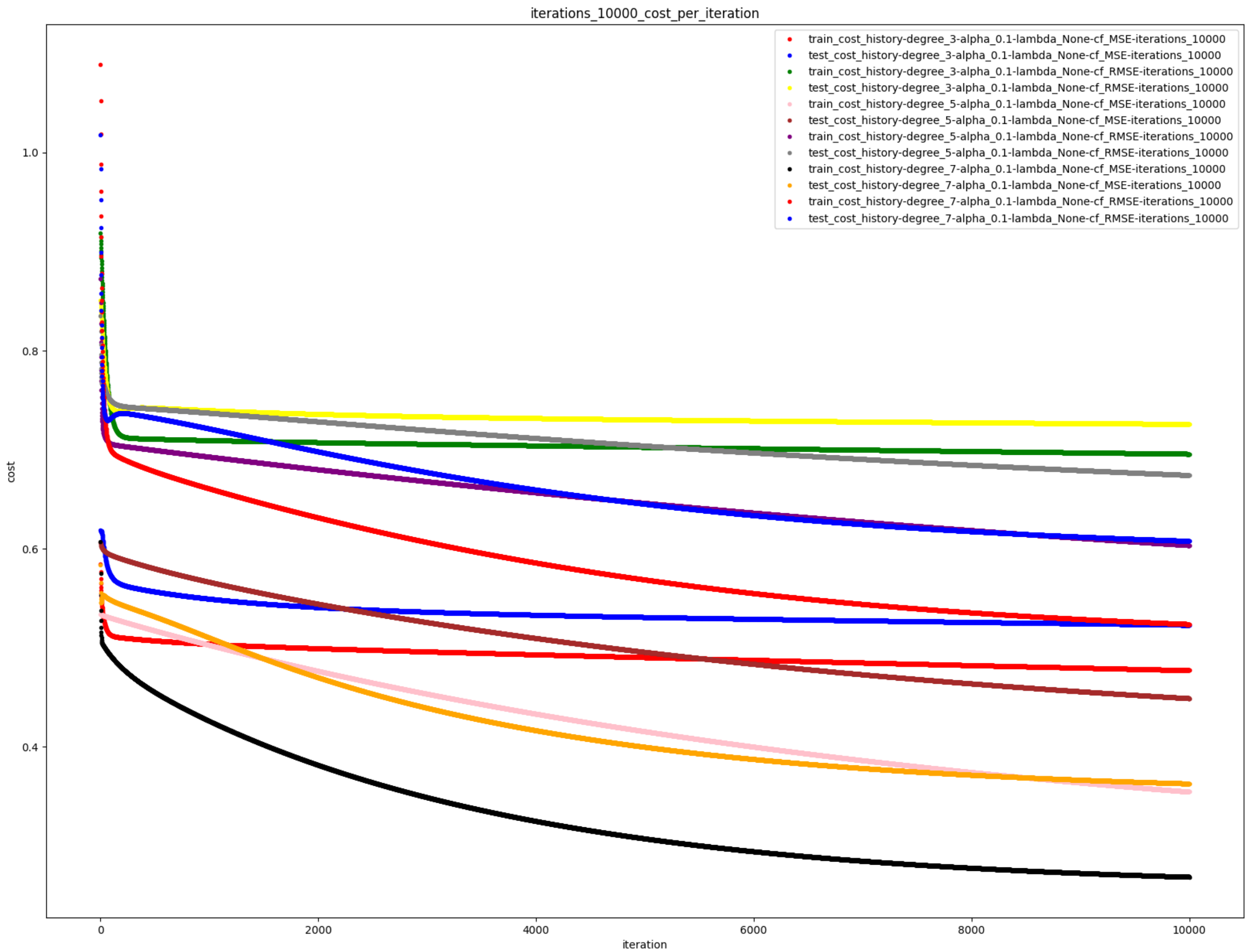


iterations_1000_best_fits
x
y
train data
test data
best_fit-degree_3-alpha_0.1-lambda_None-cf_MSE-iterations_1000
test_data_prediction-degree_3-alpha_0.1-lambda_None-cf_MSE-iterations_1000
best_fit-degree_3-alpha_0.1-lambda_None-cf_RMSE-iterations_1000
test_data_prediction-degree_3-alpha_0.1-lambda_None-cf_RMSE-iterations_1000
best_fit-degree_5-alpha_0.1-lambda_None-cf_MSE-iterations_1000
test_data_prediction-degree_5-alpha_0.1-lambda_None-cf_MSE-iterations_1000
best_fit-degree_5-alpha_0.1-lambda_None-cf_RMSE-iterations_1000
test_data_prediction-degree_5-alpha_0.1-lambda_None-cf_RMSE-iterations_1000
best_fit-degree_7-alpha_0.1-lambda_None-cf_MSE-iterations_1000
test_data_prediction-degree_7-alpha_0.1-lambda_None-cf_MSE-iterations_1000
best_fit-degree_7-alpha_0.1-lambda_None-cf_RMSE-iterations_1000
test_data_prediction-degree_7-alpha_0.1-lambda_None-cf_RMSE-iterations_1000

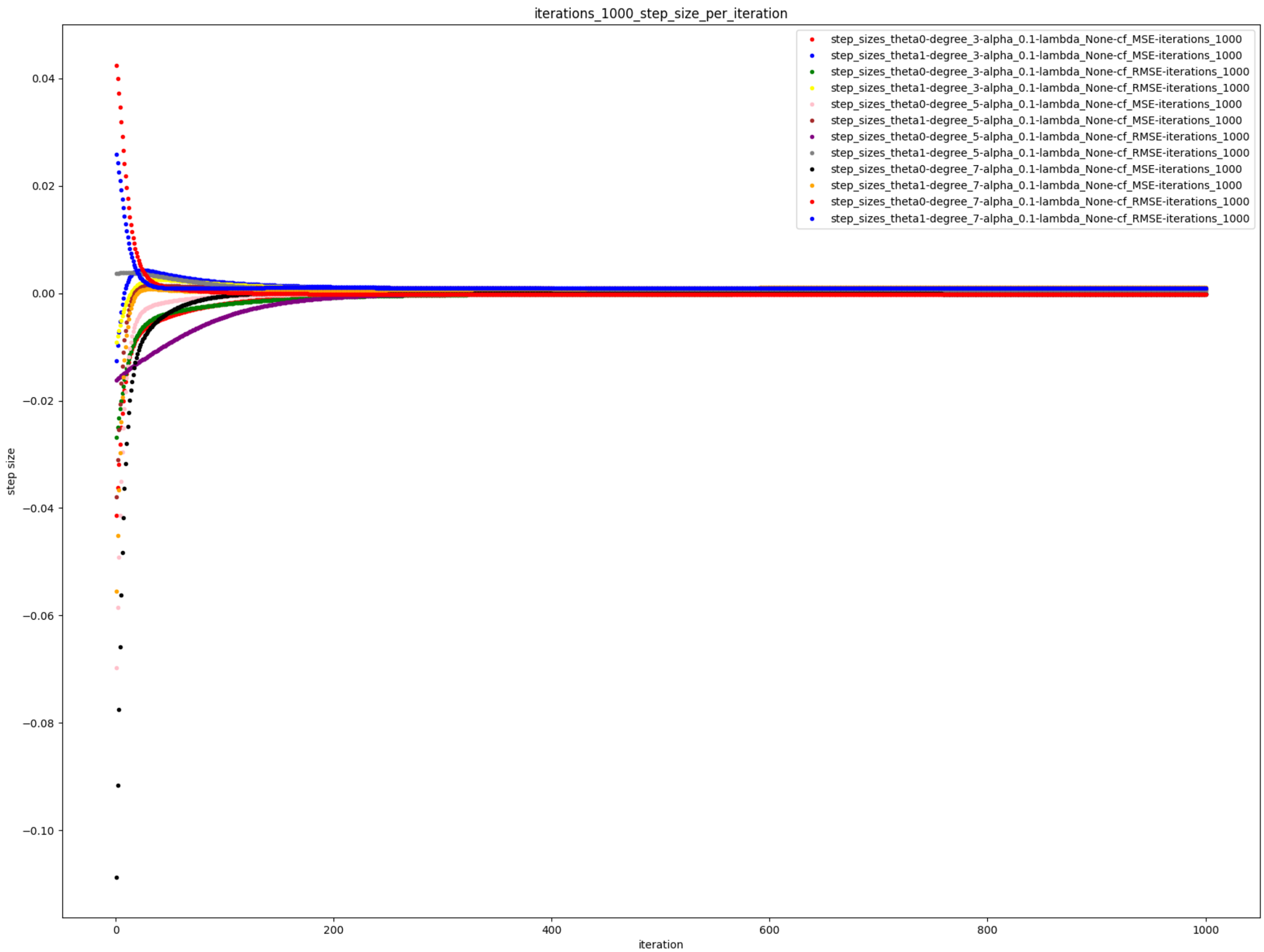

iterations_10000_best_fits
y
x
2
1
0
−1
−2
0.0
0.2
0.4
0.6
0.8
1.0
train data
test data
best_fit-degree_3-alpha_0.1-lambda_None-cf_MSE-iterations_10000
test_data_prediction-degree_3-alpha_0.1-lambda_None-cf_MSE-iterations_10000
best_fit-degree_3-alpha_0.1-lambda_None-cf_RMSE-iterations_10000
test_data_prediction-degree_3-alpha_0.1-lambda_None-cf_RMSE-iterations_10000
best_fit-degree_5-alpha_0.1-lambda_None-cf_MSE-iterations_10000
test_data_prediction-degree_5-alpha_0.1-lambda_None-cf_MSE-iterations_10000
best_fit-degree_5-alpha_0.1-lambda_None-cf_RMSE-iterations_10000
test_data_prediction-degree_5-alpha_0.1-lambda_None-cf_RMSE-iterations_10000
best_fit-degree_7-alpha_0.1-lambda_None-cf_MSE-iterations_10000
test_data_prediction-degree_7-alpha_0.1-lambda_None-cf_MSE-iterations_10000
best_fit-degree_7-alpha_0.1-lambda_None-cf_RMSE-iterations_10000
test_data_prediction-degree_7-alpha_0.1-lambda_None-cf_RMSE-iterations_10000

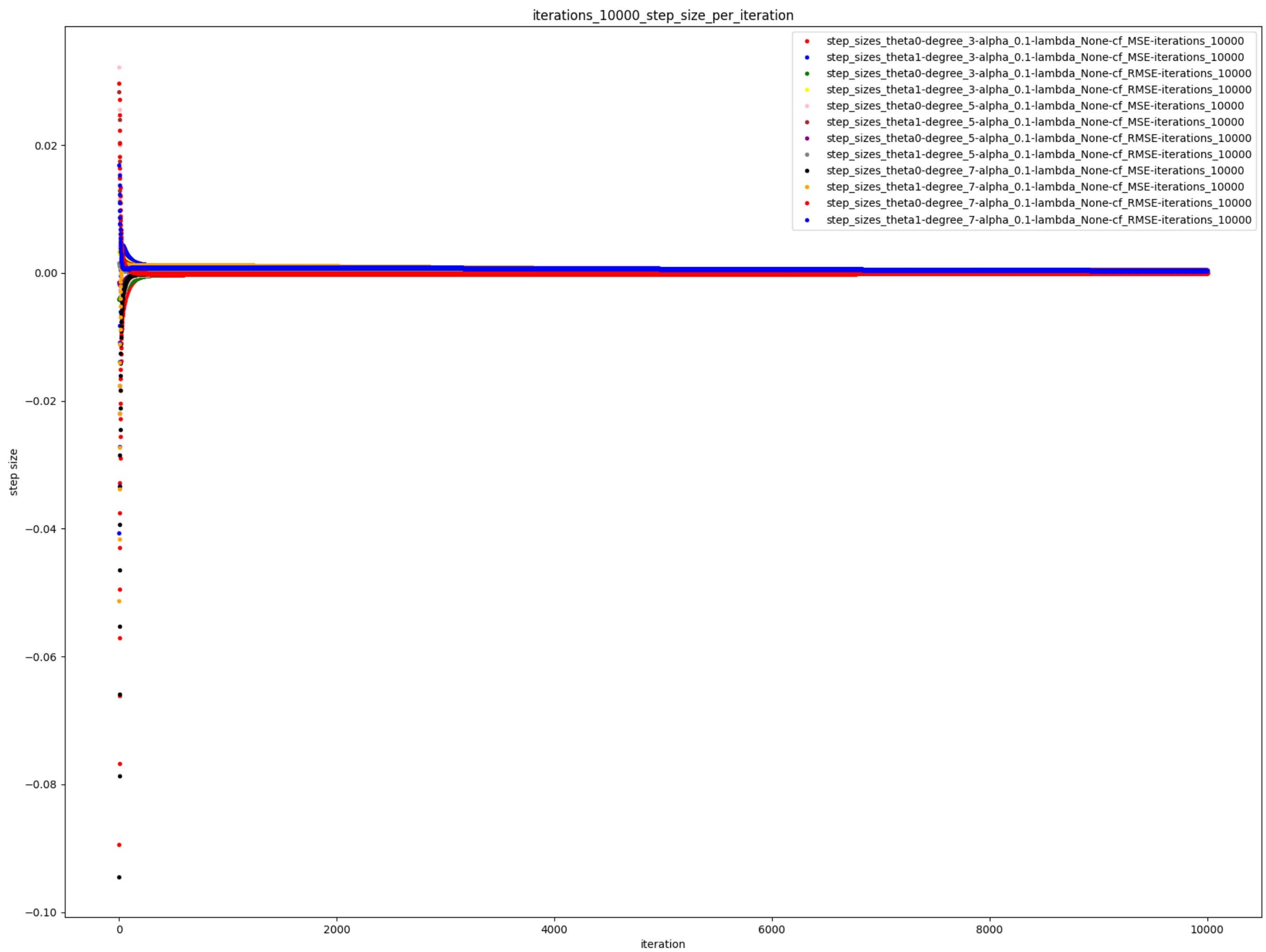
training and testing costs for combinations of degree and cost function:
file names:
iterations_1000_cost_per_iteration.png
iterations_10000_cost_per_iteration.png




