Assignment 3 - ECEN 689

Name: Prabhasa Kalkur

UIN: 127003549

General Organization of the report: Problem Description, Setup, Subsection results, Accuracy plots, Analysis.

PROBLEM 1

Problem Description - Compare Linear Least Squares and Logistic Regression for learning a linear classifier for a 2-class problem. I followed the pseudocode below:

a) Learn the classifier for varying sample sizes using training data on `fitlm` and `fitclinear` functions, respectively
b) Use the learnt classifier's properties to display and plot the estimated model. I have attached 2D and 3D scatter plots corresponding to the two methods
c) Employ Test data on the predicted model via the `predict` function
d) Accuracy of the classifier is the ratio of correctly estimated labels to true labels (the test label depends on the 'threshold' of the classifier)
e) Further analysis is made for the resulting plots. Repeat for 1b and 1c

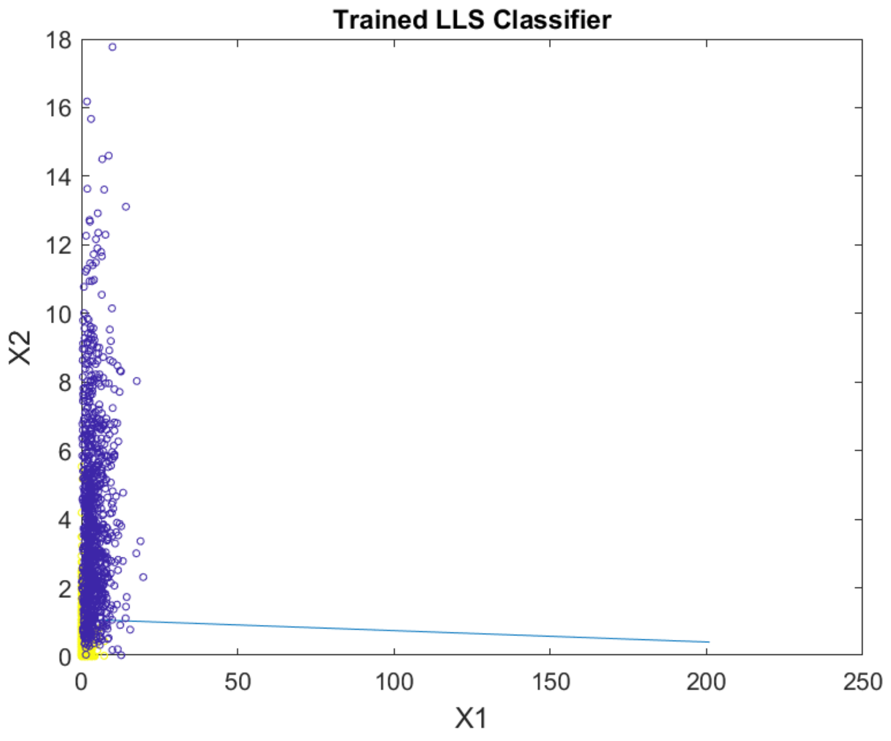
Setup – a) Sample sizes of [10 50 100 500]
b) Computed average accuracy over 200 iterations
c) No Regularization used (for fairness of comparison between logistic and least squares). The 'lambda' parameter was set to zero during comparison

Subsection results – Accuracy of the classifiers are the only performance measures. The plots are threefold, and remain the same throughout the subproblems:

