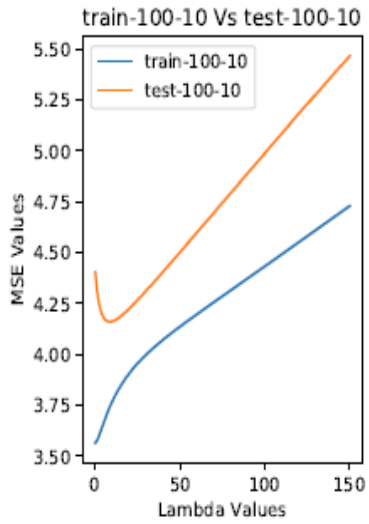
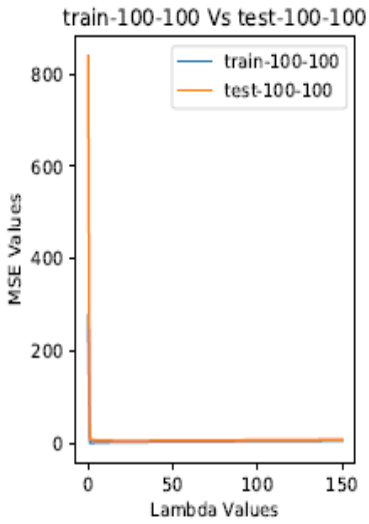
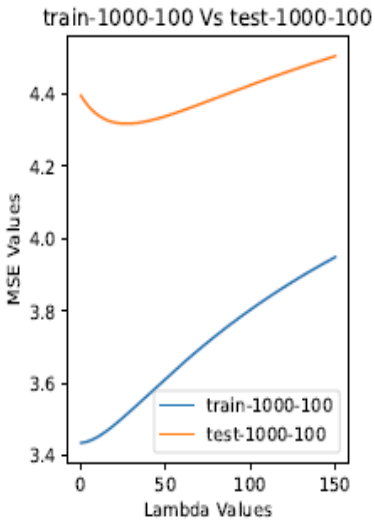
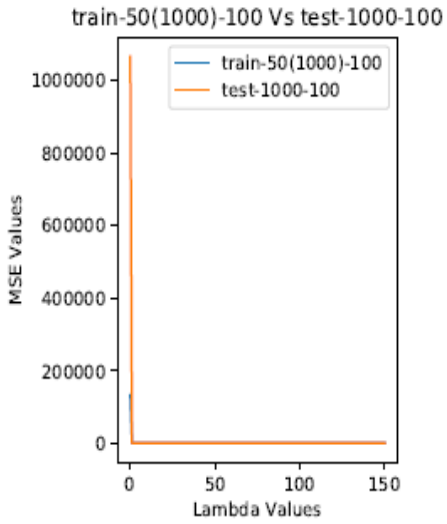
Question 1.

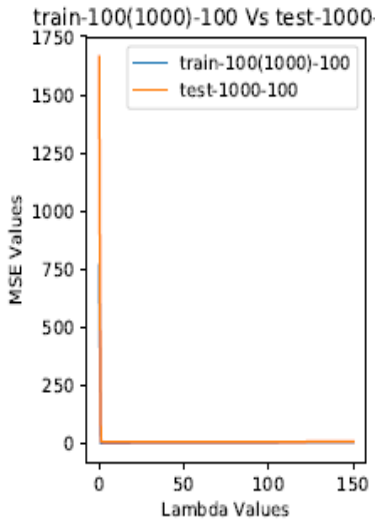
Following are the plots generated for MSE VS Lambda Values (ranging from 0 - 150) for all 6 training & test data set. (*The same can be generated by the code as well; mentioned in the readme file*)

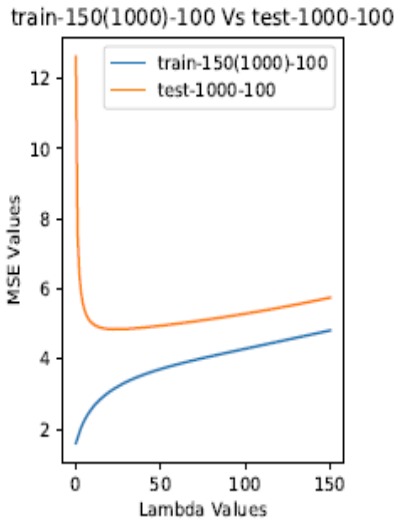












Question 1 A)

Below are the least MSE vales and corresponding lambda value for all datasets

Data Set	Least MSE (Test Dataset)	Corresponding Lambda (Test)	Least MSE (Train Dataset)	Corresponding Lambda (Train)
100-10	4.1597	9	3.5640	0
100-100	5.0728	22	0.4856	1
1000-100	4.3184	27	3.4349	0
50-(1000)-10	5.5123	8	0.3858	1
100-(1000)-10	5.1962	19	0.9167	1
150-(1000)-10	4.8437	24	1.5978	0

