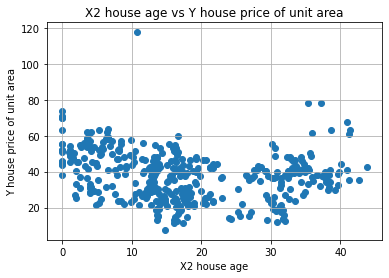
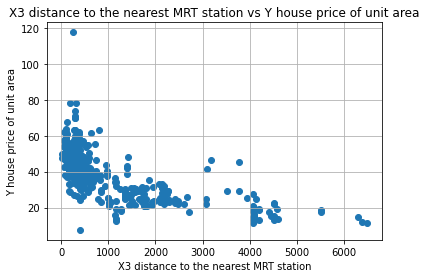
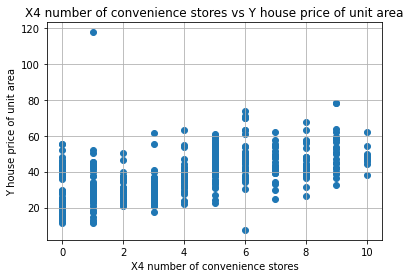
John Kenney

Part 1

Plots of the attributes of data frame x vs y output



Trial 1:

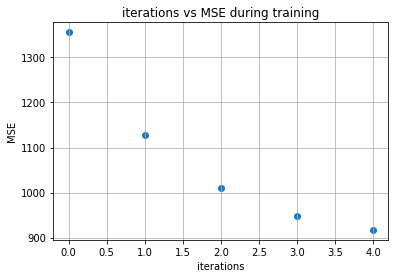
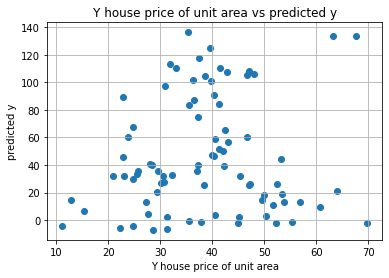
learning\_rate = .0000001

max\_iter = 5

train\_test\_ratio = .8

MSE on training set: 900.87588079

MSE on test set: 895.64000384

Trial 2:

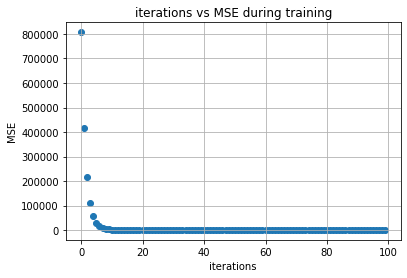
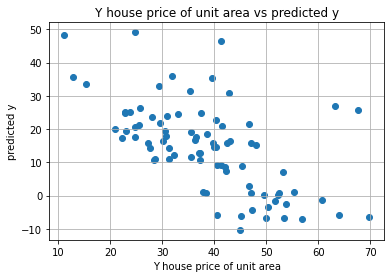
learning\_rate = .0000001

max\_iter = 100

train\_test\_ratio = .8

MSE on training set: 558.26824928

MSE on test set: 549.69713133

Trial 3:

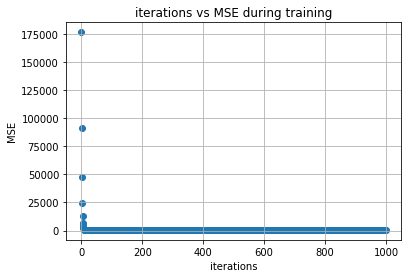
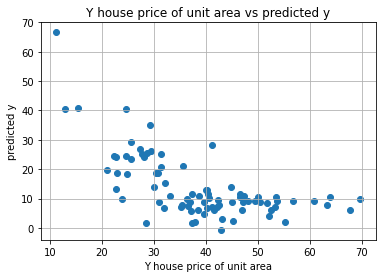
learning\_rate = .0000001

max\_iter = 1000

train\_test\_ratio = .8

MSE on training set: 534.49818537

MSE on test set: 522.08405416

Trial 4:

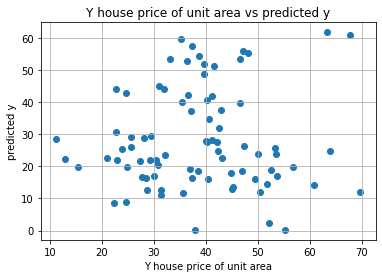
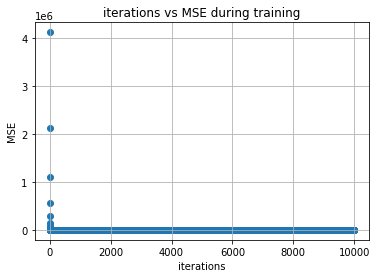
learning\_rate = .0000001

max\_iter = 10000

train\_test\_ratio = .8

MSE on training set: 235.46135048

MSE on test set: 234.63420241



Trial 5:

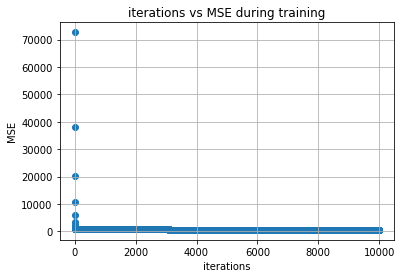
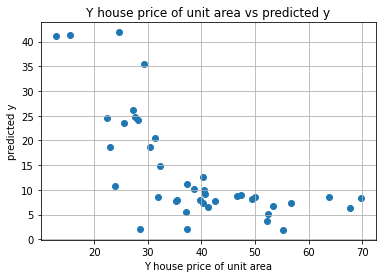
learning\_rate = .0000001

max\_iter = 10000

train\_test\_ratio = .9

MSE on training set: 546.51747774

MSE on test set: 552.15770206

Trial 6:

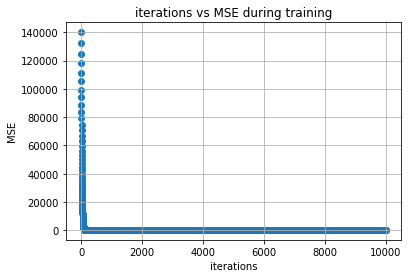
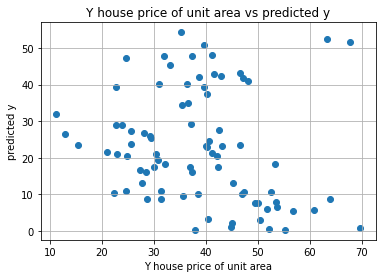
learning\_rate = .00000001

max\_iter = 10000

train\_test\_ratio = .8

MSE on training set: 351.63994805

MSE on test set: 343.87195278

Trial 7:

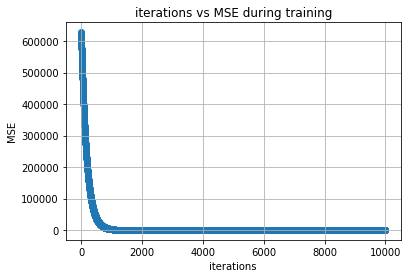
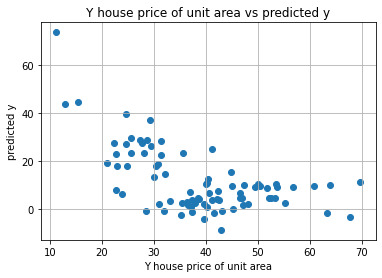
learning\_rate = .000000001

max\_iter = 10000

train\_test\_ratio = .8

MSE on training set: 633.60820208

MSE on test set: 620.36402755

Are you satisfied that you have found the best solution?

I found that trial 4 with a learning rate of .0000001, max iterations of 10000, and training rate of .8 had the lowest MSE of all the other trails. With a MSE of 234.63420241 on the testing data I am unsatisfied, but I think the best way to get a better MSE would be to collect more data because this data frame only has 414 rows.

Part 2:

Trial 1:

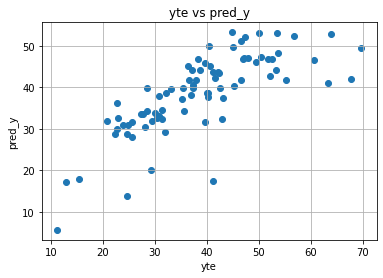
learningRate = 1e-3

max\_iterations = 10000

train\_test\_ratio = .8

R^2 = 0.5750436527049475

MSE = 62.73928858799078



Trial 2:

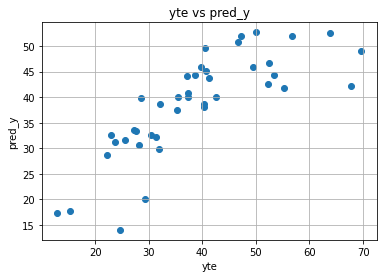
learningRate = 1e-3

max\_iterations = 1000

train\_test\_ratio = .9

R^2 = 0.6277107846212207

MSE = 64.9046803402397



Trial 3:

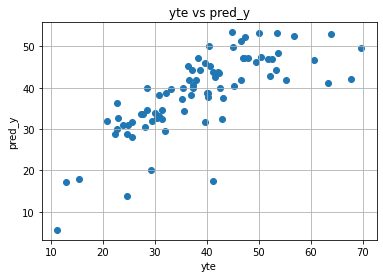
learningRate = 1e-3

max\_iterations = 1000

train\_test\_ratio = .8

R^2 = 0.5749724979174926

MSE = 62.74979366874311



Trial 4:

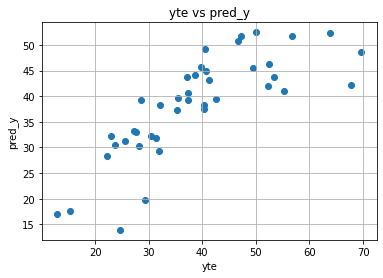
learningRate = 1e-4

max\_iterations = 1000

train\_test\_ratio = .9

R^2 = 0.6254485419739344

MSE = 65.2990783775956



Trial 5:

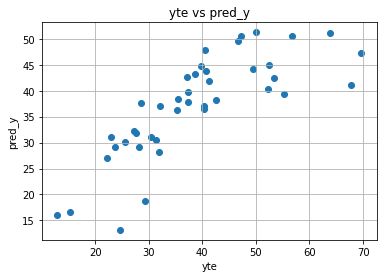
learningRate = 1e-5

max\_iterations = 1000

train\_test\_ratio = .9

R^2 = 0.6098180488307338

MSE = 68.02408925385096



Trial 6:

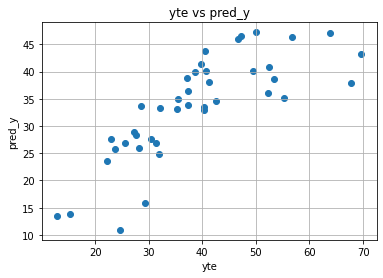
learningRate = 1e-6

max\_iterations = 10000

train\_test\_ratio = .9

R^2 = 0.4642948221810922

MSE = 93.39452201851066



Are you satisfied that the package has found the best solution. How can you check. Explain.

The best solution I found was trial 2 which had the highest R^2 value of .6277 but a little higher MSE than a few of the trails with an MSE of 64.90. This I believe is not the best solution because the R^2 and MSE are both not the best values for all the trails. I think I need more data to find a better fit for the data to get the y output.