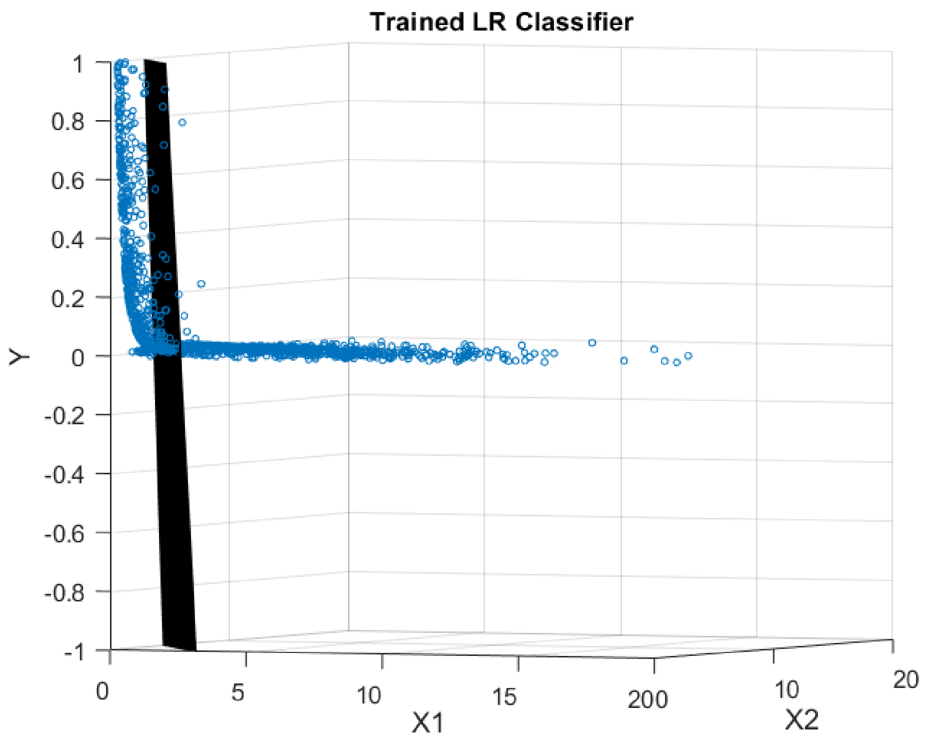
a) The trained Linear Least Squares (LLS) model
b) The trained Logistic Regression (LR) model
c) The Test set accuracy of the two models

```
Problem 1a

iterations =

   200


samples =

    10    50   100   500

The learnt linear classifier is:

ans =

0.7484 - 0.1802*x2 - 0.1506*x1

The learnt logistics classifier is:

ans =

4.609 - 1.776*x2 - 1.254*x1


avg_accuracy_LS =

    0.1328
    0.1088
    0.1121
    0.1071


avg_accuracy_LR =

    0.8922
    0.9281
    0.9316
    0.9357

Error using axis (line 241)
Too many output arguments.

Error in test_P1 (line 131)
axis2=axis([5 500 0 1]);

>>
```

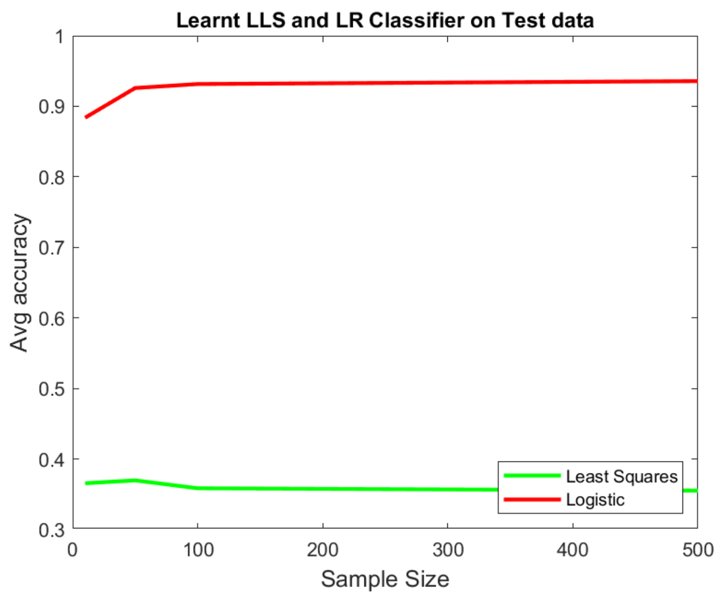
Problem 1a



LLS - Even with ~40% accuracy on the Test set, the classifier fits the training set poorly. This could be owing to the true distribution parameters, which makes it hard to find linear models. The learnt classifier has its coefficients as shown in the output above.