iterations_10000_cost_per_iteration
train_cost_history-degree_3-alpha_0.1-lambda_None-cf_MSE-iterations_10000
test_cost_history-degree_3-alpha_0.1-lambda_None-cf_MSE-iterations_10000
train_cost_history-degree_3-alpha_0.1-lambda_None-cf_RMSE-iterations_10000
test_cost_history-degree_3-alpha_0.1-lambda_None-cf_RMSE-iterations_10000
train_cost_history-degree_5-alpha_0.1-lambda_None-cf_MSE-iterations_10000
test_cost_history-degree_5-alpha_0.1-lambda_None-cf_MSE-iterations_10000
train_cost_history-degree_5-alpha_0.1-lambda_None-cf_RMSE-iterations_10000
test_cost_history-degree_5-alpha_0.1-lambda_None-cf_RMSE-iterations_10000
train_cost_history-degree_7-alpha_0.1-lambda_None-cf_MSE-iterations_10000
test_cost_history-degree_7-alpha_0.1-lambda_None-cf_MSE-iterations_10000
train_cost_history-degree_7-alpha_0.1-lambda_None-cf_RMSE-iterations_10000
test_cost_history-degree_7-alpha_0.1-lambda_None-cf_RMSE-iterations_10000
cost
1.0
0.8
0.6
0.4
0
2000
4000
6000
8000
10000
iteration

step sizes for combinations of degree and cost function:
file names:
iterations_1000_step_size_per_iteration.png
iterations_10000_step_size_per_iteration.png





iterations_10000_step_size_per_iteration
step_sizes_theta0-degree_3-alpha_0.1-lambda_None-cf_MSE-iterations_10000
step_sizes_theta1-degree_3-alpha_0.1-lambda_None-cf_MSE-iterations_10000
step_sizes_theta0-degree_3-alpha_0.1-lambda_None-cf_RMSE-iterations_10000
step_sizes_theta1-degree_3-alpha_0.1-lambda_None-cf_RMSE-iterations_10000
step_sizes_theta0-degree_5-alpha_0.1-lambda_None-cf_MSE-iterations_10000
step_sizes_theta1-degree_5-alpha_0.1-lambda_None-cf_MSE-iterations_10000
step_sizes_theta0-degree_5-alpha_0.1-lambda_None-cf_RMSE-iterations_10000
step_sizes_theta1-degree_5-alpha_0.1-lambda_None-cf_RMSE-iterations_10000
step_sizes_theta0-degree_7-alpha_0.1-lambda_None-cf_MSE-iterations_10000
step_sizes_theta1-degree_7-alpha_0.1-lambda_None-cf_MSE-iterations_10000
step_sizes_theta0-degree_7-alpha_0.1-lambda_None-cf_RMSE-iterations_10000
step_sizes_theta1-degree_7-alpha_0.1-lambda_None-cf_RMSE-iterations_10000
step size
0.02
0.00
−0.02
−0.04
−0.06
−0.08
−0.10
0
2000
4000
6000
8000
10000
iteration

final results:

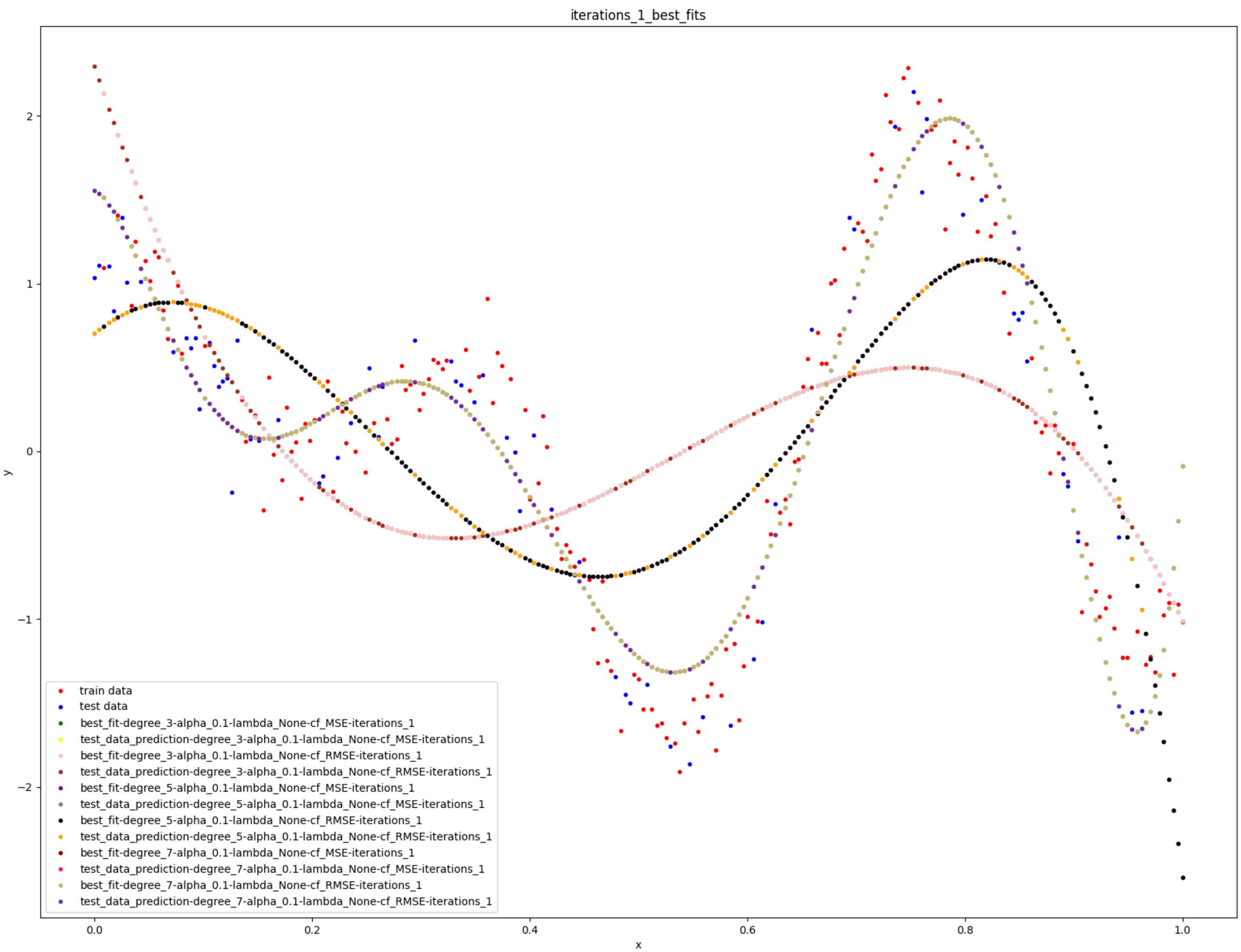
All models plus other figures are available in the folder q1-section_b-outputs

```
head of shuffled data:        x      y
0  0.014  1.034
1  0.934 -0.866
2  0.557 -1.477
3  0.913 -0.552
4  0.669  0.710
head of normalized shuffled data:           0
0  0.000000
1  0.933063
2  0.550710
3  0.911765
4  0.664300
train data = 167 rows, test data = 72 rows
model:algorithm: gd- train_cost: 0.5479277278309438 test_cost: 0.438441946654079 cf: MSE iterations: 1000 alpha: 0.1 lambda: None degree: 3
theta: [ 0.37397914 -0.82937209  0.12970555  0.44660593]
model:algorithm: gd- train_cost: 0.5266379917458248 test_cost: 0.4195140867665488 cf: MSE iterations: 10000 alpha: 0.1 lambda: None degree: 3
theta: [ 0.63231344 -2.72648856  3.37059836 -1.1042046 ]
model:algorithm: gd- train_cost: 0.7397905809353015 test_cost: 0.6616729079202487 cf: RMSE iterations: 1000 alpha: 0.1 lambda: None degree: 3
theta: [ 0.3981128  -0.92714201  0.13877864  0.53583039]
model:algorithm: gd- train_cost: 0.7265350589188025 test_cost: 0.6485263210612372 cf: RMSE iterations: 10000 alpha: 0.1 lambda: None degree: 3
theta: [ 0.62086519 -2.62418527  3.15959699 -0.98177481]
model:algorithm: gd- train_cost: 0.5239195181838167 test_cost: 0.4180416533954612 cf: MSE iterations: 1000 alpha: 0.1 lambda: None degree: 5
theta: [ 0.33762111 -1.15400089  1.16057732  0.94593459 -0.65612197 -0.85428306]
model:algorithm: gd- train_cost: 0.4063909423277386 test_cost: 0.31830229776634383 cf: MSE iterations: 10000 alpha: 0.1 lambda: None degree: 5
theta: [ 0.93379626 -5.60895279  5.19016096  6.00165706 -0.75970164 -6.59245332]
model:algorithm: gd- train_cost: 0.7294670569886603 test_cost: 0.6523219359458232 cf: RMSE iterations: 1000 alpha: 0.1 lambda: None degree: 5
theta: [ 0.27243451 -0.68979493  0.60106587  0.43671916  0.36356945 -1.19974868]
model:algorithm: gd- train_cost: 0.6552851983755722 test_cost: 0.5806218329583156 cf: RMSE iterations: 10000 alpha: 0.1 lambda: None degree: 5
theta: [ 0.80086666 -4.66966823  4.80778169  3.59662622  0.34254498 -5.56068091]
model:algorithm: gd- train_cost: 0.46410293586548307 test_cost: 0.36946327078035684 cf: MSE iterations: 1000 alpha: 0.1 lambda: None degree: 7
theta: [ 0.35977263 -1.56785286  1.21886155  1.4120172   0.79136744  0.35363735
 -1.3169317  -2.06082401]
model:algorithm: gd- train_cost: 0.3106182670455713 test_cost: 0.25519704389938497 cf: MSE iterations: 10000 alpha: 0.1 lambda: None degree: 7
theta: [ 1.27351664 -7.26334386  4.37440503  6.26519022  4.12823887  0.655642
 -3.54549145 -7.88907744]
model:algorithm: gd- train_cost: 0.7140265512098588 test_cost: 0.6388962555409986 cf: RMSE iterations: 1000 alpha: 0.1 lambda: None degree: 7
theta: [ 0.22725389 -0.74455831  1.07293005 -0.0112628   0.83612081  0.01374329
 -1.5719236  -0.38164507]
model:algorithm: gd- train_cost: 0.5707599305632259 test_cost: 0.5126187158527249 cf: RMSE iterations: 10000 alpha: 0.1 lambda: None degree: 7
theta: [ 1.12482529 -6.40229499  4.12120373  5.21311375  4.10373227 -0.2469587
 -3.3748398  -6.32077908]
```

c) solve the previous problem with the normal equation.

