Problem1.

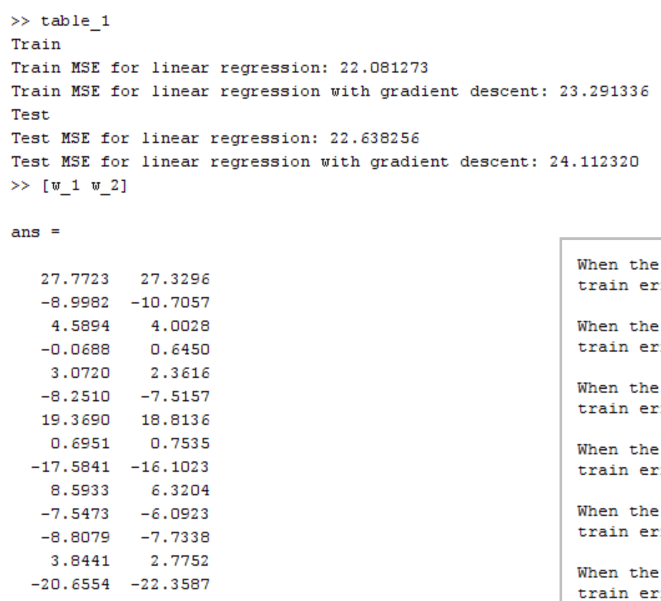
	Dec / Reg Tree	Linear Regression	Linear R – GD	Logistic R - GD
Housing MSE	train: 6.494752 test: 22.125965	train: 22.081273 test: 23.291336	train: 22.638256 test: 24.112320	meaningless
SpamBase ACC threshold = 0.4	train: 0.939321 test:0.893472	train: 0.9126 test: 0.8872	train: 0.9333 test: 0.9176	train: 0.9326 test: 0.9111

Using Logistic Regression for housing data base is meaningless, since housing data is an evaluation problem and price of house could be anything greater than 1, While logistic regression tries to map every data between zero and one.

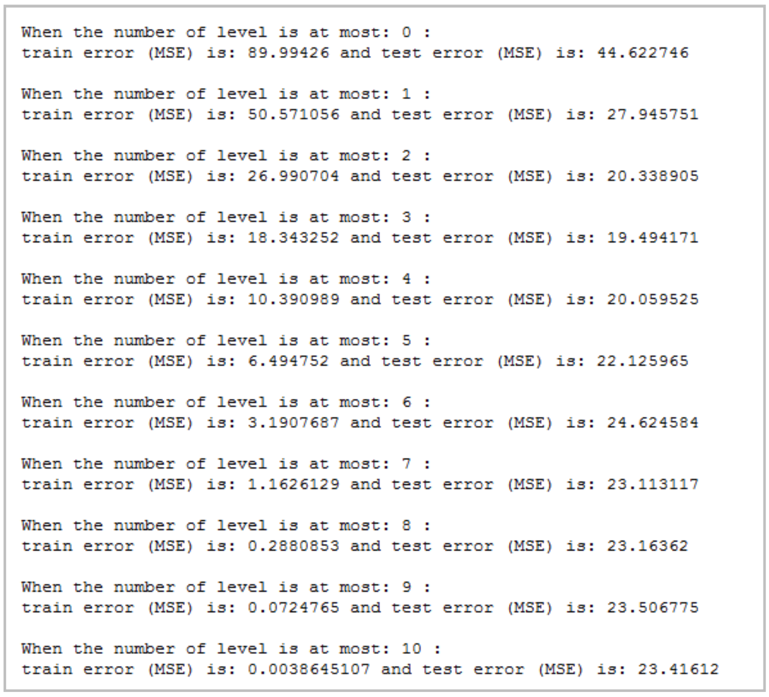
Housing Setting:

For linear regression with gradient descent, we run program in 5000 iterations. The initial learning rate is 0.05, which during running program we decrease the learning rate by half, after certain number of steps(after 1000th , 2000th and 4000th iteration).