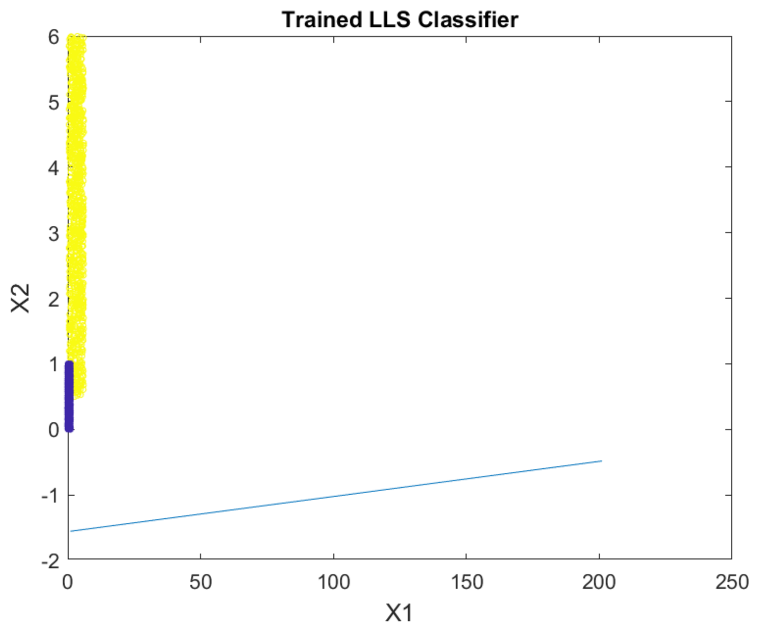
LR – The classifier performs much better and fits the data well, as shown in the 3D scatter plot. With almost 90% accuracy, the Test set can also be modeled as a Gamma distributed data.



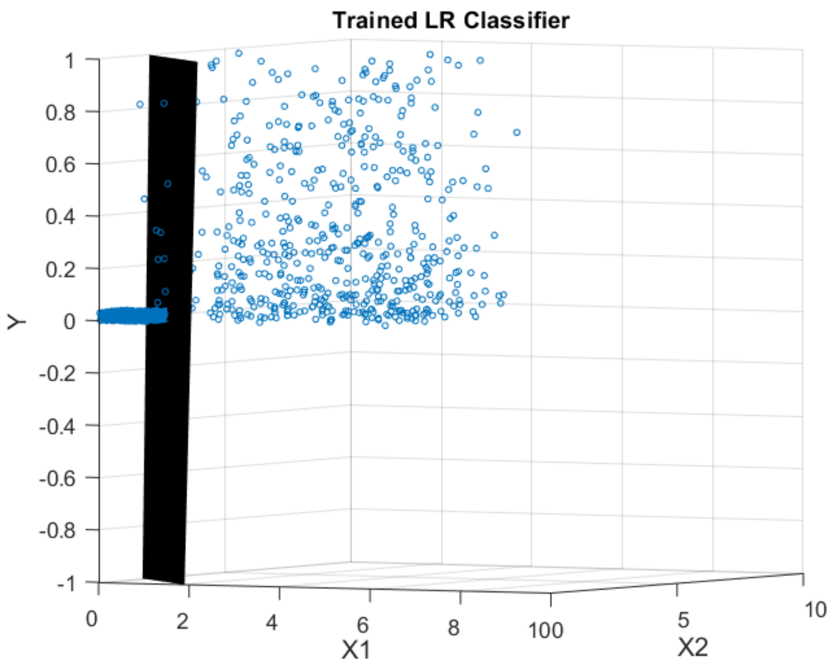
Accuracy plots - Turns out that variation in sample size does not significantly affect the performance of the classifier. This points to a 'distributed' Test data with less outliers. As expected, LLS performs much poorly in comparison to LR, even without the regularizer.