best fits:

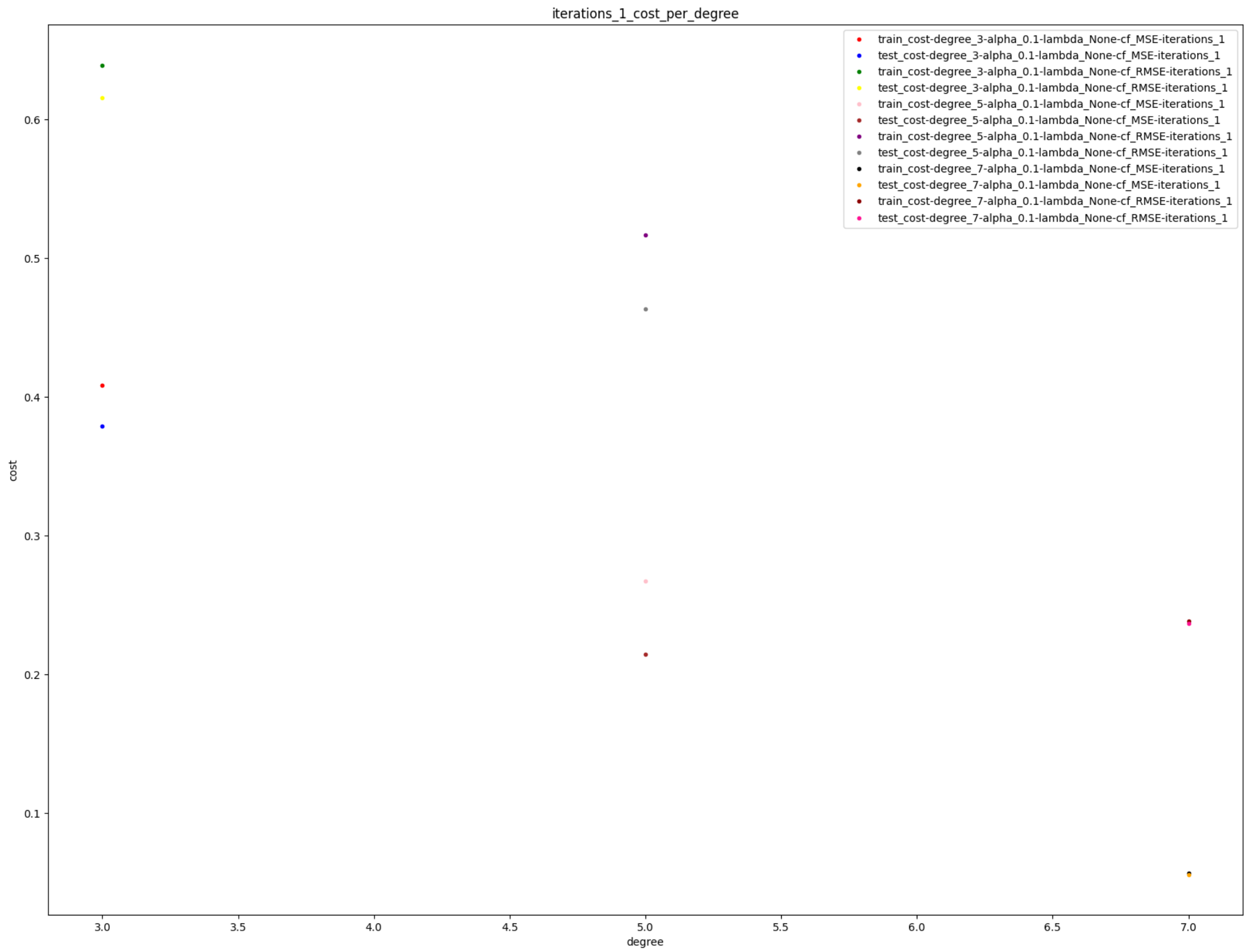
file name: iterations_1_best_fits.png



note: assume there is no **alpha** and **iterations** in the figure.

training and testing costs:

file name: iterations_1_cost_per_degree.png



note: assume there is no **alpha** and **iterations** in the figure.

final results:

All models plus other figures are available in the folder q1-section_c-outputs

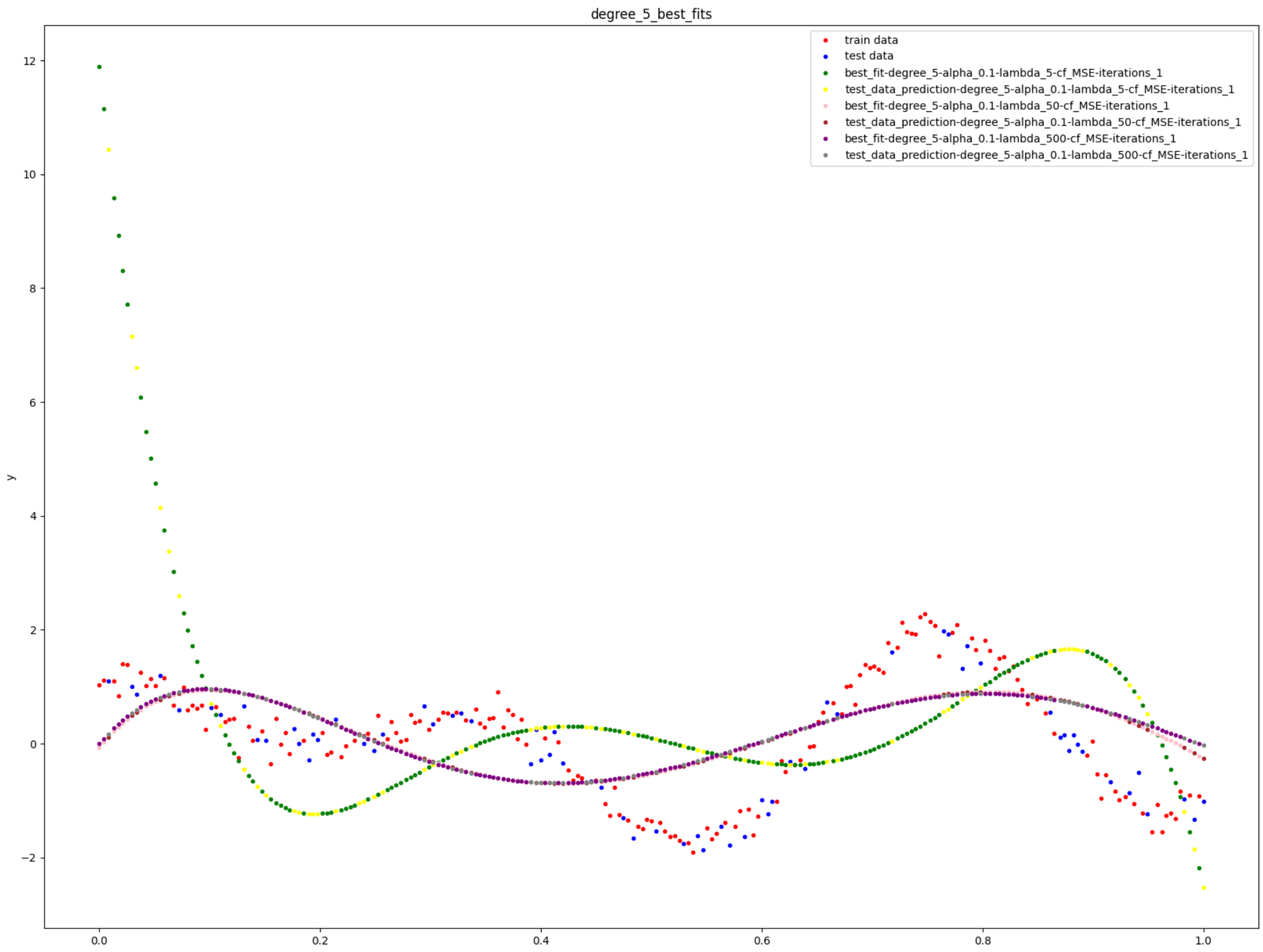
note: assume there is no **alpha** and **iterations** in the figure.

```
head of shuffled data:        x      y
0  0.751  2.286
1  0.167 -0.350
2  0.598 -1.603
3  0.404  0.246
4  0.462 -0.764
head of normalized shuffled data:           0
0  0.747465
1  0.155172
2  0.592292
3  0.395538
4  0.454361
train data = 167 rows, test data = 72 rows
model:algorithm: normal_equation- train_cost: 0.4085779521308575 test_cost: 0.37911434573243175 cf: MSE iterations: 1 alpha: 0.1 lambda: None degree: 3
theta: [  2.29275807 -20.05857585  43.96836352 -27.21535987]
model:algorithm: normal_equation- train_cost: 0.6392010263843899 test_cost: 0.6157226207736985 cf: RMSE iterations: 1 alpha: 0.1 lambda: None degree: 3
theta: [  2.29275807 -20.05857585  43.96836352 -27.21535987]
model:algorithm: normal_equation- train_cost: 0.2672354219730876 test_cost: 0.21471202810018208 cf: MSE iterations: 1 alpha: 0.1 lambda: None degree: 5
theta: [  0.7030809    5.51434315 -43.55611138  37.17897104  68.68570118
 -71.06355437]
model:algorithm: normal_equation- train_cost: 0.5169481811294896 test_cost: 0.46337029263881613 cf: RMSE iterations: 1 alpha: 0.1 lambda: None degree: 5
theta: [  0.7030809    5.51434315 -43.55611138  37.17897104  68.68570118
 -71.06355437]
model:algorithm: normal_equation- train_cost: 0.056763653843308046 test_cost: 0.056008977385087 cf: MSE iterations: 1 alpha: 0.1 lambda: None degree: 7
theta: [ 1.55481453e+00 -3.30939020e+00 -2.77151013e+02  2.74365988e+03
 -1.03175969e+04  1.82278117e+04 -1.51697453e+04  4.79468789e+03]
model:algorithm: normal_equation- train_cost: 0.2382512410110555 test_cost: 0.23666215875185243 cf: RMSE iterations: 1 alpha: 0.1 lambda: None degree: 7
theta: [ 1.55481453e+00 -3.30939020e+00 -2.77151013e+02  2.74365988e+03
 -1.03175969e+04  1.82278117e+04 -1.51697453e+04  4.79468789e+03]
```

Absolutely, the normal equation algorithm has performed far better than the gradient descent algorithm since it calculates the global minimum whereas in gradient descent we move forward to the local minimum (in linear regression, global minimum) step by step and it's possible that we never reach the minimum point because of the too large step size or too low step size, or too few iterations.

d- use lambda = 5, 50, and 500 and resolve the previous question (section c).