In picture below, you can see the result for housing database. w_1 and w_2 are weights vector for linear regression (with normal equation) and linear regression with gradient descent.



```
>> table_1
Train
Train MSE for linear regression: 22.081273
Train MSE for linear regression with gradient descent: 23.291336
Test
Test MSE for linear regression: 22.638256
Test MSE for linear regression with gradient descent: 24.112320
>> [w_1 w_2]

ans =

   27.7723   27.3296
   -8.9982  -10.7057
    4.5894    4.0028
   -0.0688    0.6450
    3.0720    2.3616
   -8.2510   -7.5157
   19.3690   18.8136
    0.6951    0.7535
  -17.5841  -16.1023
    8.5933    6.3204
   -7.5473   -6.0923
   -8.8079   -7.7338
    3.8441    2.7752
  -20.6554  -22.3587
```



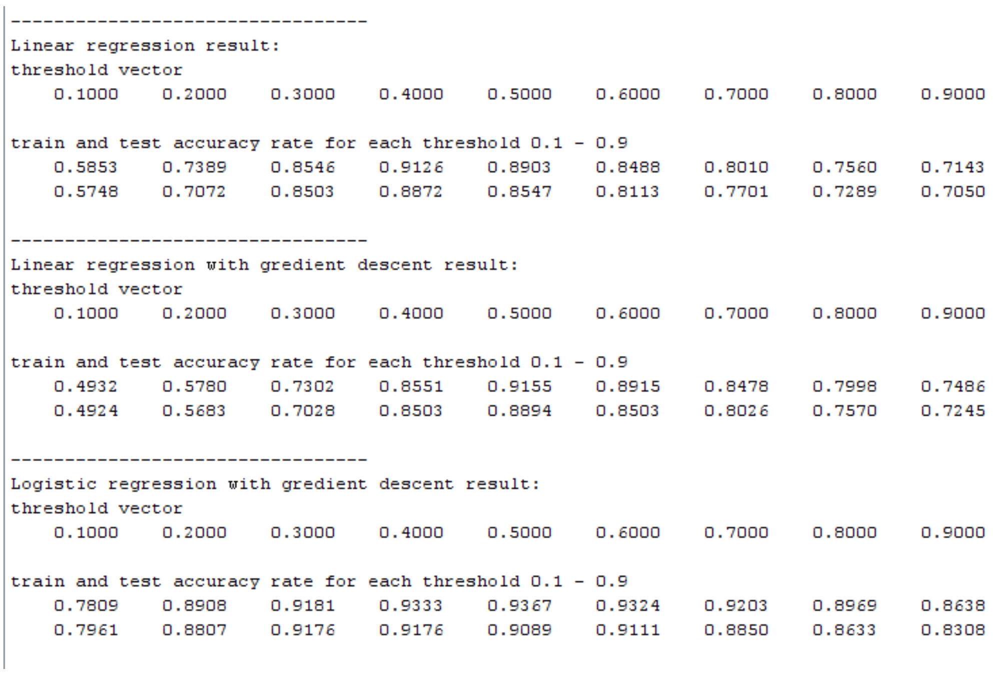
```
When the number of level is at most: 0 :
train error (MSE) is: 89.99426 and test error (MSE) is: 44.622746

When the number of level is at most: 1 :
train error (MSE) is: 50.571056 and test error (MSE) is: 27.945751

When the number of level is at most: 2 :
train error (MSE) is: 26.990704 and test error (MSE) is: 20.338905

When the number of level is at most: 3 :
train error (MSE) is: 18.343252 and test error (MSE) is: 19.494171

When the number of level is at most: 4 :
train error (MSE) is: 10.390989 and test error (MSE) is: 20.059525

When the number of level is at most: 5 :
train error (MSE) is: 6.494752 and test error (MSE) is: 22.125965

When the number of level is at most: 6 :
train error (MSE) is: 3.1907687 and test error (MSE) is: 24.624584

When the number of level is at most: 7 :
train error (MSE) is: 1.1626129 and test error (MSE) is: 23.113117

When the number of level is at most: 8 :
train error (MSE) is: 0.2880853 and test error (MSE) is: 23.16362

When the number of level is at most: 9 :
train error (MSE) is: 0.0724765 and test error (MSE) is: 23.506775

When the number of level is at most: 10 :
train error (MSE) is: 0.0038645107 and test error (MSE) is: 23.41612
```

The other picture, shows running of regression tree on housing database. Depending on the number of levels the tree has, train and test error might be different. In general, as the number of levels of tree increases, the train error decreases and test error increases. When the number of levels becomes great, we see over-fitting phenomena.