The plots and analysis are similar for the remaining subproblems:

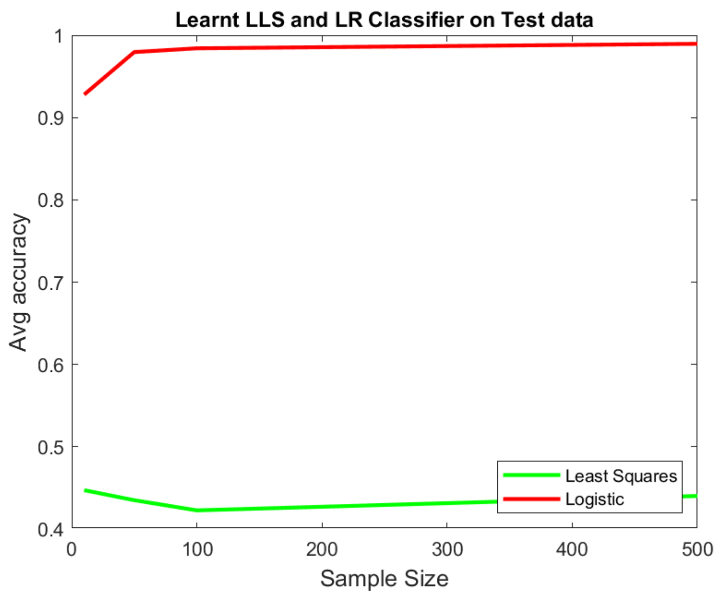
Problem 1b



LLS – This performs even worse than the earlier problem? Not true, as the accuracy plot show.



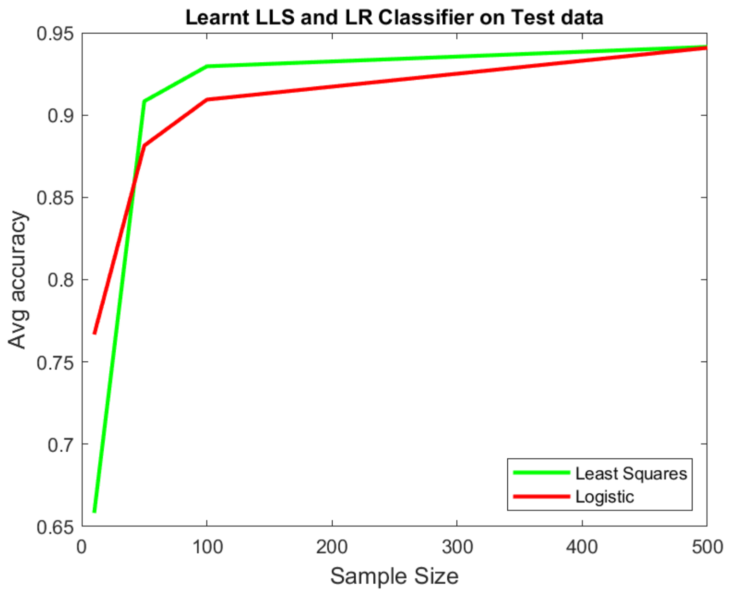
LR – The Classifier is much better than the previous subproblem.



<u>Analysis</u> - Note that the above plots are WITHOUT any regularization. As it turns out, adding an additional 'lambda' parameter improves the performance only slightly. In the cases of 1a and 1b, **by about 2-3%**, which is significantly low in comparison to the ever-growing gap between LLS and LR. Or so we think, until:

```
>> iterations

iterations =

   200

>> vpa(model_LLS(x1,x2),4)

ans =

0.7191 - 0.1685*x2 - 0.1575*x1

>> vpa(model_LR(x1,x2),4)

ans =

5.31 - 1.192*x2 - 1.146*x1

>> avg_accuracy_LLS

avg_accuracy_LLS =

    0.6583
    0.9084
    0.9296
    0.9414

>> avg_accuracy_LR

avg_accuracy_LR =

    0.7667
    0.8814
    0.9094
    0.9408

>>
```

Problem 1c



(I am unsure of the reason behind a drastic change in performance improvement in this case. I might have been wrong somewhere in the code)

PROBLEM 3

Problem Description – The German dataset consisted of a 24-dimensional (numbered A through X) 2-class classification problem, with an additional column of class labels. The task is essentially the same as Problem 1, with the catch that the Train and Test sets are to be taken from the same file.