Following is the screenshot of the console output showing the Least MSE's and corresponding lambda values.

```
C:\Users\user\Desktop\Fall 2017 Course Material\Data Mining\Homework\Homework-1>python -W ignore homework1-Q1.py

******* L2 regularized linear regression For train-100-10.csv --- test-100-10.csv *******
For Training Data Set, Least MSE values is  3.56399319501  for lambda value =  0
For Test Data Set, Least MSE values is  4.15966392778  for lambda value =  9

******* L2 regularized linear regression For train-100-100.csv --- test-100-100.csv *******
For Training Data Set, Least MSE values is  0.485637033065  for lambda value =  1
For Test Data Set, Least MSE values is  5.07275045774  for lambda value =  22

******* L2 regularized linear regression For train-1000-100.csv --- test-1000-100.csv *******
For Training Data Set, Least MSE values is  3.4349195199  for lambda value =  0
For Test Data Set, Least MSE values is  4.31837045664  for lambda value =  27

******* L2 regularized linear regression For train-50(1000)-100.csv --- test-1000-100.csv *******
For Training Data Set, Least MSE values is  0.385822448947  for lambda value =  1
For Test Data Set, Least MSE values is  5.51227390988  for lambda value =  8

******* L2 regularized linear regression For train-100(1000)-100.csv --- test-1000-100.csv *******
For Training Data Set, Least MSE values is  0.916747892102  for lambda value =  1
For Test Data Set, Least MSE values is  5.1961997105  for lambda value =  19

******* L2 regularized linear regression For train-150(1000)-100.csv --- test-1000-100.csv *******
For Training Data Set, Least MSE values is  1.59775573414  for lambda value =  0
For Test Data Set, Least MSE values is  4.84372038141  for lambda value =  24


Plot generated successfully !!
```

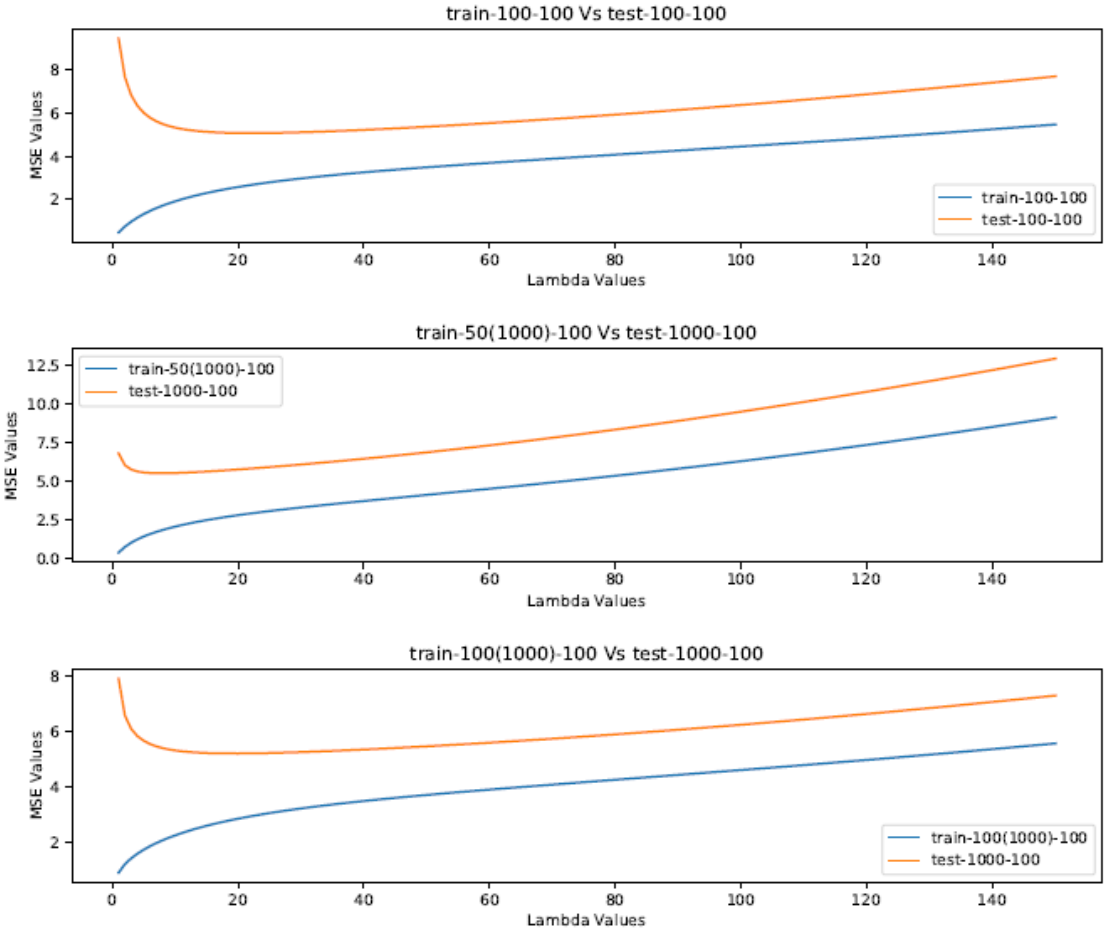
Question 1 B)