SpamBase setting:

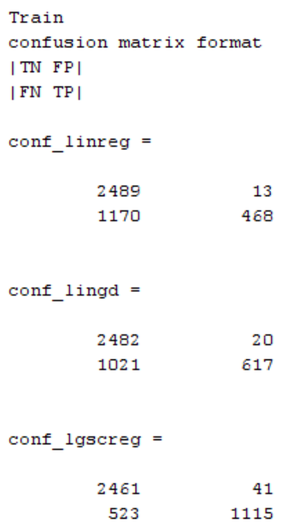
We choose 1/10 of data points randomly as a test data, rest of it is for train data.

For linear regression with gradient descent and logistic regression with gradient descent, we run program in 1000 iterations. The learning rate is 0.1. In picture below, we calculate the accuracy rate ($\frac{TN+TP}{N}$) for each of threshold 0.1-0.9. Thus, based on the table below, we choose threshold = 0.4 and 0.5 as the best threshold.

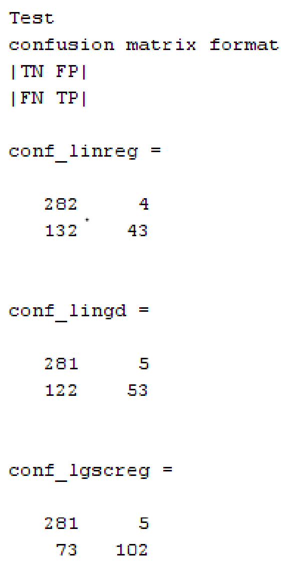


```
---------------------------------
Linear regression result:
threshold vector
    0.1000    0.2000    0.3000    0.4000    0.5000    0.6000    0.7000    0.8000    0.9000

train and test accuracy rate for each threshold 0.1 - 0.9
    0.5853    0.7389    0.8546    0.9126    0.8903    0.8488    0.8010    0.7560    0.7143
    0.5748    0.7072    0.8503    0.8872    0.8547    0.8113    0.7701    0.7289    0.7050

---------------------------------
Linear regression with gredient descent result:
threshold vector
    0.1000    0.2000    0.3000    0.4000    0.5000    0.6000    0.7000    0.8000    0.9000

train and test accuracy rate for each threshold 0.1 - 0.9
    0.4932    0.5780    0.7302    0.8551    0.9155    0.8915    0.8478    0.7998    0.7486
    0.4924    0.5683    0.7028    0.8503    0.8894    0.8503    0.8026    0.7570    0.7245

---------------------------------
Logistic regression with gredient descent result:
threshold vector
    0.1000    0.2000    0.3000    0.4000    0.5000    0.6000    0.7000    0.8000    0.9000

train and test accuracy rate for each threshold 0.1 - 0.9
    0.7809    0.8908    0.9181    0.9333    0.9367    0.9324    0.9203    0.8969    0.8638
    0.7961    0.8807    0.9176    0.9176    0.9089    0.9111    0.8850    0.8633    0.8308
```

In following picture, you can also see the confusion matrix of each technique. Threshold is 0.4.



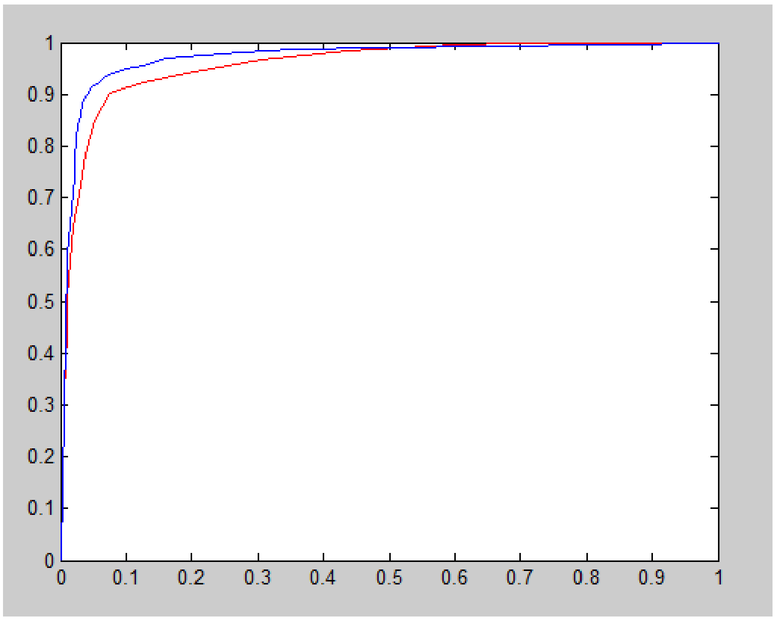
```
Train
confusion matrix format
|TN FP|
|FN TP|

conf_linreg =

        2489          13
        1170         468


conf_lingd =

        2482          20
        1021         617


conf_lgscreg =

        2461          41
         523        1115
```



```
Test
confusion matrix format
|TN FP|
|FN TP|

conf_linreg =

   282     4
   132    43


conf_lingd =

   281     5
   122    53


conf_lgscreg =

   281     5
    73   102
```