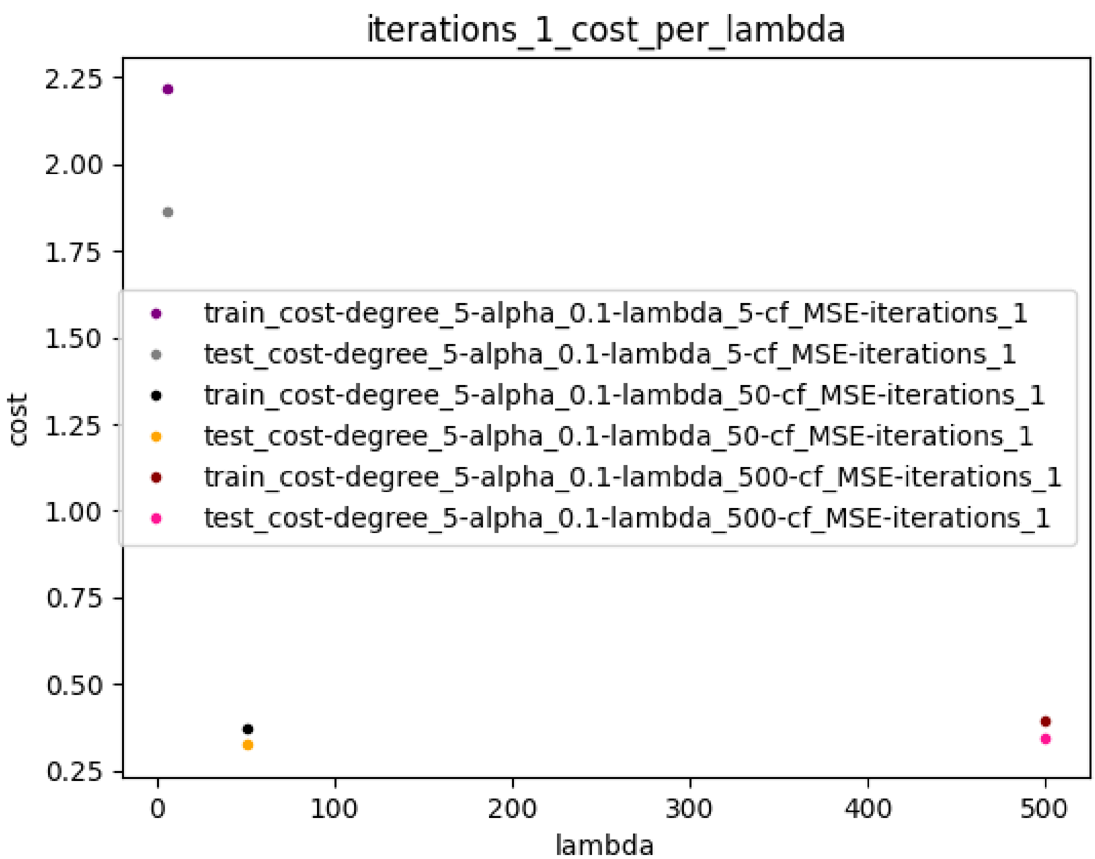
best fits:
file name: degree_5_best_fits.png
note: assume there is no **alpha** and **iterations** in the figure.



training and testing cost per lambda:

file name: iterations_1_cost_per_lambda.png

note: assume there is no **alpha** and **iterations** in the figure.



final results:

All models plus other figures are available in the folder q1-section_d-outputs

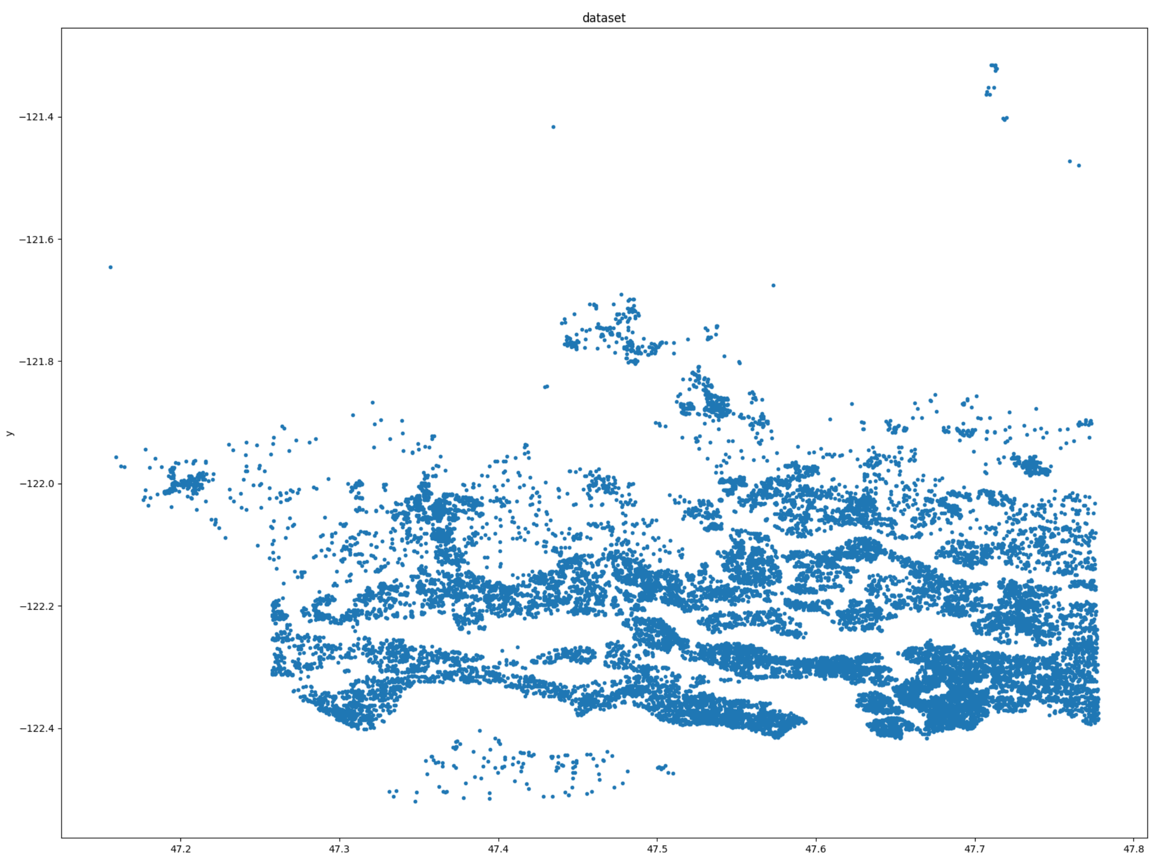
```
head of shuffled data:        x      y
0  0.921 -0.832
1  0.238  0.238
2  0.341  0.416
3  0.805  1.810
4  0.602 -1.279
head of normalized shuffled data:           0
0  0.919878
1  0.227181
2  0.331643
3  0.802231
4  0.596349
train data = 167 rows, test data = 72 rows
model:algorithm: normal_equation- train_cost: 2.2161302224522275 test_cost: 1.8612164023288504 cf: MSE iterations: 1 alpha: 0.1 lambda: 5 degree: 5
theta: [   11.89703496  -187.96332125   961.96513936 -2157.88481797
  2196.2646173   -826.79715914]
model:algorithm: normal_equation- train_cost: 0.3743149427843061 test_cost: 0.3247186457664754 cf: MSE iterations: 1 alpha: 0.1 lambda: 50 degree: 5
theta: [-6.48347218e-02  2.35294301e+01 -1.70856271e+02  4.00107153e+02
 -3.71832178e+02  1.18858879e+02]
model:algorithm: normal_equation- train_cost: 0.391742570374538 test_cost: 0.34178630826764117 cf: MSE iterations: 1 alpha: 0.1 lambda: 500 degree: 5
theta: [-5.46310707e-03  2.32252883e+01 -1.73206933e+02  4.14336083e+02
 -3.94948953e+02  1.30572031e+02]
```

note: assume there is no **alpha** and **iterations** in the figure.

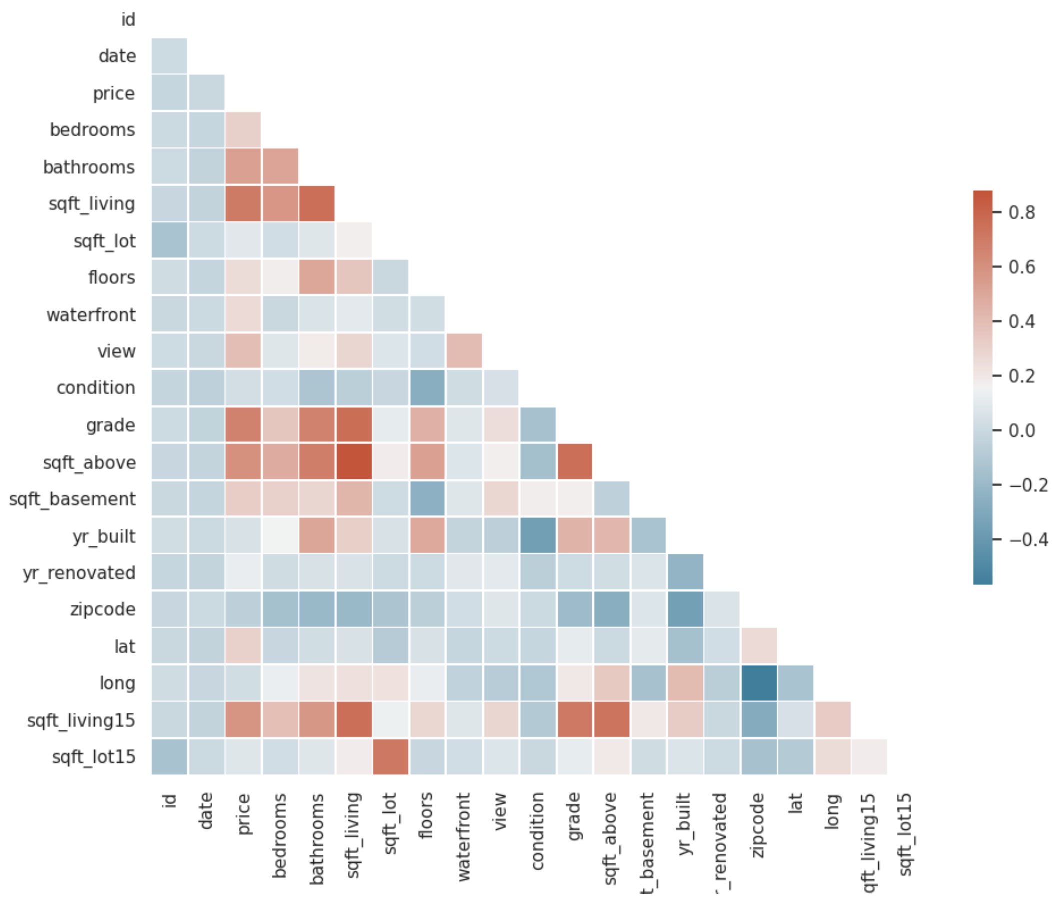
We can conclude that increasing the absolute value of lambda decreases the values of theta since it prevents the theta vector to get large values because the cost function returns a high cost with big elements in theta vector which is not desired.

2- problem: house price prediction

a- plot the data with lat and long attributes.
x = lat, y = long



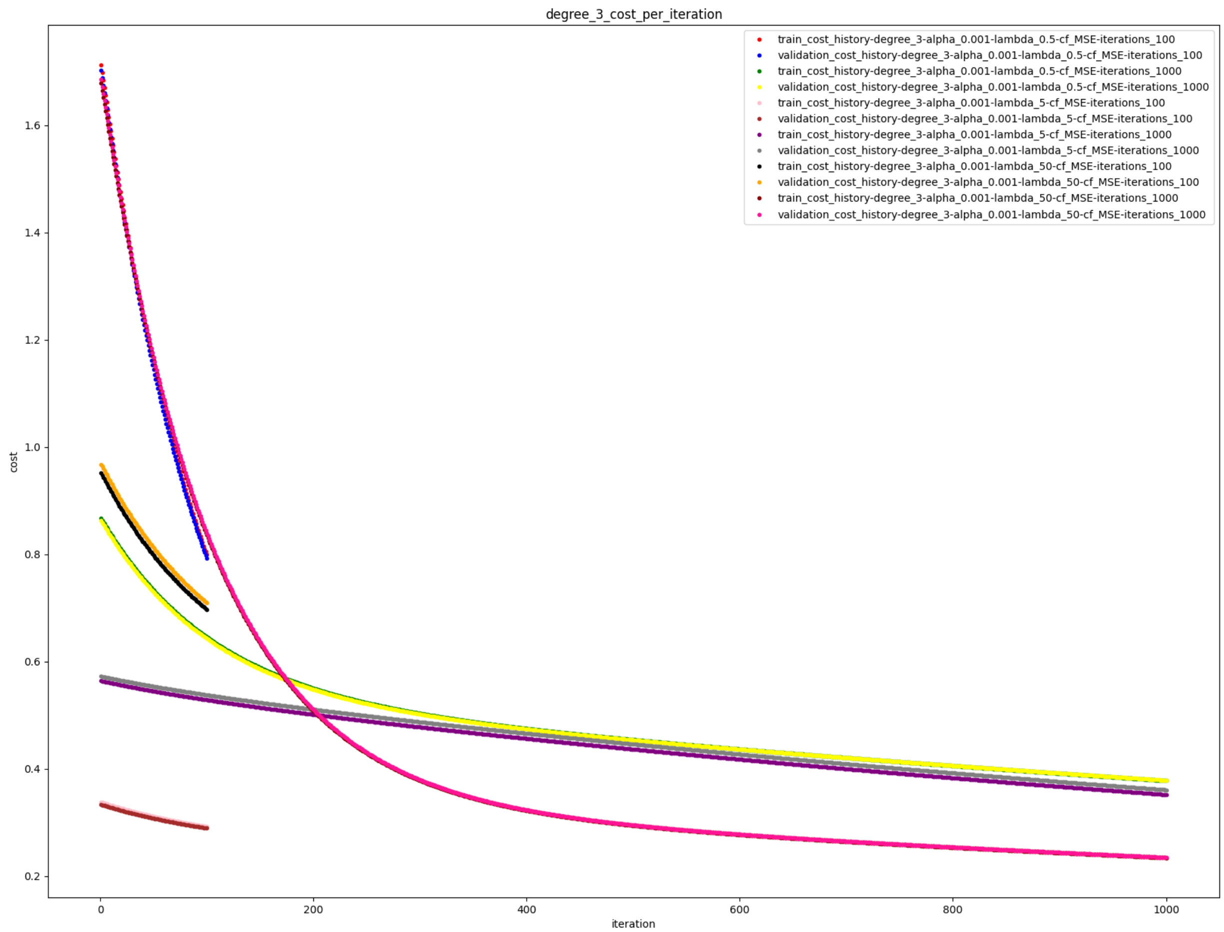
b- plot correlation hit map between dataset features.



