Assignment-1

Machine Learning- CS584

By- Nivedita Kalele

CWID- A20329966

Parametric Regression

Introduction

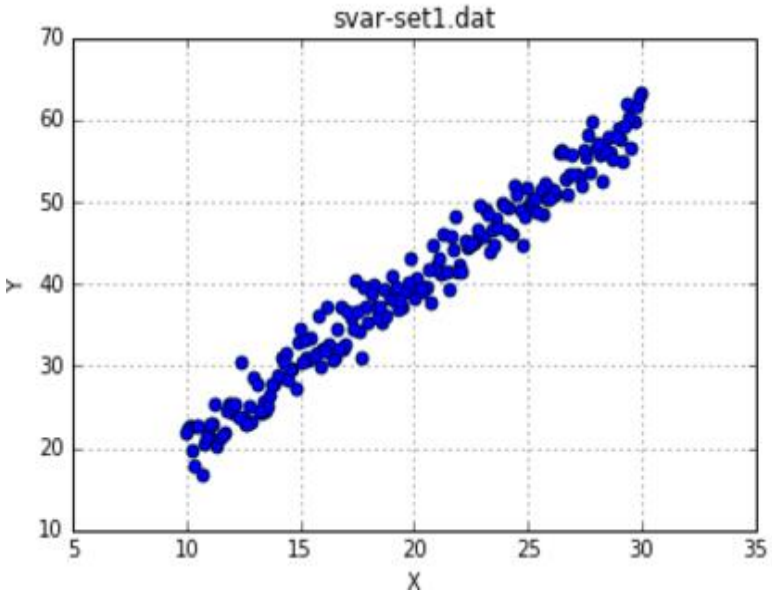
In this assignment we have implemented the model which will perform Linear and polynomial regression on the given data. Regression estimates the parameters of the data. Once these parameters are found we can use these parameters for prediction on other datasets. And we can get the whole distribution. So, in this assignment parameters are estimated on the given data by applying model and predictions are done on the test data.

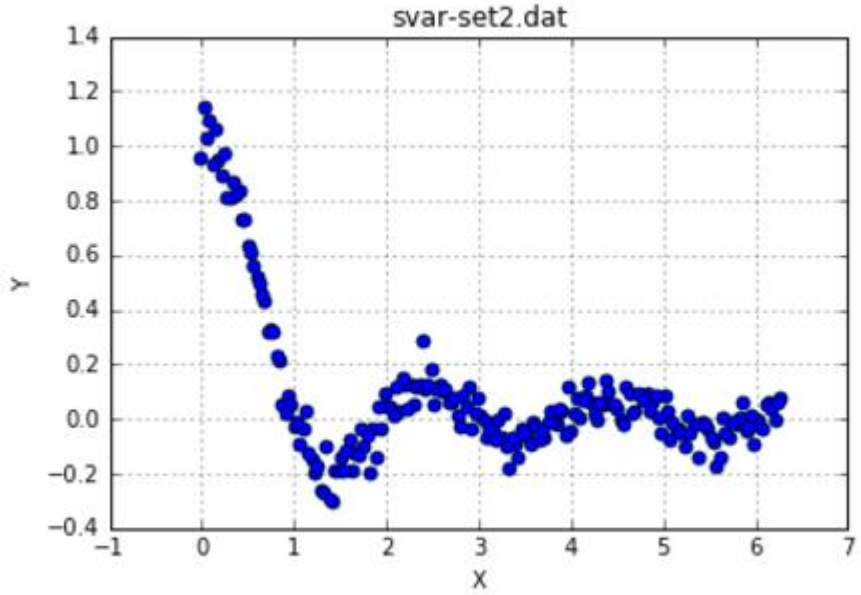
1. **Univariate data:**
   svar-set1.dat, svar-set2.dat, svar-set3.dat and svar-set4 are the four datasets given. Single feature is there in these files.

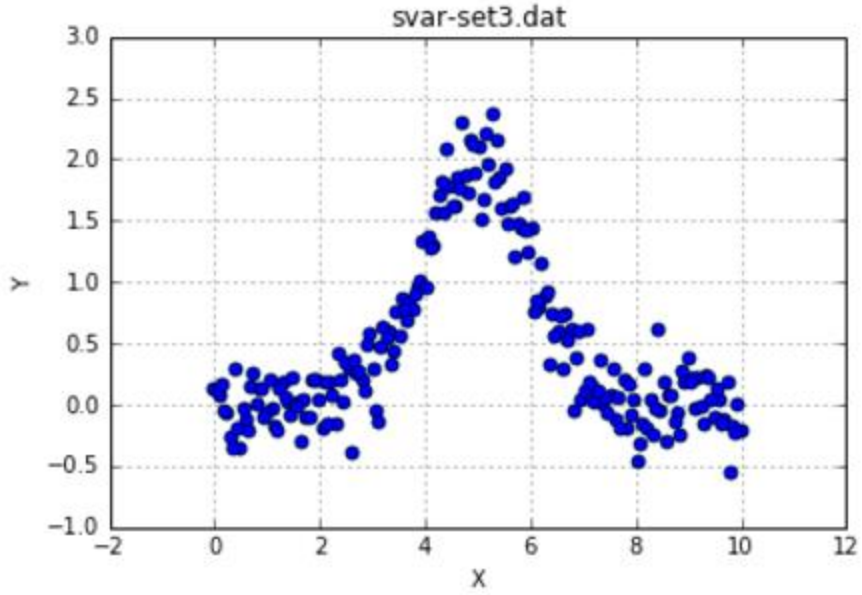
   1.1 Loading the data:
   The data is loaded directly from the url using urlopen and loadtxt functions.
   The plotting of the data is done using plot function of matplotlib.