Following are the plots generated for MSE VS Lambda Values (ranging from 1 - 150) for 3 training & test data set along with the console output. (*The same can be generated by the code as well; mentioned in the readme file*)

Console Output :-

```
C:\Users\user\Desktop\Fall 2017 Course Material\Data Mining\Homework\Homework-1>python -W ignore homework1-Q1B.py

******* L2 regularized linear regression For train-100-100.csv --- test-100-100.csv *******
For Training Data Set, Least MSE values is  0.485637033065  for lambda value =  0
For Test Data Set, Least MSE values is  5.07275045774  for lambda value =  21

******* L2 regularized linear regression For train-50(1000)-100.csv --- test-1000-100.csv *******
For Training Data Set, Least MSE values is  0.385822448947  for lambda value =  0
For Test Data Set, Least MSE values is  5.51227390988  for lambda value =  7

******* L2 regularized linear regression For train-100(1000)-100.csv --- test-1000-100.csv *******
For Training Data Set, Least MSE values is  0.916747892102  for lambda value =  0
For Test Data Set, Least MSE values is  5.1961997105  for lambda value =  18


Plot generated successfully !!
```

Plots :-



Question 1 C)

As shown in the plot 1A & 1B, These 3 dataset shows very high MSE for lambda value 0, This is due to the **Over-fitting** of the model.

Over-fitting occurs when our model is excessively complex, such as having too many parameters relative to the number of observations. A model which has been over-fitted has poor prediction on the test data which is why we are seeing very high test set MSE values.

To avoid over-fitting, we use **Regularization technique** . Under regularization, we impose "Occam's razor" principle and add a penalty term for model complexity . Most commonly used techniques to minimize the over-fitting is L2 regularization (ridge regression) and L1 regularization (LASSO).

L2 Regularization :-

$E(w) = || Xw - y ||^2 + \lambda || w ||^2$
where $\lambda >= 0$ and $|| w ||^2 = W^T W$

According to the above formula, when lambda is 0, the penalty term becomes zero (i.e no regularization) which caused the over-fitting in the model.

Question 2.

Implemented the 10 Fold CV technique to find the best choice lambda value. Code has be attached with the submission and detailed instructions are given in the readme file to execute the code.

Question 2 A).

Following are the values for best choice lambda and the corresponding test set MSE for each test dataset.

Data Set	Least MSE (Avg of 10 folds CV)	Best Choice Lambda from CV	Corresponding test set MSE
100-10	4.1865	13	4.1735
100-100	4.4666	20	5.0768
1000-100	4.1396	39	4.3252
50-(1000)-10	5.2852	24	5.8789
100-(1000)-10	4.8522	31	5.2525
150-(1000)-10	4.8769	47	4.9290

Following is the screenshot of the console output showing the Cross Validation output and best lambda value and corresponding test set MSE.

```
C:\Users\user\Desktop\Fall 2017 Course Material\Data Mining\Homework\Homework-1>python -W ignore homework1-Q2.py

********** Running Cross Validation for Traing DataSet -  train-100-10.csv *************
From CV, Least MSE is  4.18654949545  for Lambda Value =  13
      Now Calculating MSE on the TestData using Best choice lambda (obtained From CV) =  13
      For best choice lambda 13 corresponding test set MSE is  4.17350735396



********** Running Cross Validation for Traing DataSet -  train-100-100.csv *************
From CV, Least MSE is  4.4665722192  for Lambda Value =  20
      Now Calculating MSE on the TestData using Best choice lambda (obtained From CV) =  20
      For best choice lambda 20 corresponding test set MSE is  5.07675140801



********** Running Cross Validation for Traing DataSet -  train-1000-100.csv *************
From CV, Least MSE is  4.13964107453  for Lambda Value =  39
      Now Calculating MSE on the TestData using Best choice lambda (obtained From CV) =  39
      For best choice lambda 39 corresponding test set MSE is  4.32518286252



********** Running Cross Validation for Traing DataSet -  train-50(1000)-100.csv *************
From CV, Least MSE is  5.28522135586  for Lambda Value =  24
      Now Calculating MSE on the TestData using Best choice lambda (obtained From CV) =  24
      For best choice lambda 24 corresponding test set MSE is  5.87891144633



********** Running Cross Validation for Traing DataSet -  train-100(1000)-100.csv *************
From CV, Least MSE is  4.85220982582  for Lambda Value =  31
      Now Calculating MSE on the TestData using Best choice lambda (obtained From CV) =  31
      For best choice lambda 31 corresponding test set MSE is  5.25247825109



********** Running Cross Validation for Traing DataSet -  train-150(1000)-100.csv *************
From CV, Least MSE is  4.87691289085  for Lambda Value =  47
      Now Calculating MSE on the TestData using Best choice lambda (obtained From CV) =  47
      For best choice lambda 47 corresponding test set MSE is  4.92900317377
```