c- is it possible to remove one or more of the features? is so, why?
yes, we can delete **<id>** and **<date>** columns since they don't provide any useful information and insight, in other words, their **gain ratio** is too low.
also, we can remove **<sqft_above>** or **<sqft_living>** because of the too high correlation, in other words, they provide the same information for the problem.

d- use gradient descent algorithm to fit a regression model. choose the best model as far as possible. once use all of the features and again remove selected features in section c then try again.
to solve this problem, 50% of data is used to train the models, 30% is used for validation and the rest is used for testing. the dataset is shuffled and normalized before training.
"id", "date", "zipcode", and "sqft_above" columns are removed in the second phase.

alpha = 0.001, degree = 3



train data = 10806 rows, test data = 6483 rows

model:algorithm: gd- train_cost: 0.7988139709294926 validation_cost: 0.7918911956363699 cf: MSE iterations: 100 alpha: 0.001 lambda: 0.5 degree: 3

model:algorithm: gd- train_cost: 0.37766068717820905 validation_cost: 0.37830270332160826 cf: MSE iterations: 1000 alpha: 0.001 lambda: 0.5 degree: 3

model:algorithm: gd- train_cost: 0.29217469371274457 validation_cost: 0.2892469839148515 cf: MSE iterations: 100 alpha: 0.001 lambda: 5 degree: 3

model:algorithm: gd- train_cost: 0.3516472889746067 validation_cost: 0.36053020040800193 cf: MSE iterations: 1000 alpha: 0.001 lambda: 5 degree: 3

model:algorithm: gd- train_cost: 0.6970221952832046 validation_cost: 0.7092553827426626 cf: MSE iterations: 100 alpha: 0.001 lambda: 50 degree: 3

model:algorithm: gd- train_cost: 0.23396173996006375 validation_cost: 0.2344914978529544 cf: MSE iterations: 1000 alpha: 0.001 lambda: 50 degree: 3

best model with alpha = 0.001, degree = 3:
model:algorithm: gd- train_cost: 0.23396173996006375 validation_cost: 0.2344914978529544 cf: MSE iterations: 1000 alpha: 0.001 lambda: 50 degree: 3

test cost on best model:
0.7863967153026591

alpha = 0.1, degree = 3

model:algorithm: gd- train_cost: 0.06095081335593591 validation_cost: 0.062117364375011724 cf: MSE iterations: 100 alpha: 0.1 lambda: 0.5 degree: 3

model:algorithm: gd- train_cost: 0.004684560895313374 validation_cost: 0.00467537572410383 cf: MSE iterations: 1000 alpha: 0.1 lambda: 0.5 degree: 3

model:algorithm: gd- train_cost: 0.03560847603012767 validation_cost: 0.03599898395854619 cf: MSE iterations: 100 alpha: 0.1 lambda: 5 degree: 3

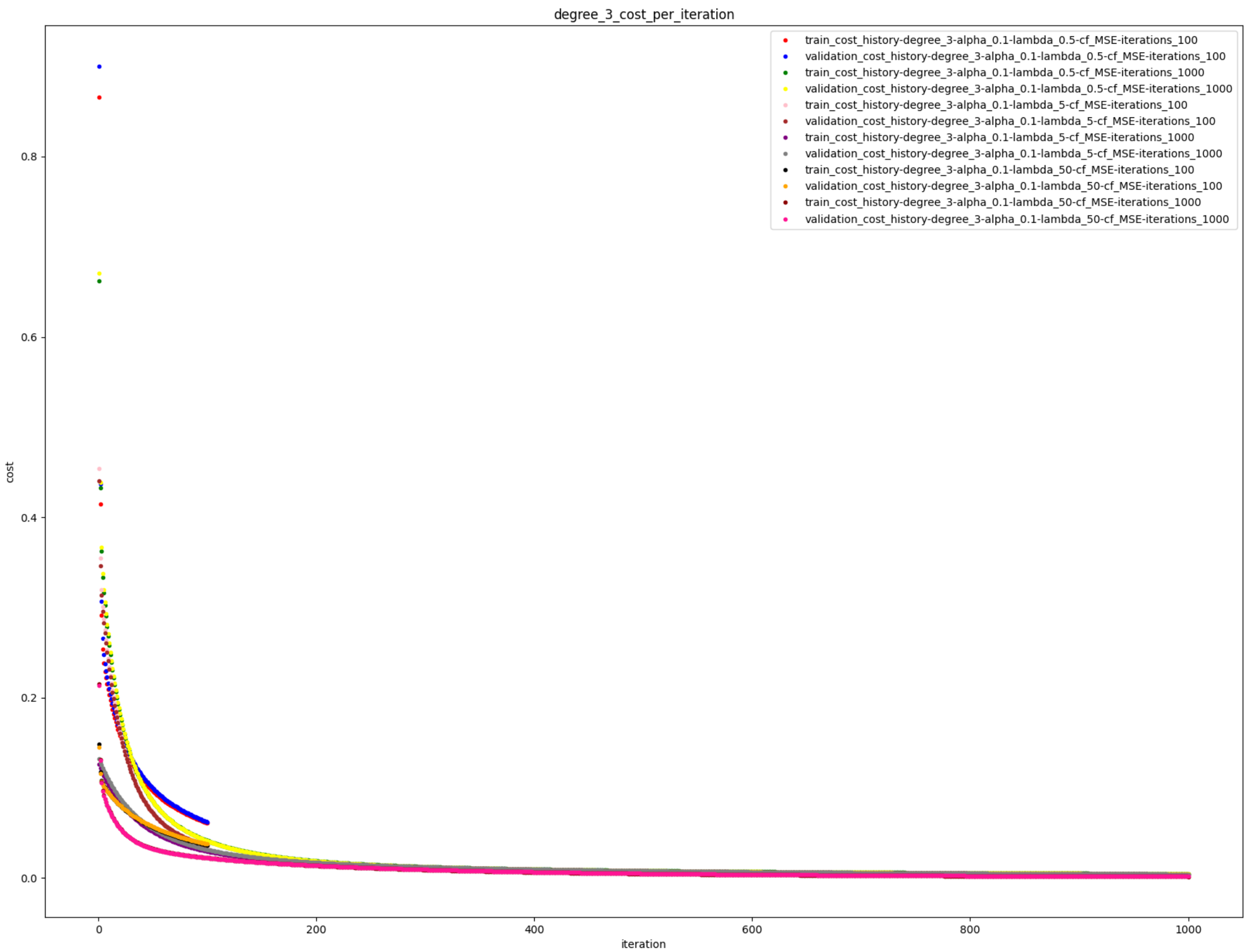
model:algorithm: gd- train_cost: 0.004027994004611019 validation_cost: 0.004105465695322559 cf: MSE iterations: 1000 alpha: 0.1 lambda: 5 degree: 3

model:algorithm: gd- train_cost: 0.036665490985332796 validation_cost: 0.037637718028954964 cf: MSE iterations: 100 alpha: 0.1 lambda: 50 degree: 3

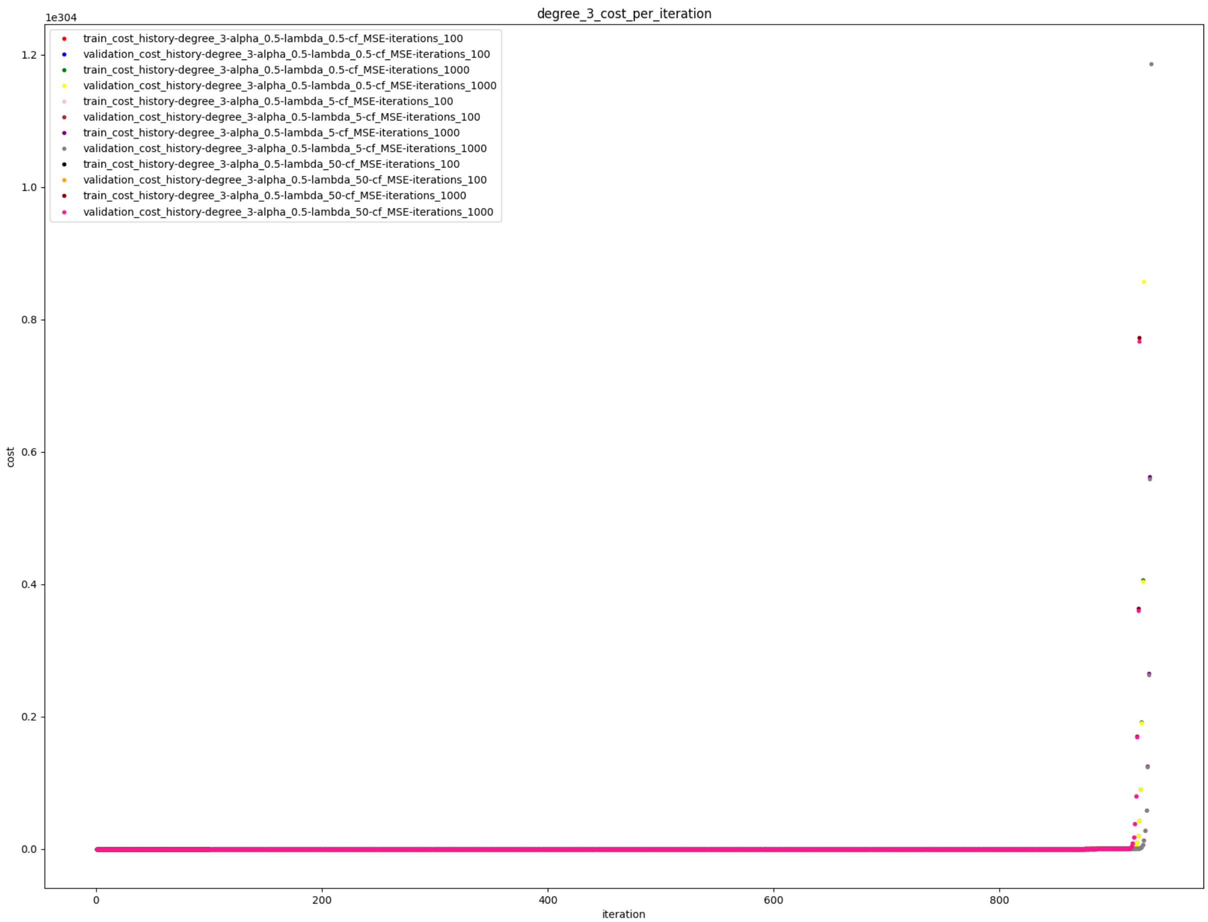
model:algorithm: gd- train_cost: 0.0016206050611930491 validation_cost: 0.0016903803945552893 cf: MSE iterations: 1000 alpha: 0.1 lambda: 50 degree: 3