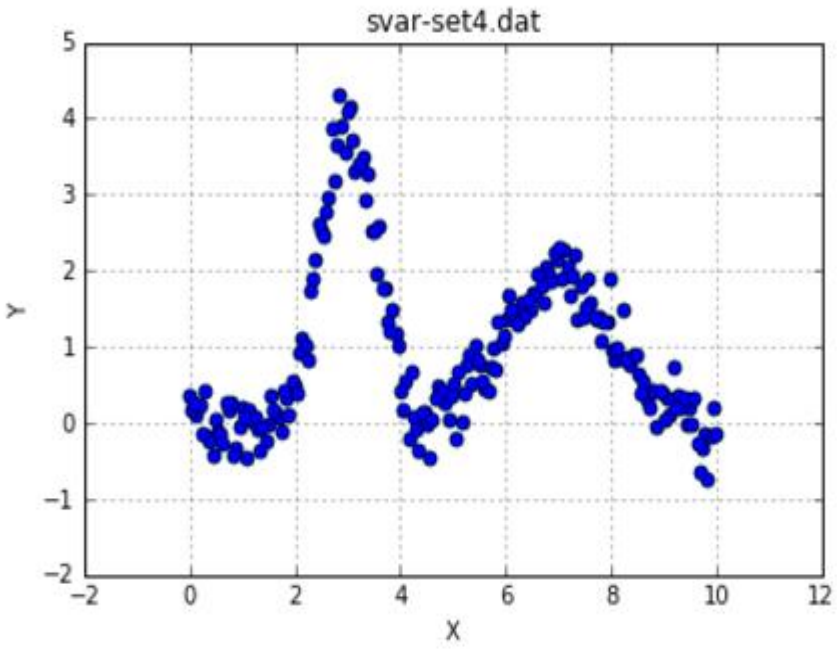
   Input Data

   Data Plots-




svar-set2.dat
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
Y
−1
0
1
2
3
4
5
6
7
X


svar-set3.dat
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
Y
−2
0
2
4
6
8
10
12
X



1.2 <u>Linear Regression:</u>

For applying Linear regression we did split the data using 10 fold cross validation in the function k_fold and then fit_model function is applied on the training dataset. The parameters obtained from this are applied on the testing dataset.

For linear regression formula used is:

Y_hat = theta0 + theta1*x

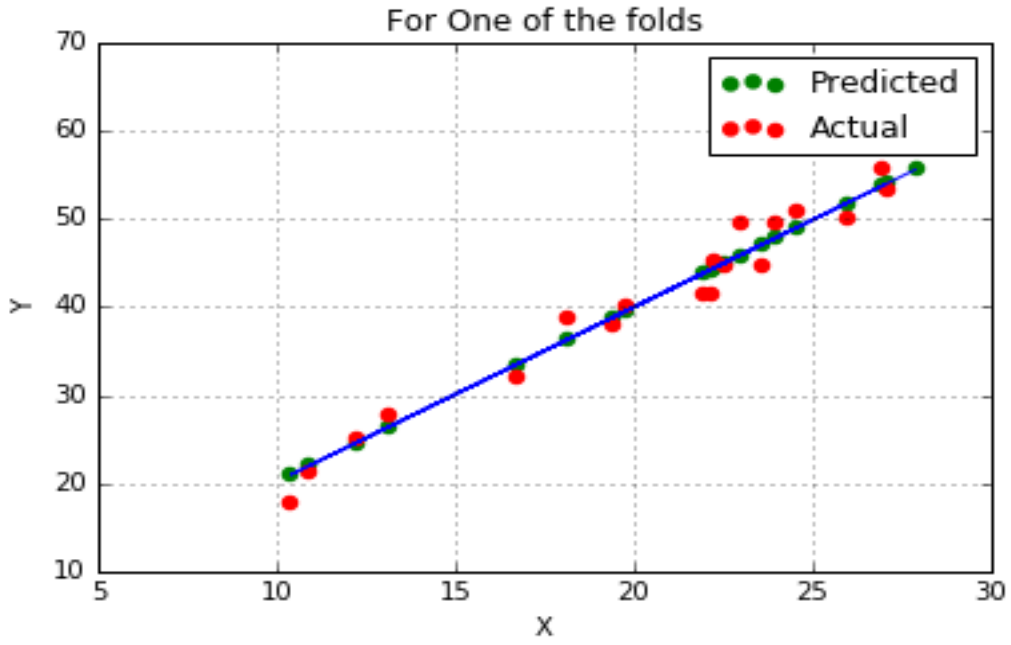
For Mean square error:

MSE= (y_hat-y)^2

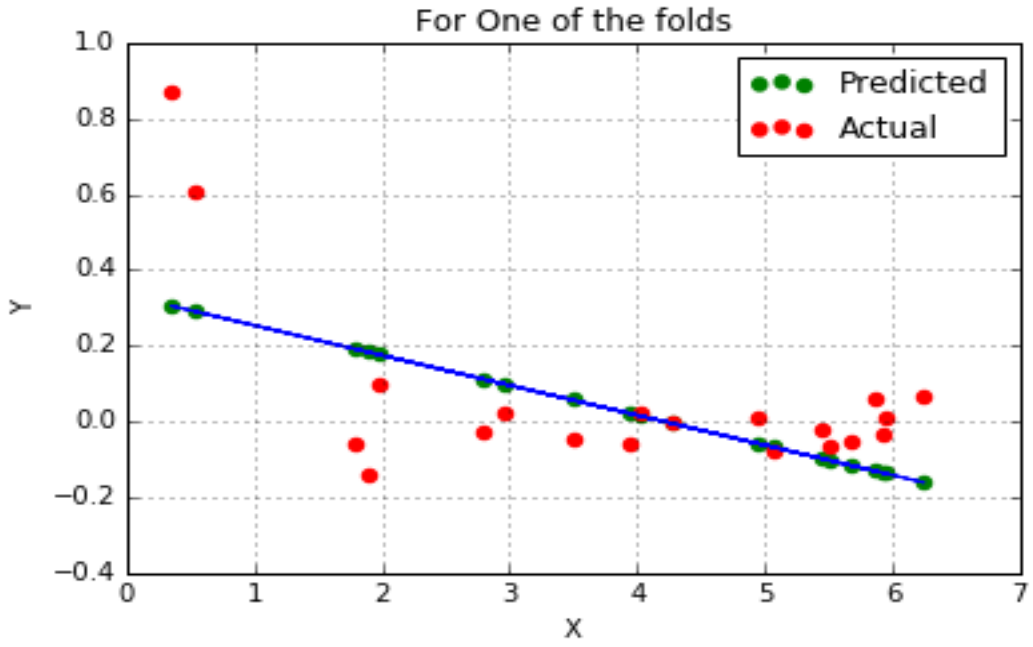
Here, y_hat is the prediction.

Plots obtained after applying model on training data and predicting on test data:

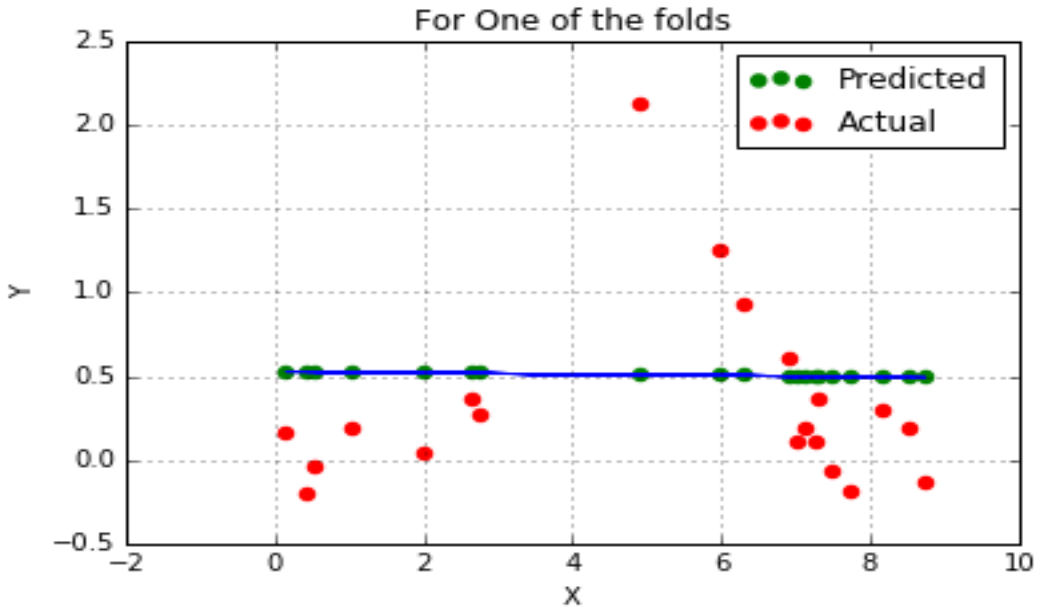
For svar-set1.dat



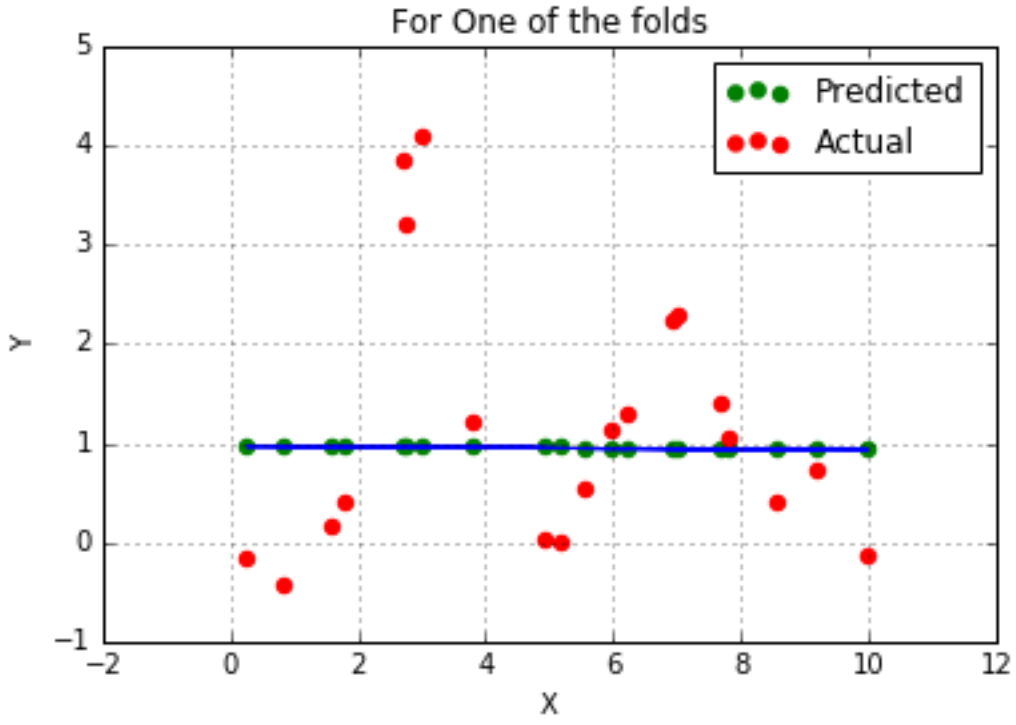
For svar-set2.dat



For svar-set3.dat



For mvar-set4.dat



Comparison of applying model on training data and predicting on training and testing datasets

	Predicting on training	Predicting on testing
Dataset1	4.36875256474	4.22549268163
Dataset2	0.061200692676	0.0594871873194
Dataset3	0.504339248263	0.498417166302
Dataset4	1.21228704771	1.20020250224

Comparison of readymade model and my model

	My model	Readymade Function
Dataset1	4.36875256474	4.22549268163
Dataset2	0.061200692676	0.061200692676
Dataset3	0.504339248263	0.504339248263
Dataset4	1.21228704771	1.21228704771

MSE with Reduced Training Data

Dataset1: 4.29856341131
Dataset2: 0.0531134130605
Dataset3: 0.566798369649
Dataset4: 1.57115125797

Conclusion:

We can see that error after predicting model on training dataset is more than predicting it on test dataset. We can say that, model fits better on test data set.

1.3 **Testing on different polynomial model**:

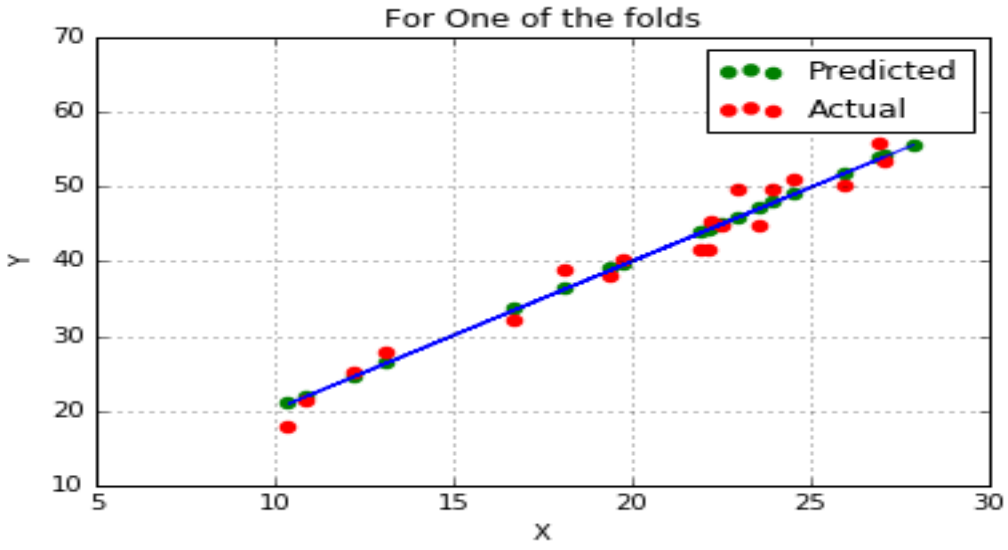
Polynomial regression using formula:

Y_hat= theta0 + theta1*x + theta2 * x^2+ ……..+ theta_n*x_n^2

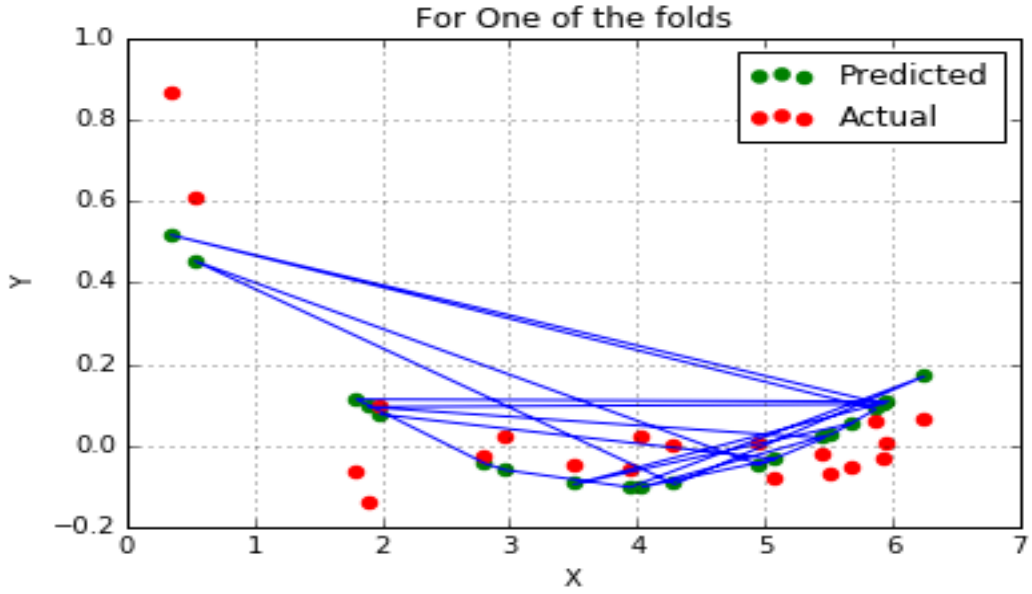
Degree=2

Plots after applying model on test data:

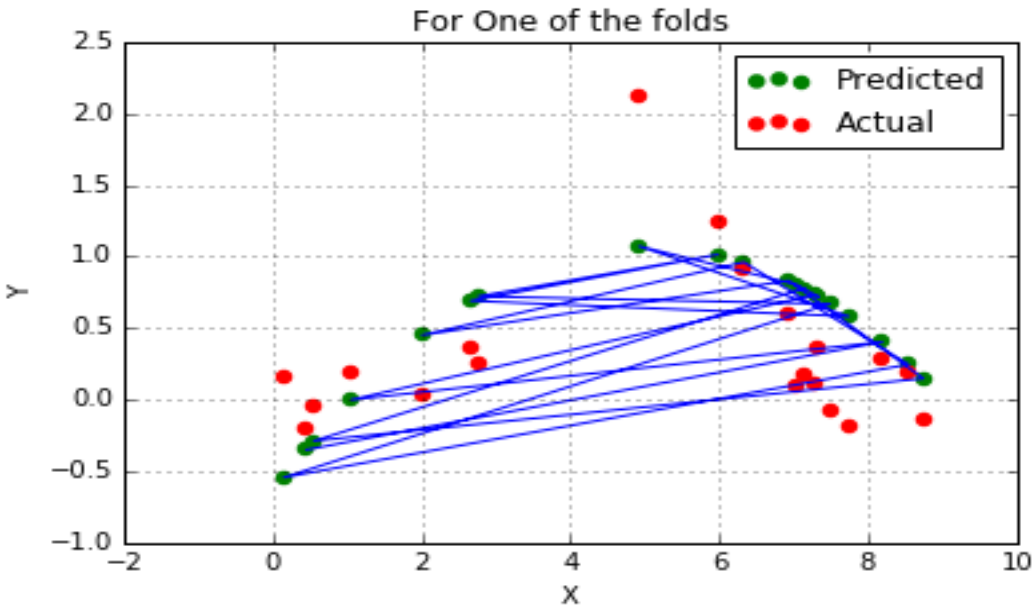
For svar-set1.dat



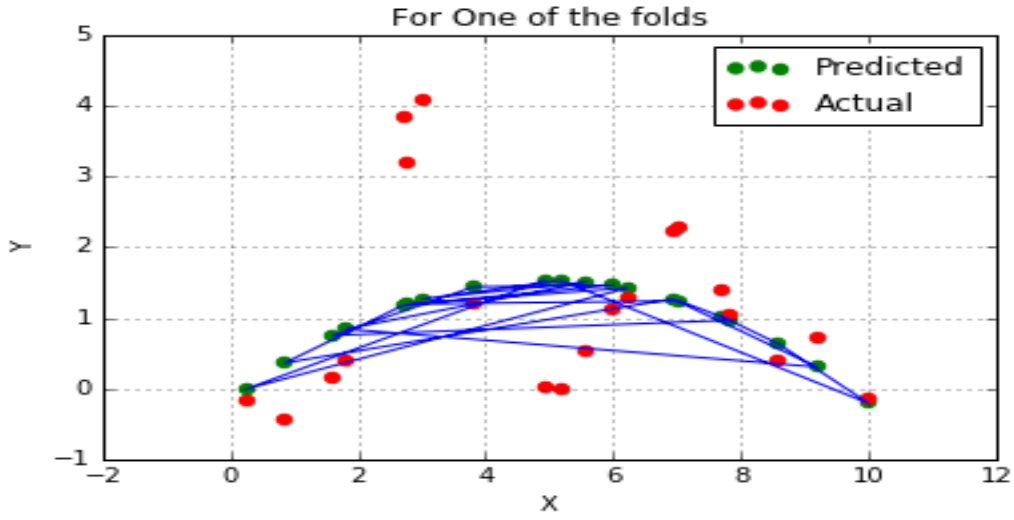
For svar-set2.dat



For svar-set3.dat



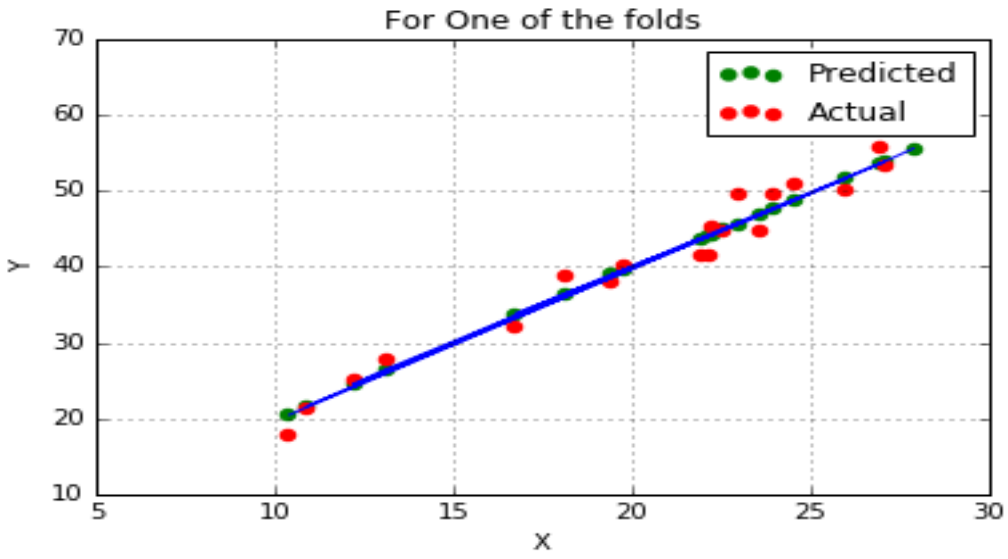
For svar-set4.dat



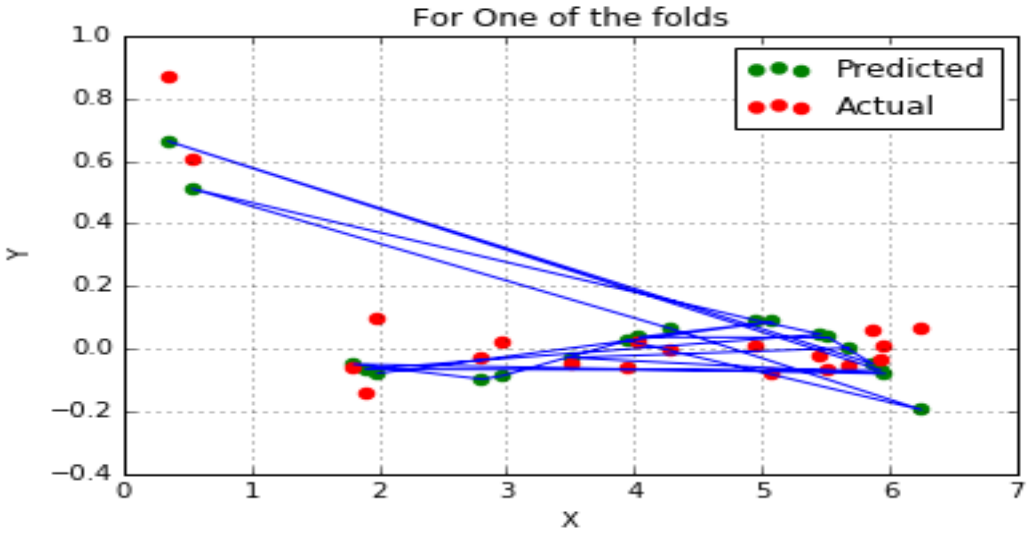
Degree=3

Plots after applying model on test data:

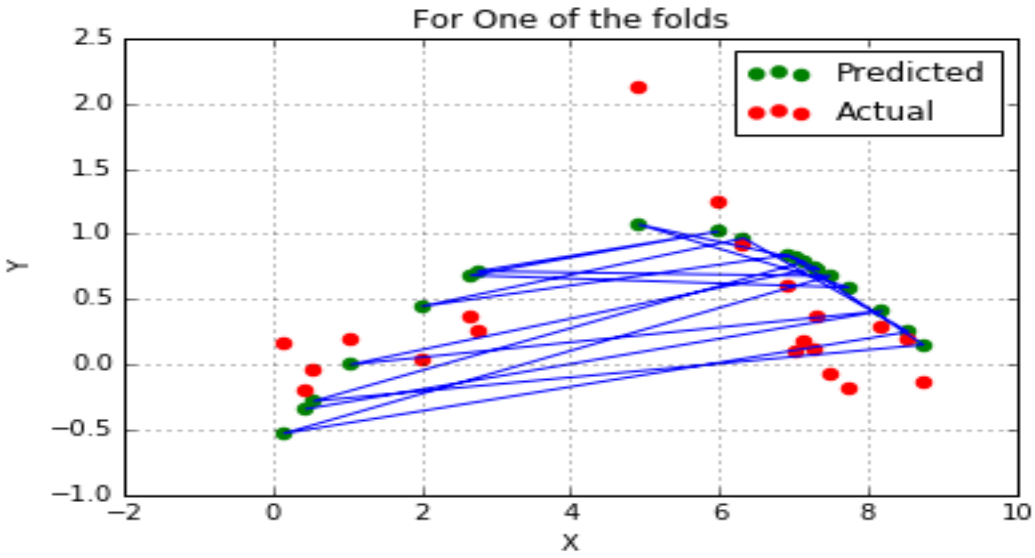
For svar-set1.dat



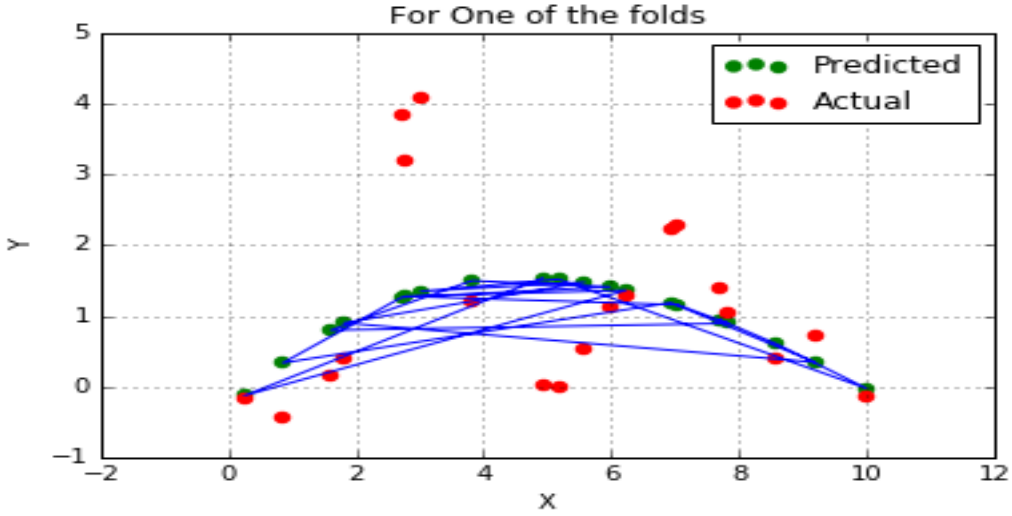
For svar-set2.dat



For svar-set3.dat



For svar-set4.dat



Test on Reduced train data:

For this part I did split data as 75% and 25% which makes training data of size 150. Earlier it was 180 as data was divided using 10 fold cross validation.

Errors on test sets with **degree 2**

1. Svar-set1.dat

```
Theta: [-0.03139  2.01791 -0.00079]
MSE with defined algorithm: 4.43075071194
MSE with Reduced Training Data: 4.34355934721
```

2. Svar-set2.dat

```
Theta: [ 0.65072 -0.38322  0.04864]
MSE with defined algorithm: 0.0401079638308
MSE with Reduced Training Data: 0.0313576616778
```

3. Svar-set3.dat

```
Theta: [-0.59112  0.65606 -0.06583]
MSE with defined algorithm: 0.260585516772
MSE with Reduced Training Data: 0.299849287794
```

4. Svar-set4.dat

```
Theta: [-0.15142  0.68784 -0.06932]
MSE with defined algorithm: 0.940830430186
MSE with Reduced Training Data: 1.18469433286
```

Errors on test sets with **degree 3**

1. Svar-set1.dat

```
Theta: [-11.83040  3.99671 -0.10493  0.00173]
MSE with defined algorithm: 4.44976586085
MSE with Reduced Training Data: 4.21425175252
```

2. Svar-set2.dat

```
Theta: [ 0.99849 -1.05577  0.31689 -0.02846]
MSE with defined algorithm: 0.021370426046
MSE with Reduced Training Data: 0.0189648710986
```

3. Svar-set3.dat

```
Theta: [-0.59459  0.66051 -0.06695  0.00007]
MSE with defined algorithm: 0.262182711888
MSE with Reduced Training Data: 0.304931547546
```

4. Svar-set4.dat

```
Theta: [-0.39888  0.98910 -0.14489  0.00504]
MSE with defined algorithm: 0.938203377002
MSE with Reduced Training Data: 1.1623516602
```

Conclusion:

As we can see, for dataset1 mse is less for degree 2 than degree 3.So we can say that for degree 2 dataset1 fits well.

