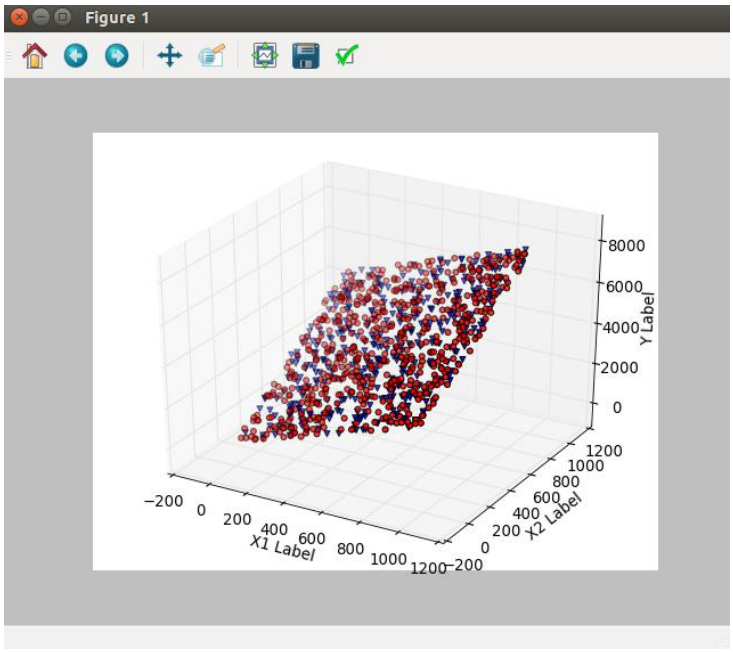
Report of project2 in MLT by Tongzhe Zhang 903181877

1. Best4linreg.py

(shown as following, red circle stands for training data and blue triangle stands for testing data)



Results:

LinRegLearner

In sample results

RMSE: 1.00205903582

corr: 0.99999982563

Out of sample results

RMSE: 0.969373489645

corr: 0.999999829386

KNN

In sample results

RMSE: 55.8619860155

corr: 0.999458130652

Out of sample results

RMSE: 83.568903898

corr: 0.998737538654

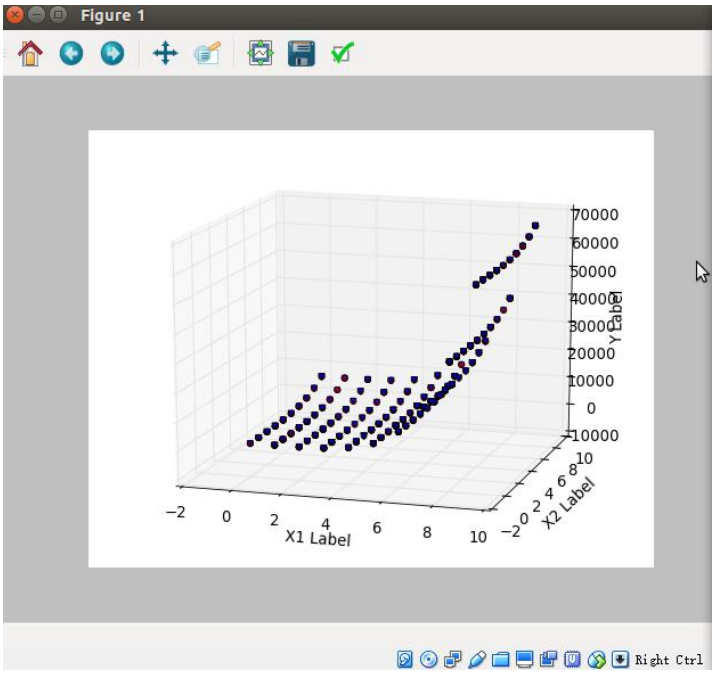
We can see that Linear Regression Learner performs much better than KNN in both train data and test data in the both aspect(RMSE and corr). For linear regression learner, both RMSE and corr are close in train set and test set.

Explain: I generate the random X1s and X2s between 1 and 1000, X1s and X2s both contains 1000 points. And then, get Y through the equation Y = X1 + X2 + noise(noise is a random 1000 points array according to N(0,1)) . The function we used are linear equation, it is more

suitable to the LinRegLearner.