best model with alpha = 0.1, degree = 3:
model:algorithm: gd- train_cost: 0.0016206050611930491 validation_cost: 0.0016903803945552893 cf: MSE iterations: 1000 alpha: 0.1 lambda: 50 degree: 3

test cost on best model:
0.0018371923705064145



alpha = 0.5, degree = 3



model:algorithm: gd- train_cost: 6.827099424427372e+31 validation_cost: 6.7832743397032205e+31 cf: MSE iterations: 100 alpha: 0.5 lambda: 0.5 degree: 3

model:algorithm: gd- train_cost: inf validation_cost: inf cf: MSE iterations: 1000 alpha: 0.5 lambda: 0.5 degree: 3

model:algorithm: gd- train_cost: 9.209280188501032e+32 validation_cost: 9.150162185191889e+32 cf: MSE iterations: 100 alpha: 0.5 lambda: 5 degree: 3

model:algorithm: gd- train_cost: inf validation_cost: inf cf: MSE iterations: 1000 alpha: 0.5 lambda: 5 degree: 3

model:algorithm: gd- train_cost: 1.07020384495143e+33 validation_cost: 1.0633325930243435e+33 cf: MSE iterations: 100 alpha: 0.5 lambda: 50 degree: 3

model:algorithm: gd- train_cost: inf validation_cost: inf cf: MSE iterations: 1000 alpha: 0.5 lambda: 50 degree: 3

best model with alpha = 0.5, degree = 3:
model:algorithm: gd- train_cost: 6.827099424427372e+31 validation_cost: 6.7832743397032205e+31 cf: MSE iterations: 100 alpha: 0.5 lambda: 0.5 degree: 3

test cost on best model:
6.809610365025575e+31

alpha = 0.001, degree = 2

model:algorithm: gd- train_cost: 0.7916973908978685 validation_cost: 0.7910479972846826 cf: MSE iterations: 100 alpha: 0.001 lambda: 0.5 degree: 2

model:algorithm: gd- train_cost: 0.11533543420865726 validation_cost: 0.1159030780419314 cf: MSE iterations: 1000 alpha: 0.001 lambda: 0.5 degree: 2

model:algorithm: gd- train_cost: 1.2269232037067852 validation_cost: 1.2114924788245145 cf: MSE iterations: 100 alpha: 0.001 lambda: 5 degree: 2

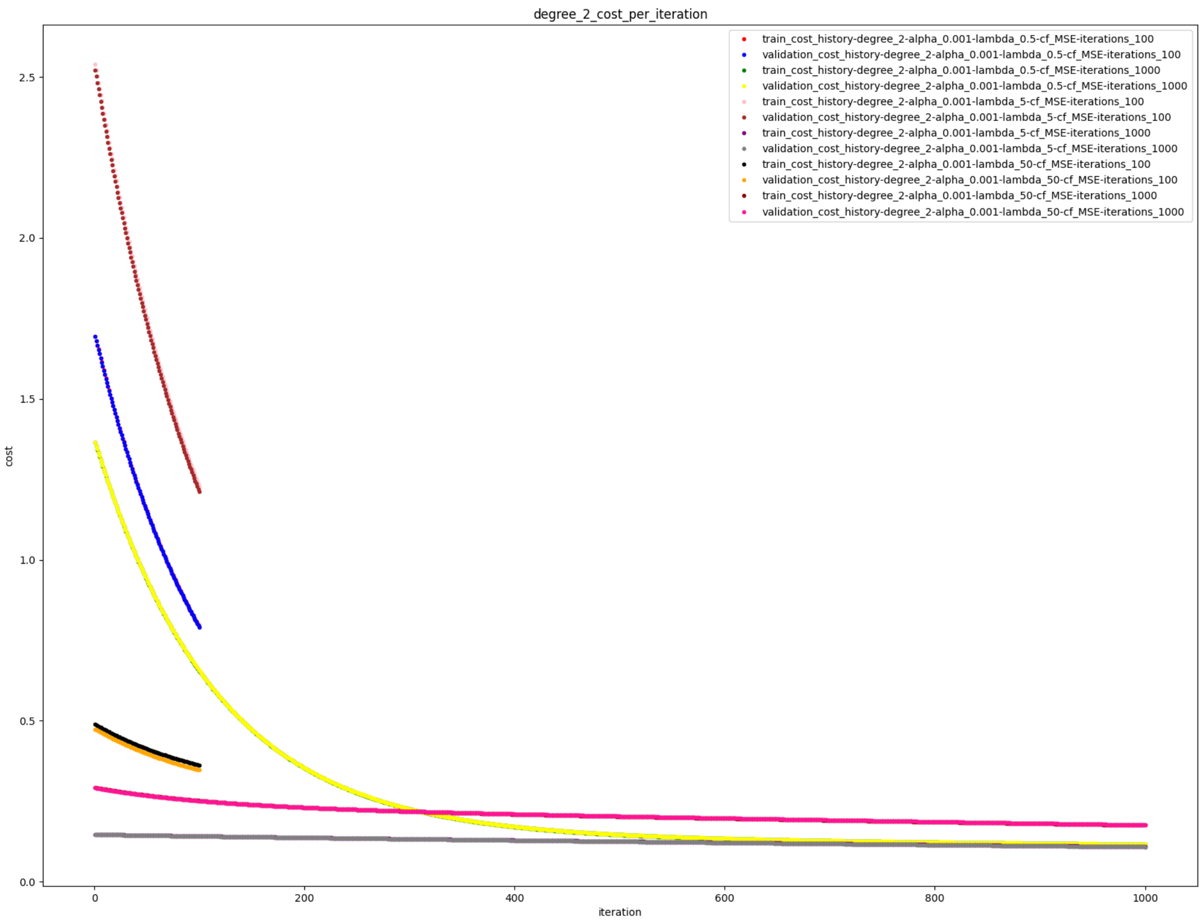
model:algorithm: gd- train_cost: 0.10835128463638545 validation_cost: 0.10800229491411835 cf: MSE iterations: 1000 alpha: 0.001 lambda: 5 degree: 2

model:algorithm: gd- train_cost: 0.36180105700777787 validation_cost: 0.3468368296982039 cf: MSE iterations: 100 alpha: 0.001 lambda: 50 degree: 2

model:algorithm: gd- train_cost: 0.17555705340715014 validation_cost: 0.17573936021805497 cf: MSE iterations: 1000 alpha: 0.001 lambda: 50 degree: 2

best model with alpha = 0.001, degree = 2:
model:algorithm: gd- train_cost: 0.10835128463638545 validation_cost: 0.10800229491411835 cf: MSE iterations: 1000 alpha: 0.001 lambda: 5 degree: 2

test cost on best model:
0.10757126258459974



alpha = 0.1, degree = 2

model:algorithm: gd- train_cost: 0.0230507417345265 validation_cost: 0.023242313006507205 cf: MSE iterations: 100 alpha: 0.1 lambda: 0.5 degree: 2

model:algorithm: gd- train_cost: 0.0026066893421623563 validation_cost: 0.002573714902075l905 cf: MSE iterations: 1000 alpha: 0.1 lambda: 0.5 degree: 2

model:algorithm: gd- train_cost: 0.03841893719979733 validation_cost: 0.03704448556074236 cf: MSE iterations: 100 alpha: 0.1 lambda: 5 degree: 2

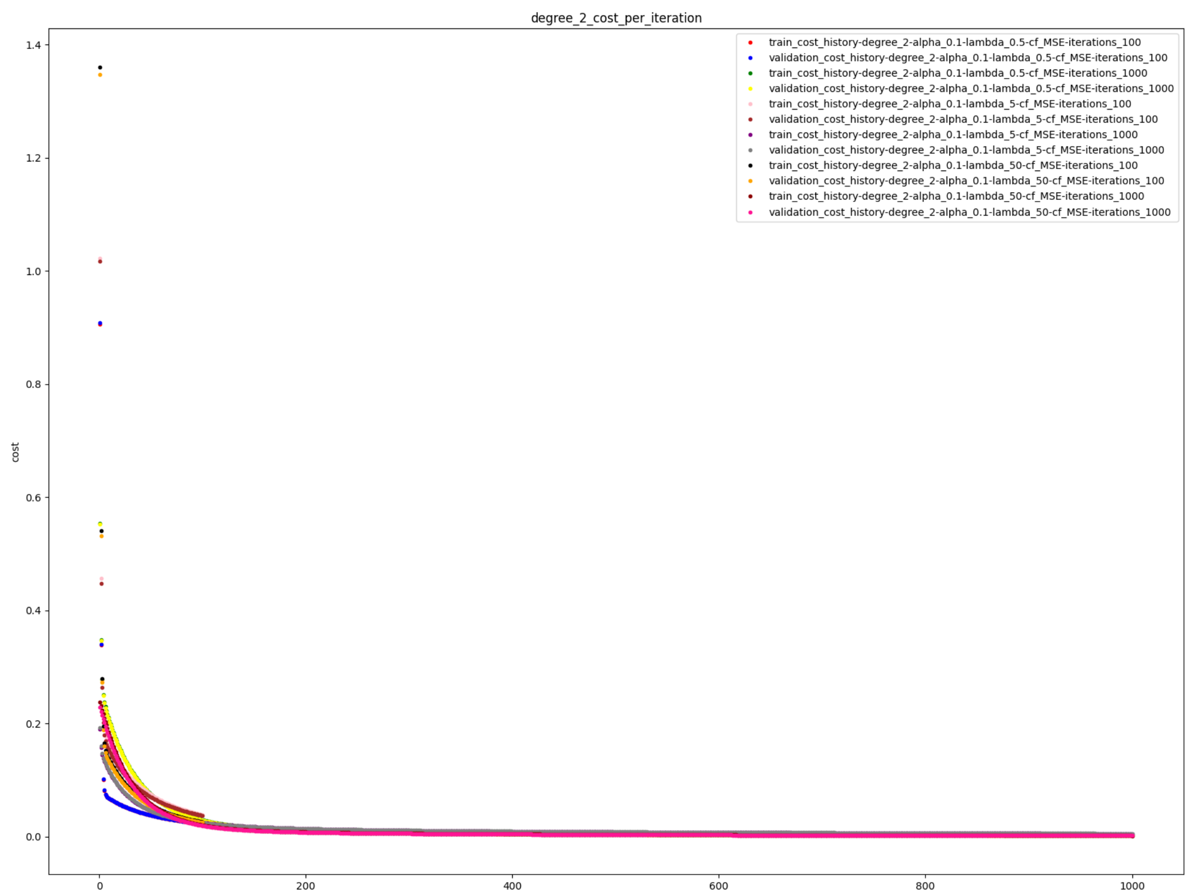
model:algorithm: gd- train_cost: 0.005083463347699982 validation_cost: 0.005306261259628261 cf: MSE iterations: 1000 alpha: 0.1 lambda: 5 degree: 2

model:algorithm: gd- train_cost: 0.026165784564623304 validation_cost: 0.025227154121630836 cf: MSE iterations: 100 alpha: 0.1 lambda: 50 degree: 2