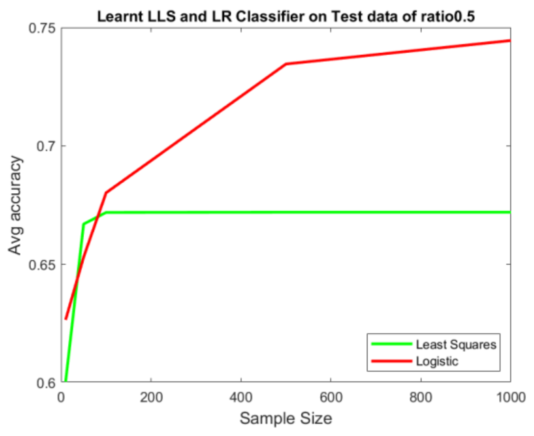
Setup – a) Sample sizes of [10 50 100 500 1000]

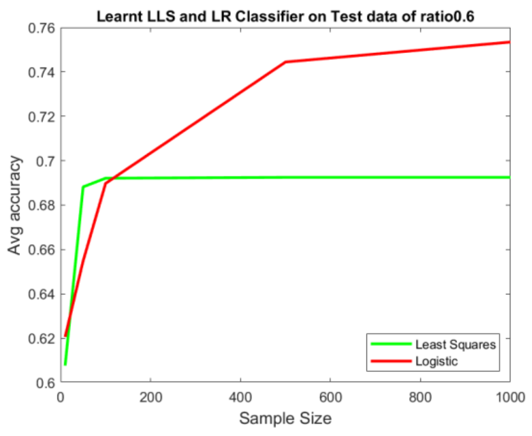
b) Computed average accuracy over 200 iterations

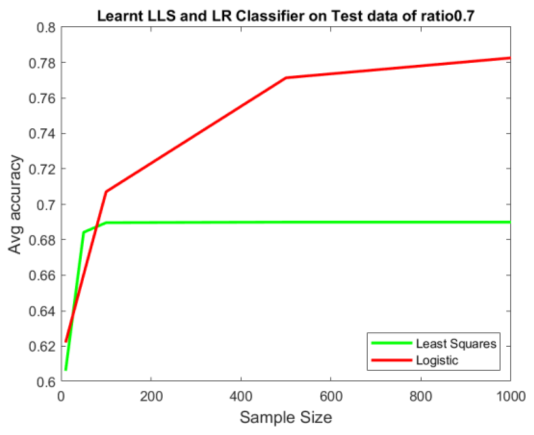
c) No Regularization used (for fairness of comparison between logistic and least squares). The 'lambda' parameter was set to zero during comparison

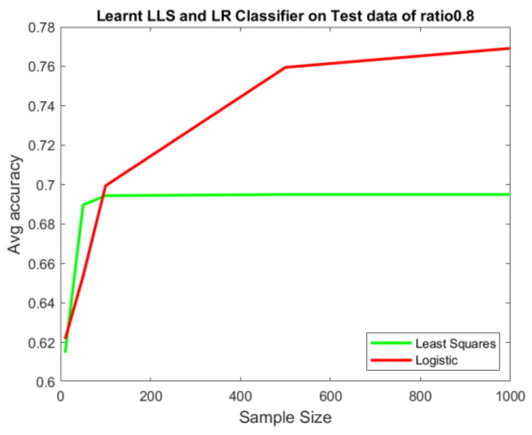
d) Compare the accuracy of different Train to Test set ratios (0.5 to 0.9)