Question 2 B).

In question 1(a), we used a given range of lambda values (from 0 - 150) and trained our model on the whole training dataset , and tested the model on test data set for the complete range of lambda values.

Whereas,

In question 2, we used 10 fold CV technique to obtain the best choice of lambda which was used in the later part to get the MSE's for the corresponding test data set. Cross validation technique is the industry standard term to validate the model since in real world use-case we can't lose the data by diving into training & test set. According to CV technique, we split the dataset in the multiple folds and use it's subset for training as well as test the model. Most common option for number of folds are 5,10 and LOOCV (leave one out cross validation). In our case we choose 10 folds cross validation.

In 10 fold CV approach, we ONLY used training dataset to train the model as well as test the model. We splitted the training dataset in to 10 folds and under each iteration we calculate the test set MSE by considering one fold as a test dataset to test our model and 9 other folds as training dataset to train our model. We then took the average MSE for all 10 folds. We did this for each lambda value ranging from 0-150. We then choose the best choice lambda which gives us the lease average MSE.

Following is the pseudo code for both the approaches which was implemented in the code.

Pseudo Code for 1 A):-

Load the test & Training dataset.
Build the feature list and output list (X Matrix & Y Matrix)
for each lambda ranging 0 - 150
    Calculate W Matrix using X,Y training matrix and lambda
    get MSE for training & test dataset by using same W Matrix

Pseudo Code for CV Technique:-

Load the test & Training dataset.
Split the training dataset into 10 folds
Build the feature list and output list (X Matrix & Y Matrix)
for each lambda ranging 0 - 150
    for each fold ranging 1-10
        build the CV test set X & Y matrix -- *Only from one fold for which the loop is running*
        build the CV training set X & Y matrix -- *from rest 9 fold*
        Calculate W Matrix using X,Y training set (CV) and lambda
        get MSE for test dataset (CV) by using same W Matrix
    take the average for MSE from each fold.
Find the least average MSE and its corresponding lambda value -- *best choice lambda*
retrain the entire training dataset using the best choice lambda

Question 2 C).

Cross Validation may requires to do too much heavy computation in case our prediction algorithm is very complex.

Computational performance of a CV algorithm can increase very high in case of the big-data.

Choosing wrong CV technique can also leads to bias-variance trade-off.

Question 2 D).

There are mainly two factors which effects the performance of CV. **Bias and Variance**

Bias :- The Validation set approach leads to a overestimate the test errors since in this approach we divide the data into two parts, and only one part of the data is used to build the statistical model. Using this logic, we can say that LOOCV technique will give almost unbiased test errors because we use almost complete data (n-1) to build the model under this approach. Whereas, K-fold CV technique will use (k-1)n/k size data to build the model - which is fewer then we use in LOOCV. Therefore, from bias reduction prospective, LOOCV is preferred over K-Fold CV.

Variance :- According to LOOCV technique, we average the output of n fitted models which is trained on almost identical size of the data (n-1). Therefore the outputs for all n folds are highly correlated with each other. Whereas, in K-fold technique, we get the model output which is somewhat less correlated with each other due to the less overlap between the data for each fold. Since the mean of highly correlated values has the higher variance than the mean of less correlated values. Therefore the test errors estimated from LOOCV tends to have the higher variance than k-fold CV.

To summarize, there is a bias-variance trade off associated with the choice of Cross Validation technique, which is the major area of research currently.

Apart from these two factor, the size of the data can also increase the computational performance of CV since it runs many folds on the training dataset to average the test errors.

Question 3.

Following are the plots generated for Learning curve for 1000-100 data set with the lambda values of 1,25,150 . (*The same can be generated by the code as well; mentioned in the readme file*)

Note :- Next run of the code may generate slightly different plots since we are using the random sampling of the data. However, the plot shape would remain very similar to this.