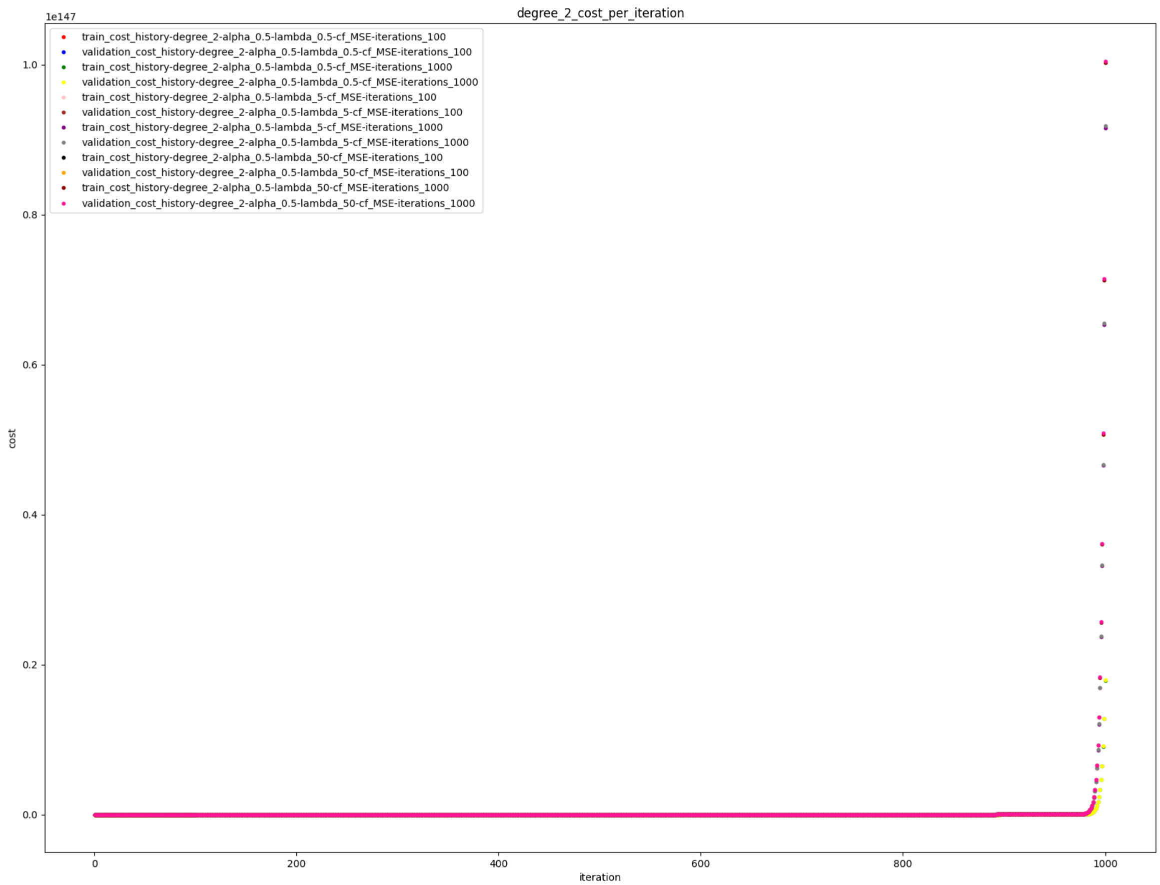
model:algorithm: gd- train_cost: 0.0015756286098117438 validation_cost: 0.001579416726282 cf: MSE iterations: 1000 alpha: 0.1 lambda: 50 degree: 2

best model with alpha = 0.1, degree = 2:
model:algorithm: gd- train_cost: 0.0015756286098117438 validation_cost: 0.001579416726282 cf: MSE iterations: 1000 alpha: 0.1 lambda: 50 degree: 2

test cost on best model:
0.0014896222045468472



alpha = 0.5, degree = 2



model:algorithm: gd- train_cost: 108175360650557.45 validation_cost: 108480788008486.36 cf: MSE iterations: 100 alpha: 0.5 lambda: 0.5 degree: 2

model:algorithm: gd- train_cost: 1.786589303575044e+146 validation_cost: 1.791633642531719e+146 cf: MSE iterations: 1000 alpha: 0.5 lambda: 0.5 degree: 2

model:algorithm: gd- train_cost: 449528994880931.5 validation_cost: 450798249689150.7 cf: MSE iterations: 100 alpha: 0.5 lambda: 5 degree: 2

model:algorithm: gd- train_cost: 9.157645538652459e+146 validation_cost: 9.183502347885044e+146 cf: MSE iterations: 1000 alpha: 0.5 lambda: 5 degree: 2

model:algorithm: gd- train_cost: 207977166870282.66 validation_cost: 208564551251089.56 cf: MSE iterations: 100 alpha: 0.5 lambda: 50 degree: 2

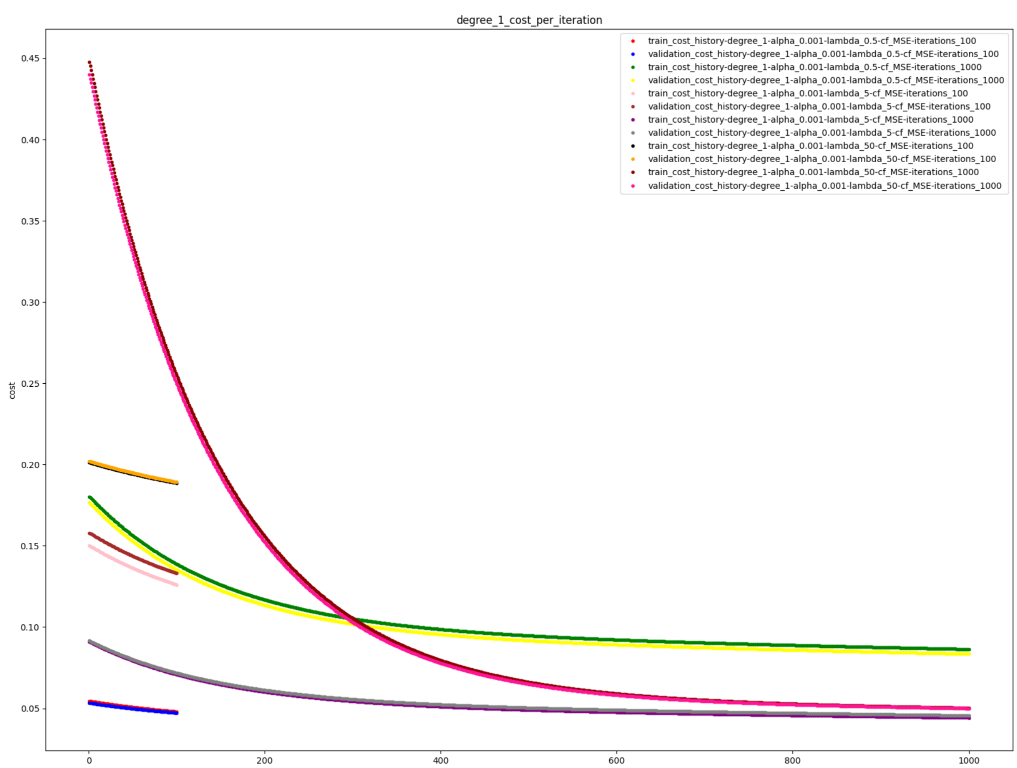
model:algorithm: gd- train_cost: 1.0022153425693e+147 validation_cost: 1.0050458726078732e+147 cf: MSE iterations: 1000 alpha: 0.5 lambda: 50 degree: 2

best model with alpha = 0.5, degree = 2:
model:algorithm: gd- train_cost: 108175360650557.45 validation_cost: 108480788008486.36 cf: MSE iterations: 100 alpha: 0.5 lambda: 0.5 degree: 2

test cost on best model:
108637630449210.2

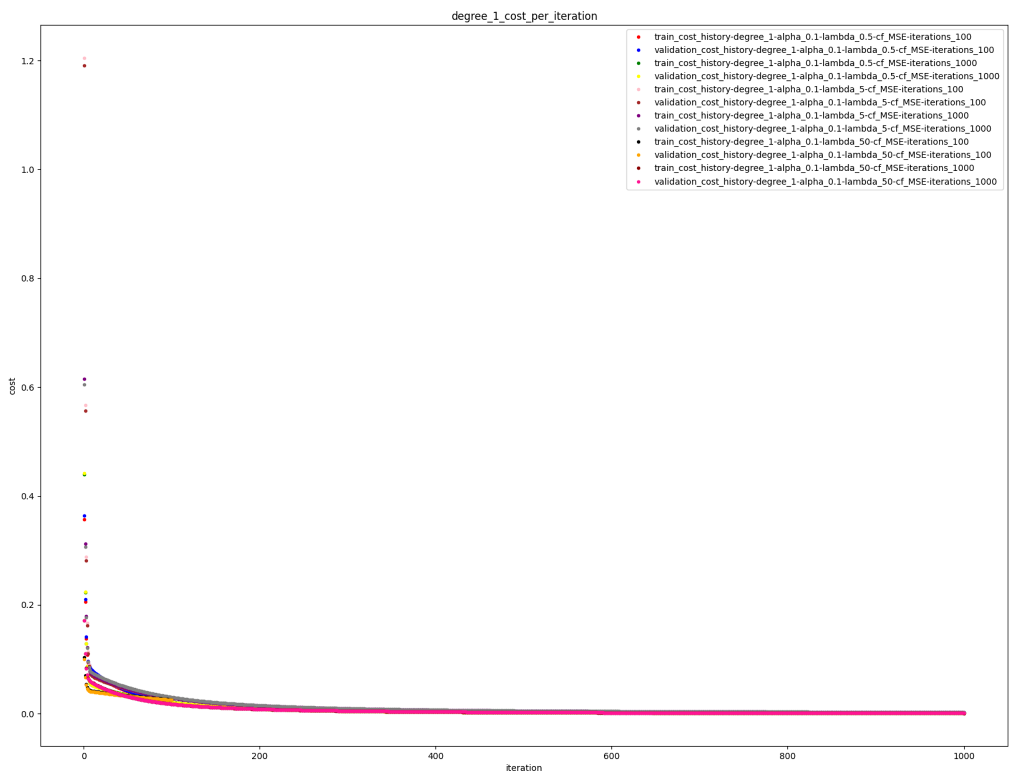
the rest of the results are summarized:

alpha = 0.001, degree = 1



best model:
model:algorithm: gd- train_cost: 0.04424991968911361 validation_cost: 0.0455508635709605 cf: MSE iterations: 1000 alpha: 0.001 lambda: 5 degree: 1
test cost on best model:
0.045206349979736805

alpha = 0.1, degree = 1

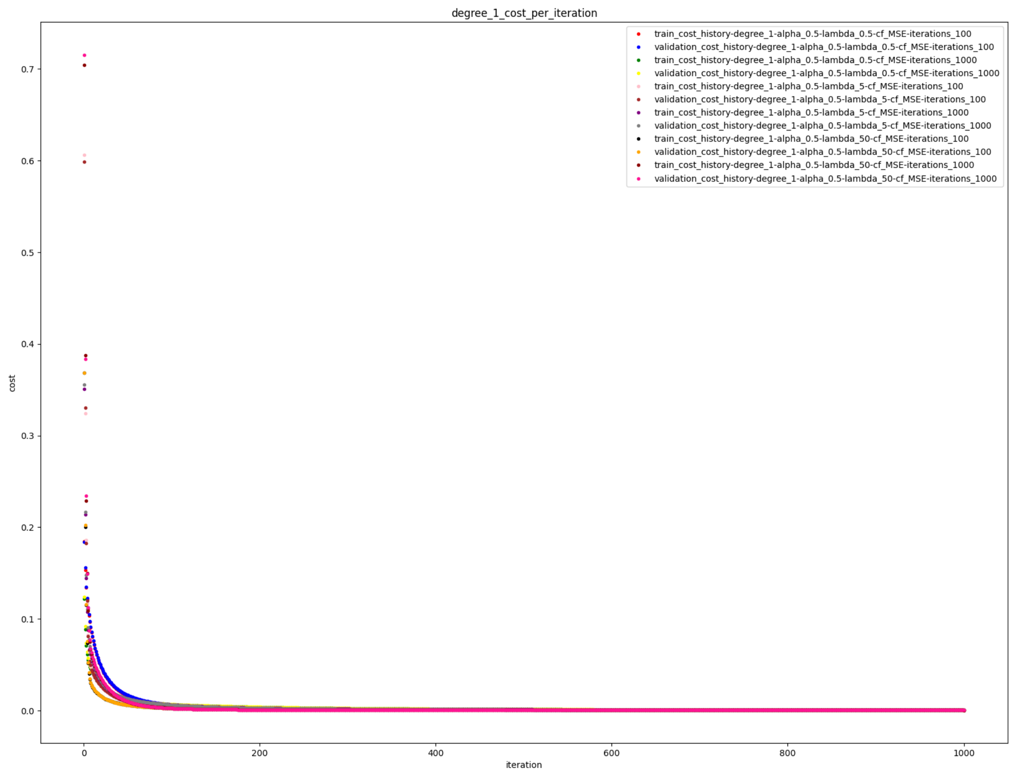


best model:
model:algorithm: gd- train_cost: 0.000959616604664554 validation_cost: 0.0009649598374774853 cf: MSE iterations: 1000 alpha: 0.1 lambda: 50 degree: 1
test cost on best model:
0.0010472252912346388