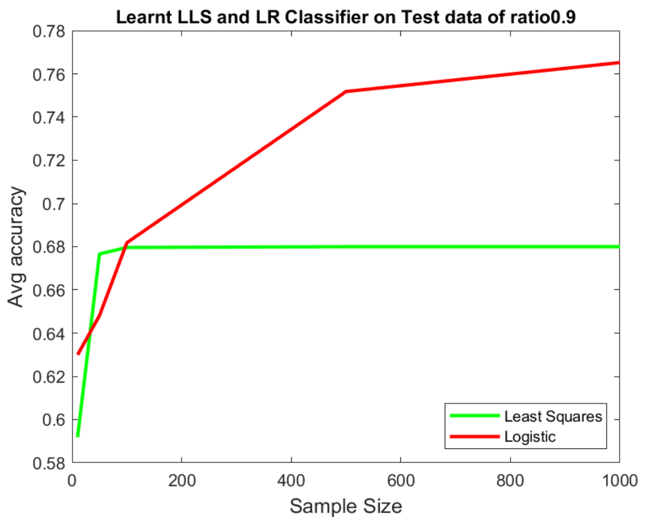
Subsection results, Plots and Analysis – Accuracy of the classifier is the only performance measure. The Test set accuracy of the two models was plotted against the sample size. As before, the performance improved only slightly after Regularization, hence has not been included as a separate plot. Further exploration can be made via techniques such as Precision-Recall, that help justify the choice of Train to Test ratio. Partition of the search space, and usage of decision trees can also be looked at (have not explored the above).