In chart below, you can see the ROC for linear regression with gradient descent (red one) and logistic regression with gradient descent (blue one). Area under red curve is 0.9583 and area under blue curve is 0.9715. Thus, blue curve is better which means logistic regression predictor is better.

```
auc_1 =

    0.9583


auc_2 =

    0.9715
```



Problem 4.

If the transfer function of the hidden units is linear, then we can simply reduce a three layer to a two layer, because linear function over a set of linear function is also equivalent to and can be written as one linear function. Since XOR problem, based on book, is a non-linearly separable function, thus we cannot implement it by a three layer network with linear functions.

Problem 5. Advice for applying machine learning.

Diagnostic for Debugging – improving algorithm

common approach: try more train sample, smaller features set, different lambda, … But common approach is consuming.

better approach:

- run diagnostics to find what the problem is.
- fix the problem.

Bias is high training error. Variance is much lower training error rather than test error(over-fitting)

Bias vs variance is one common diagnostic.

Optimization Algorithm Diagnostic:

1. Is algorithm converging?
2. Are you optimizing right function?

Error Analyses

Some applications, consider many learning components together.

Ablative analysis: Remove components from your system one at a time, to see how it breaks. How much does each of components really help?

Starting a problem

two approaches:

1. careful design – benefit: nice, more scalable, risk-premature optimization
2. build and fix – benefit: faster time to market

Problem 5. Paper: A few useful things to know about machine learning

1. Paper focuses on classification problem.
2. Learning = representation + evaluation + optimization
    - Classifier must be represented in a formal language. example (Decision tree, Neural network)
    - An evaluation function is needed to distinguish good classifiers from bad ones. example (Accuracy, Square error, Likelihood)
    - We need an objective function to search among the classifiers for the highest-scoring one. (Greedy search, Gradient Descent)

3. Generalization:
    - The goal of each machine learning algorithm is to generalize out of given train data.
    - Separation of test and train data is mandatory. We should always keep some data for test.
    - Although holding out data for test reduces the available amount of data for train, using cross validation can mitigate the problem.

4. The need for knowledge in learning should not be surprising. Machine learning is not magic; it can't get something from nothing. However, having a lot of data is not enough alone. Having a good representation is also important.
   Every learner must embody some knowledge or assumptions beyond the data it's given in order to generalize beyond it. No learner can beat random guessing over all possible functions to be learned.
   Some general assumptions like smoothness, similar examples having similar classes, limited dependences, or limited complexity—are often enough to do very well, and this is a large part of why machine learning has been so successful.

5. When the learner outputs a classifier which works accurate on train data, while only 50% accurate on test data. That is over-fitting. One way to understand over-fitting is by decomposing generalization error into *bias* and *variance.* Bias is a learner's tendency to consistently learn the same wrong thing. Variance is the tendency to learn random things irrespective of the real signal.

6. Intuition fails in high dimension.
   After over-fitting, the biggest problem in machine learning is the curse of dimensionality. Generalizing correctly becomes exponentially harder as the dimensionality (number of features) of the examples grows, because a fixed-size training set covers a dwindling fraction of the input space. More seriously, the similarity-based reasoning that machine learning algorithms depend on breaks down in high dimensions. Learners can implicitly take advantage of algorithms reducing the dimensionality.

7. Feature Engineering.
   If you have many independent features that each correlate well with the class, learning is easy. On the other hand, if the class is a very complex function of the features, you may not be able to learn it. Often, the raw data is not in a form that is amenable to learning, but you can construct features from it that are. This is typically where most of the effort in a machine learning project goes. It is often also one of the most interesting parts, where intuition, creativity and "black art" are as important as the technical stuff.