2. Best4KNN.py

(shown as following, red circle stands for training data and blue triangle stands for testing data)



Result:

LinRegLearner

In sample results

RMSE: 10889.3732174

corr: 0.821448773341

Out of sample results

RMSE: 10608.0214323

corr: 0.790224033668

KNN

In sample results

RMSE: 262.818106873

corr: 0.9999052985

Out of sample results

RMSE: 417.300861605

corr: 0.999698955801

We can see that KNN learner performs much better than Linear Regression Learner in both train data and test data in the both aspect(RMSE and corr). For KNN learner, both corr are close in train set and test set.

Explain: I generate the random X1s and X2s between 1 and 10, X1s and X2s both contains 1000 points. And then, get Y through the equation Y = X1^5 + X2^4 + noise(noise is a random 1000 points array according to N(0,1)) . The function we used are non-linear equation, the

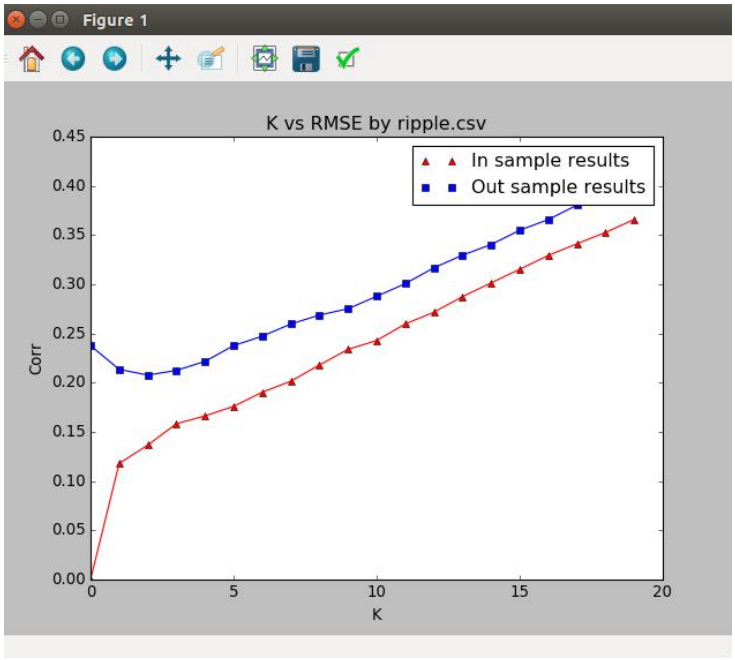
data is far from linearity. The data is not suitable to the LinRegLearner, KNN learner is better for this kind data.

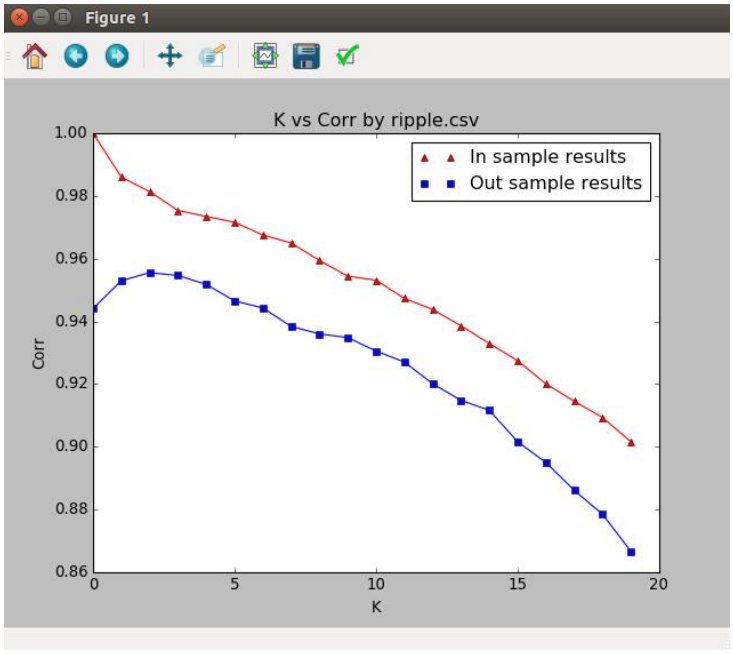
OVERFITTING

3. Consider the dataset ripple with KNN. For which values of K does overfitting occur? (Don't use bagging).

We use different K values to get results as following:

K	Rmse of in	Corr of in	Rmse of out	Corr of out
1	[[0.	1.	0.23771622	0.94411118]
2	[0.11786343	0.98597168	0.21354713	0.95295227]
3	[0.13659019	0.98136033	0.20776215	0.9555375]
4	[0.1583197	0.97538322	0.21248699	0.95460991]
5	[0.16624035	0.9734276	0.22162065	0.95177297]
6	[0.1759407	0.9716154	0.23768911	0.94650823]
7	[0.19037242	0.96752818	0.24723957	0.94426897]
8	[0.20149644	0.96498862	0.25962873	0.93838348]
9	[0.21800166	0.95941857	0.26872318	0.93599266]
10	[0.23391497	0.95440205	0.27512339	0.93482837]
11	[0.24285558	0.95312337	0.28779893	0.93053033]
12	[0.2596621	0.94732369	0.30066072	0.92702161]
13	[0.27163518	0.94378341	0.31668069	0.92009861]
14	[0.2874497	0.93856413	0.32960096	0.91478847]
15	[0.30141489	0.93294987	0.34036253	0.91163561]
16	[0.31520901	0.9274228	0.354872	0.90163719]
17	[0.32929112	0.92006809	0.36579765	0.89496521]
18	[0.34116793	0.9145111	0.3803897	0.88611278]
19	[0.35259918	0.90933376	0.39238625	0.87845479]
20	[0.36587153	0.90160947	0.40626173	0.86663569]]





Formal definition: hyperparameter choices that result in decreasing training error concurrent with increasing test error indicate overfitting.

So we can get results from above data and chart.

When k increases from 1 to 3, the training error decreases while the test error increases, this area is called overfitting.

We also can tell this result from the NO.2 picture, during the 1-3 of k, Corr of in sample is better than out sample and in different direction(one increase, another decrease).

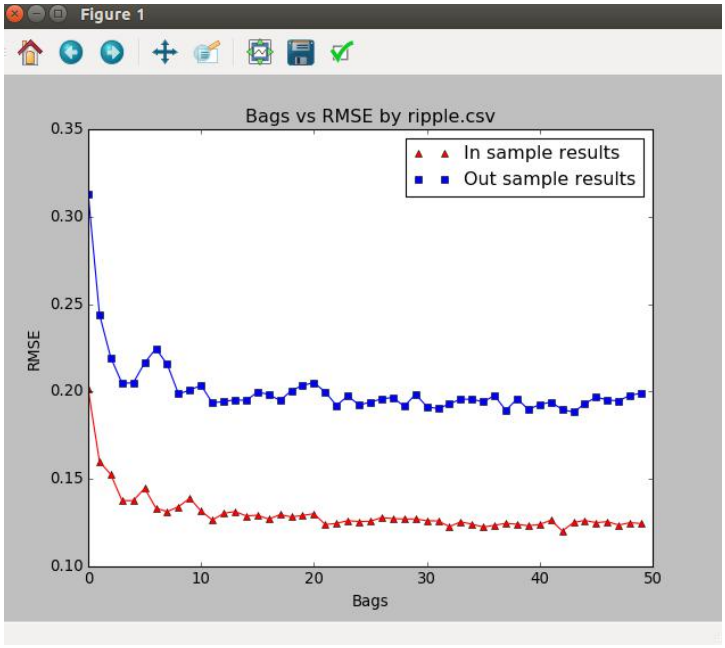
It make sense because when you predict the Y value according to nearest one points, it is overfitting.

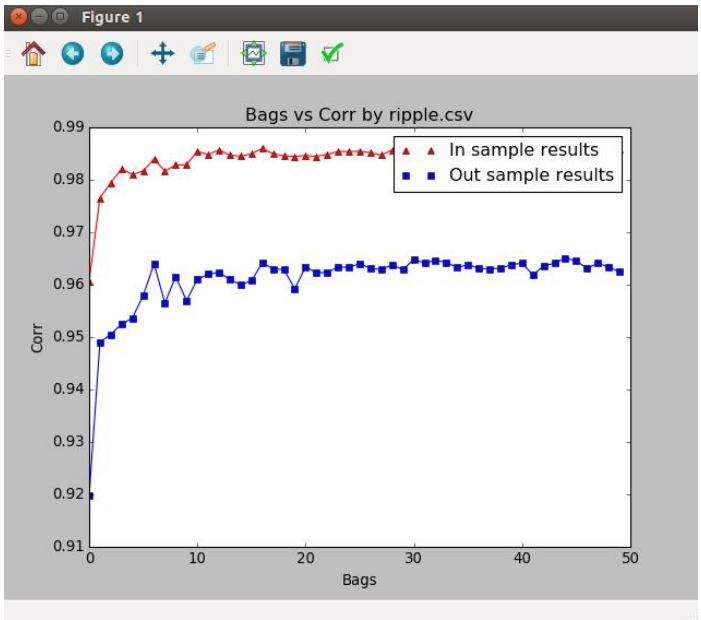
4. Now use bagging in conjunction with KNN with the `ripple` dataset. Choose some K keep it fixed. How does performance vary as you increase the number of bags? Does overfitting occur