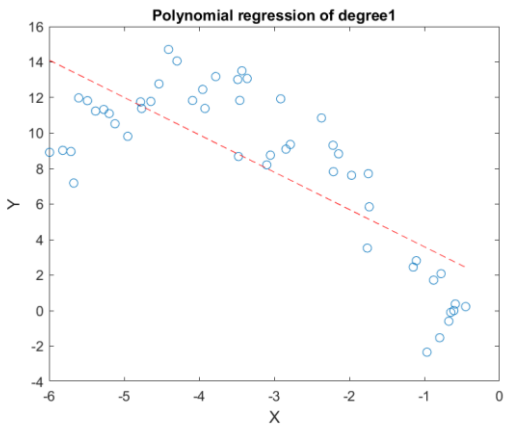


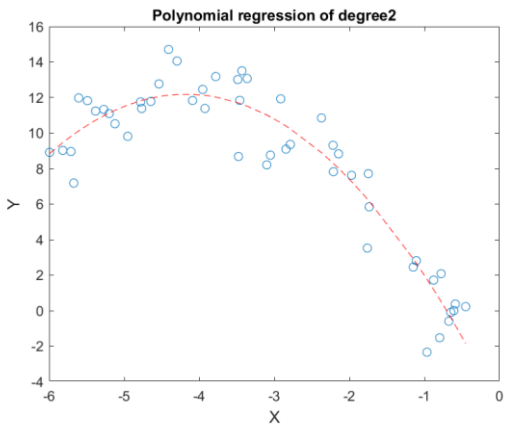


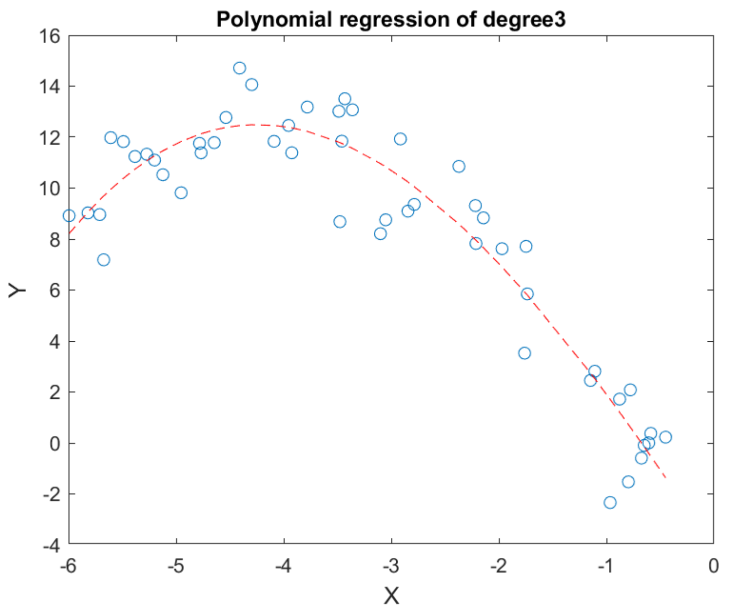
It turns out that the classifiers' performance is almost **similar across different ratio values**. The accuracy peaks somewhere between having 60% and 80% training data. The ratio and accuracy also **depend on the amount of Class-1 and Class-2 labels** among the Train-Test sets. Sample sizes larger than the train set were chosen to understand **bagging (repeatedly drawing samples with replacement)**, which essentially helps reduce variance and avoid overfitting. The plot shows that it helps improve LR accuracy, whereas LLS remains constant.

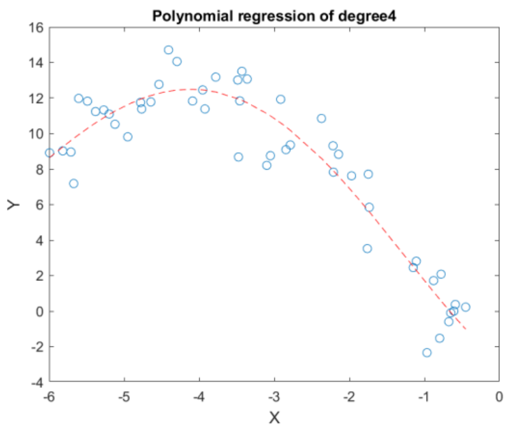
PROBLEM 4

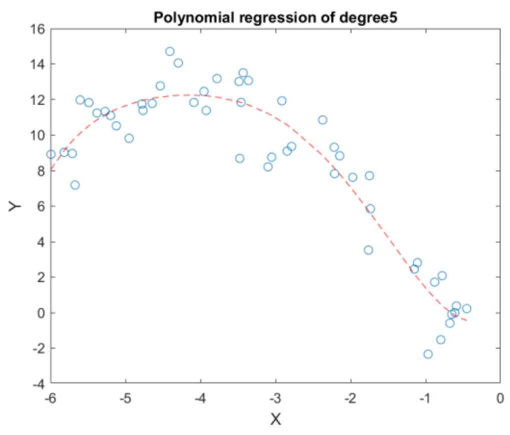
Problem description and plots - The 1D Regression problem is basically that of curve fitting. The definition of 'Overfitting' depends on the error between the Train and Test data. Thus, the best-fit polynomial can only be decided according to the Test data. The polynomial coefficients and the total squared error for each degree of the regression model is as below.