alpha = 0.5, degree = 1

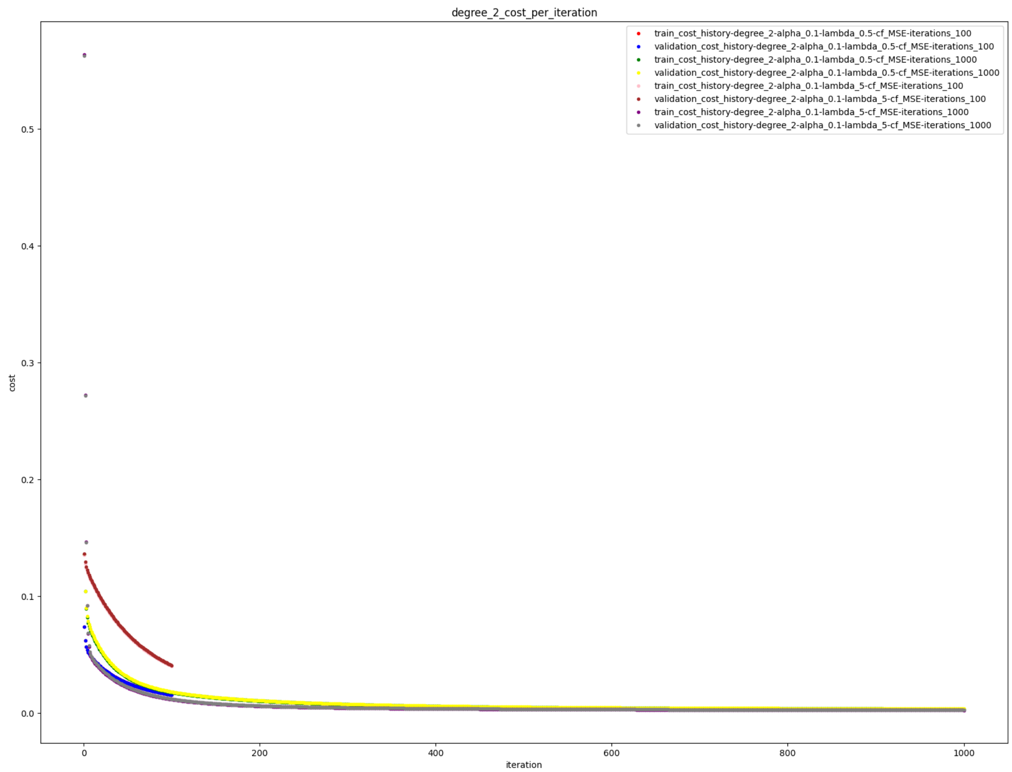
best model:
model:algorithm: gd- train_cost: 0.0003738714672594293 validation_cost: 0.0003760298354493929 cf: MSE iterations: 1000 alpha: 0.5 lambda: 50 degree: 1

test cost on best model:
0.00041495426735845524

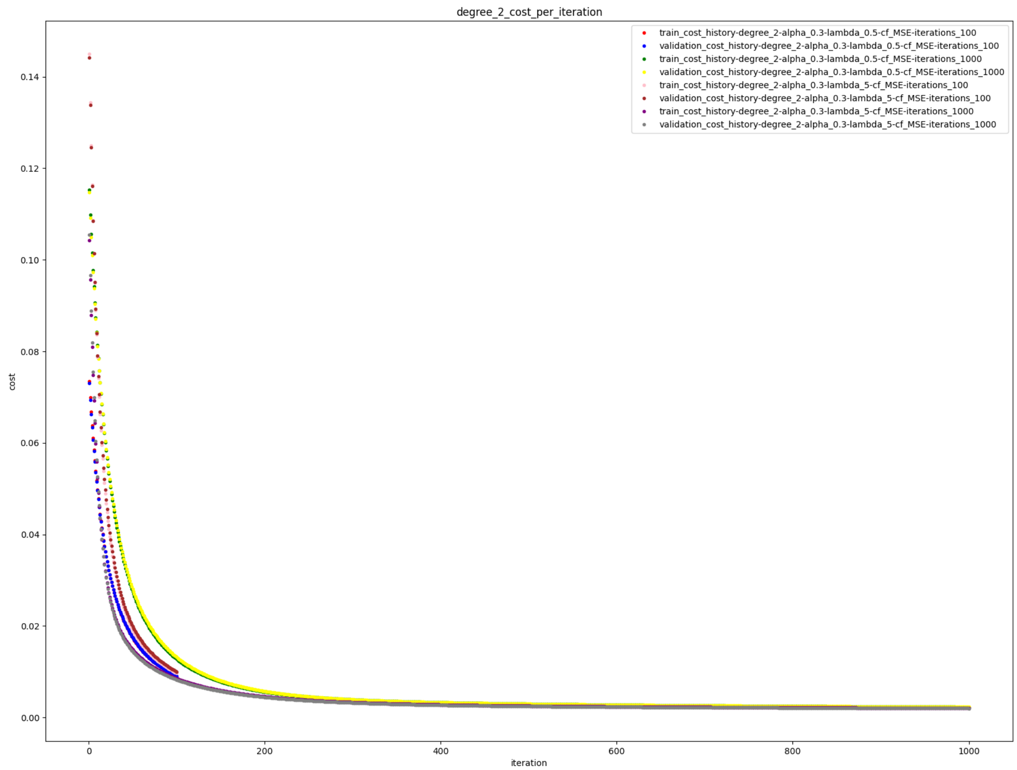


best model without removing any features:
alpha = 0.1, degree = 3:
model:algorithm: gd- train_cost: 0.0016206050611930491 validation_cost: 0.0016903803945552893 cf: MSE iterations: 1000 alpha: 0.1 lambda: 50 degree: 3

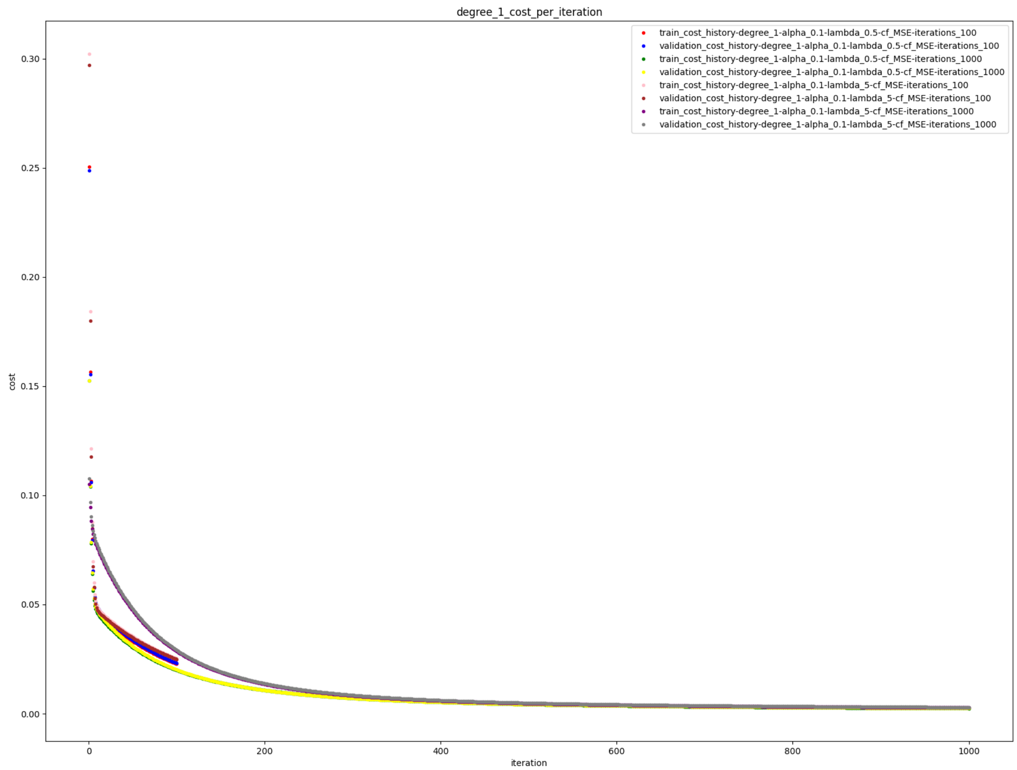
removing features:



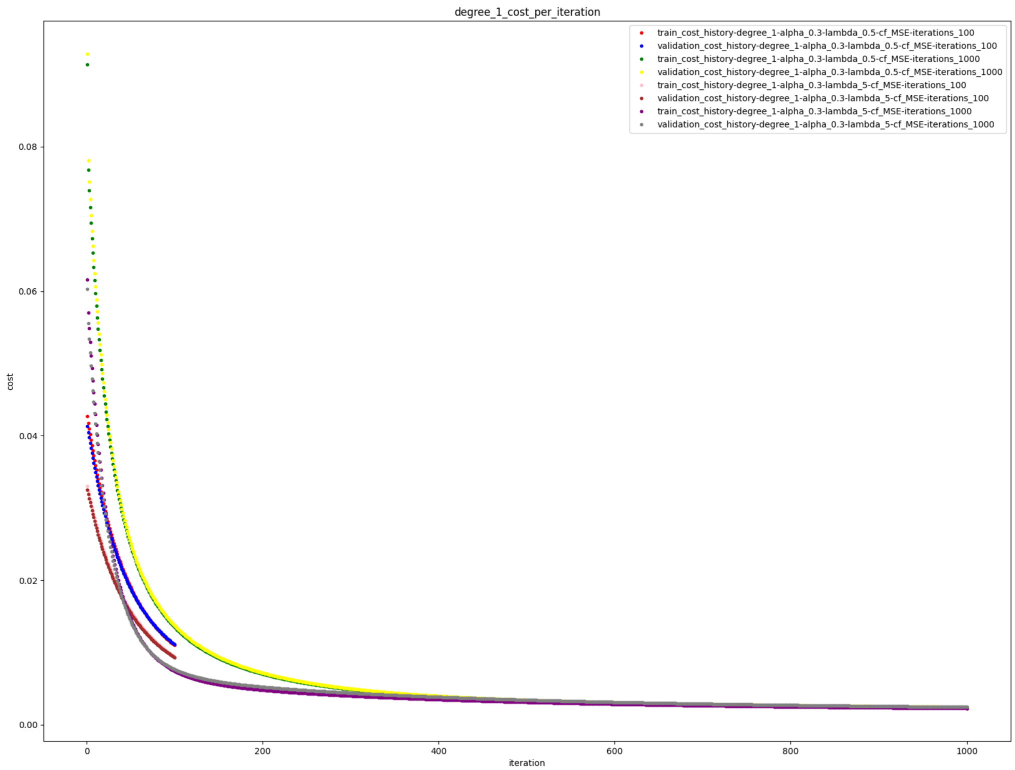
best model:
model:algorithm: gd- train_cost: 0.002664300188894608 validation_cost: 0.0027390606569459774 cf: MSE iterations: 1000 alpha: 0.1 lambda: 5 degree: 2
test cost on best model:
0.002837981302535363



best model:
model:algorithm: gd- train_cost: 0.0020571761715198838 validation_cost: 0.0020057087610222952 cf: MSE iterations: 1000 alpha: 0.3 lambda: 5 degree: 2
test cost on best model:
0.002012497380810018



best model:
model:algorithm: gd- train_cost: 0.002451082694009735 validation_cost: 0.002510112234240649 cf: MSE iterations: 1000 alpha: 0.1 lambda: 0.5 degree: 1
test cost on best model:
0.0024204517827956474



best model:
model:algorithm: gd- train_cost: 0.0023684684179785223 validation_cost: 0.0024095358784667694 cf: MSE iterations: 1000 alpha: 0.3 lambda: 0.5 degree: 1
test cost on best model:
0.002401802215155933

best model by removing features:
model:algorithm: gd- train_cost: 0.0020571761715198838 validation_cost: 0.0020057087610222952 cf: MSE iterations: 1000 alpha: 0.3 lambda: 5 degree: 2
test cost on best model:
0.002012497380810018

c- solve previous section with normal equation