with respect to the number of bags?

We fix the k equals to 3.

The result can be shown as following:





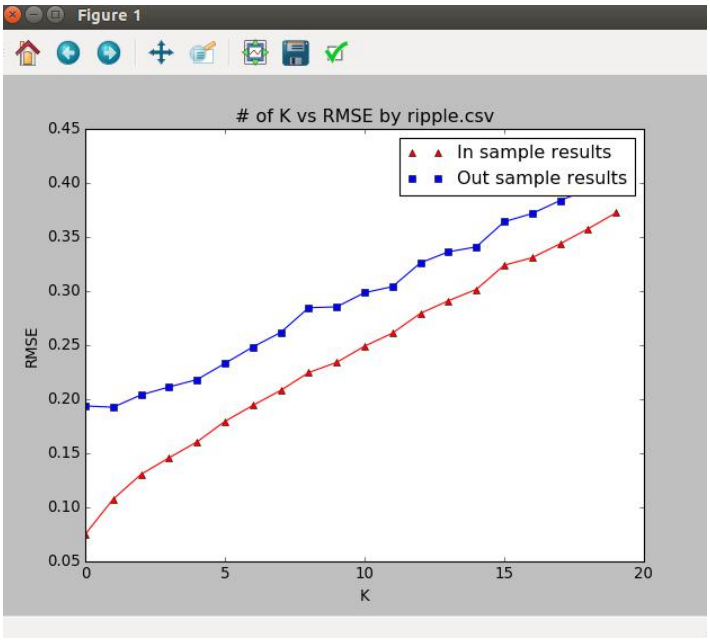
The larger the bags, the better performance the learner is. But after bags = 20, the both performance of training and testing are tend to stable, so we can just use bags = 20 to improve the KNNLearner.

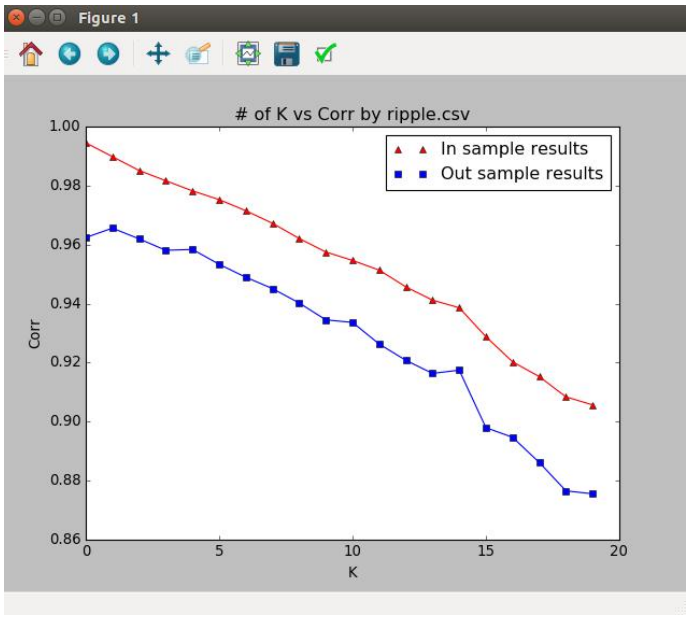
While the bags increases from 1 to 50, the RMSE and corr of training and testing data are both improving, it doesn't happen that they are obvious discordance, so we can regard that overfitting doesn't occur with respect to the number of bags.

5. Can bagging reduce or eliminate overfitting with respect to K for the ripple dataset? Fix the number of bags and vary K.

We fix bags equals to 20.

The result is as following:





Yes, bagging can reduce or eliminate overfitting with respect to K for the ripple dataset. We can tell this from above chart. But the performance will down after k = 2 or 3.