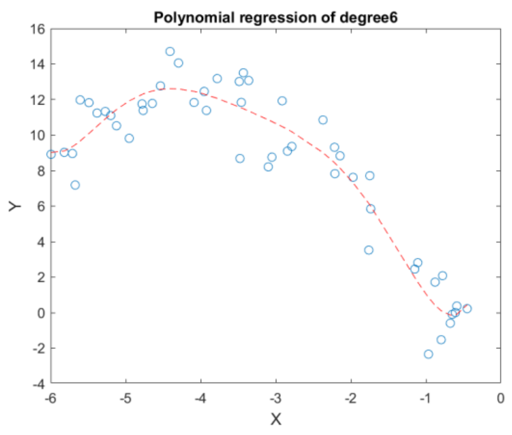


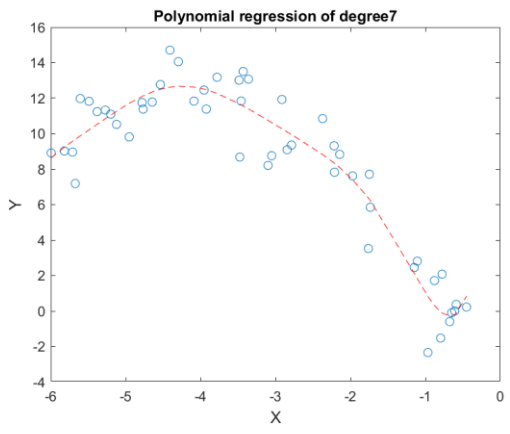


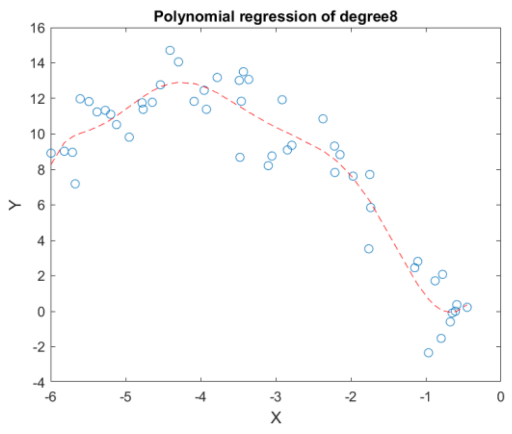








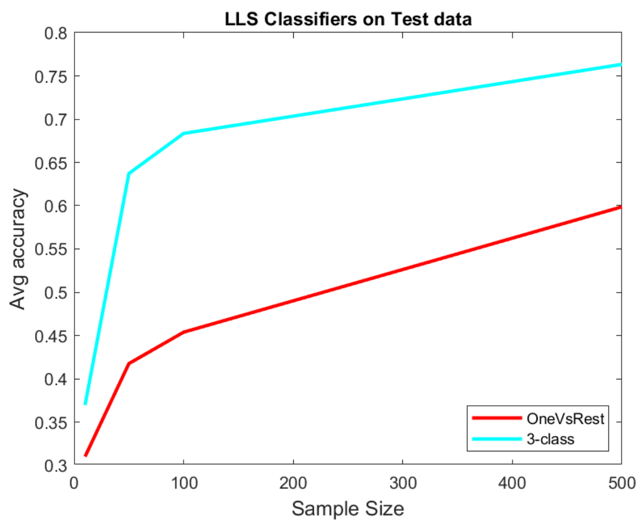




It can be seen that **although the total error reduces with increasing degree, the model also overfits the data**. This tradeoff can be studied w.r.t Test data.

PROBLEM 2

Analysis - This is similar to Problems 1 and 3, in that the LLS model is evaluated using **`fitcecoc`** instead of `fitclinear`. It shows that solving a 3-class problem is more accurate than taking a One Vs Rest approach. This matches with our discussion in the class



```
Degree of polynomial = 1

ans =

   -2.1026    1.4932

Total squared error = 435.185969
Degree of polynomial = 2

ans =

   -1.0104   -8.4400   -5.4365

Total squared error = 111.254248
Degree of polynomial = 3

ans =

    0.0779   -0.2664   -6.4703   -4.2179

Total squared error = 107.918623
Degree of polynomial = 4

ans =

    0.0343    0.5196    1.6331   -3.4146   -2.8358

Total squared error = 106.677803
Degree of polynomial = 5

ans =

    0.0318    0.5505    3.5885    9.7606    5.7709    0.4809

Total squared error = 104.506018
Degree of polynomial = 6

ans =

    0.0345    0.6969    5.5148   21.6434   42.7045   33.4853    8.6195

Total squared error = 98.435929
Degree of polynomial = 7

ans =

    0.0129    0.3255    3.3452   17.9283   53.4402   86.2082   62.1647   15.5546

Total squared error = 97.004430
Degree of polynomial = 8
```

```
ans =

   -0.0101   -0.2498   -2.5024  -13.0163  -37.0731  -54.9226  -33.6750   -4.4605↙
1.4785

Total squared error = 95.180353
>>
```