For dataset2, mse is less for degree 3 than degree 2. So this dataset fits well for degree 3.

For dataset3, mse for both degree2 and degree 3 is almost equal.

For dataset4, mse is less for degree 3 than degree 2. So this dataset fits well for degree 3.

Also, For dataset1 and dataset2 after reducing training dataset for all degrees 1, 2 and 3 mse reduces. i.e model fits on these datasets after reducing training dataset.

For dataset3 and dataset4, mse increases after reducing training dataset for all degrees 1, 2 and 3 i.e model fits well for large training dataset.

2. Multivariate data:

mvar-set1.dat, mvar-set2.dat, mvar-set3.dat and mvar-set4 are the four datasets given. Single feature is there in these files.

2.1 Loading the data:

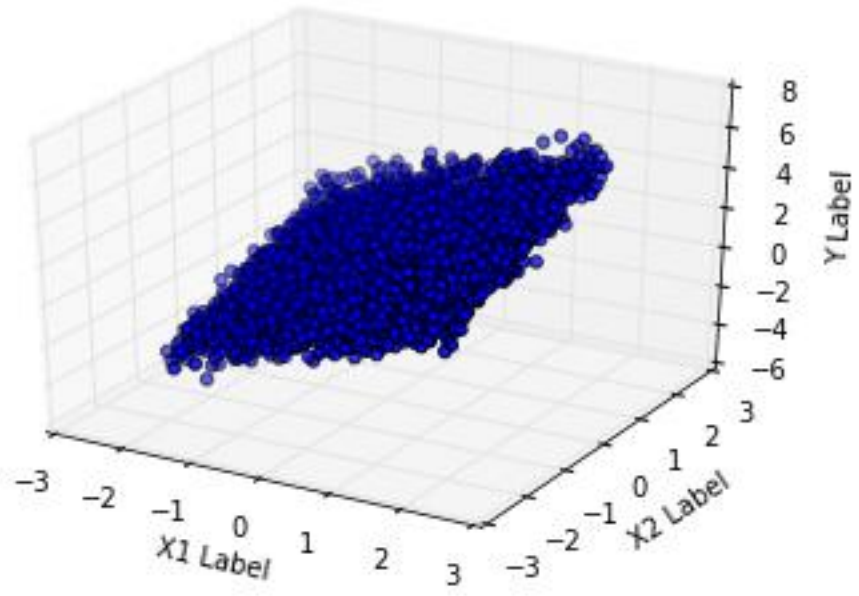
The data is loaded directly from the url using urlopen and loadtxt functions.

The plotting of the data is done using plot function of matplotlib.

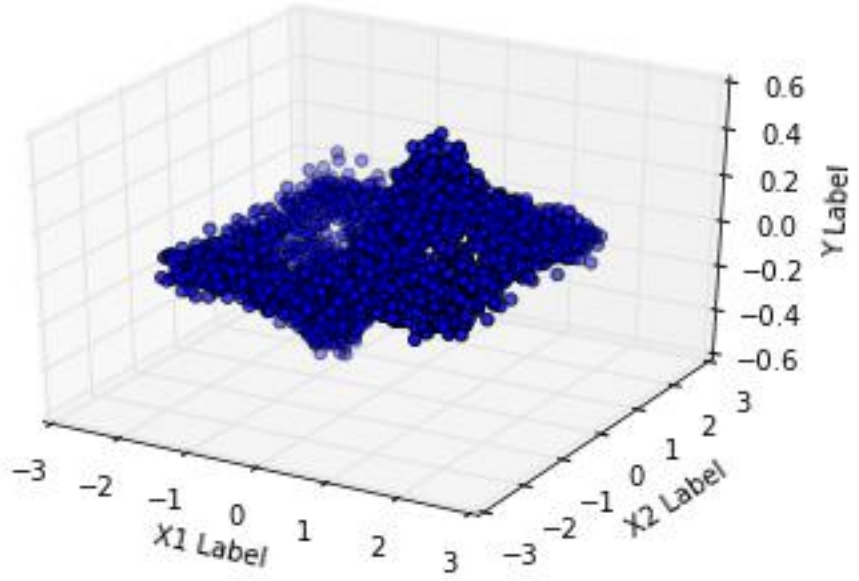
Input Data

Data Plots-

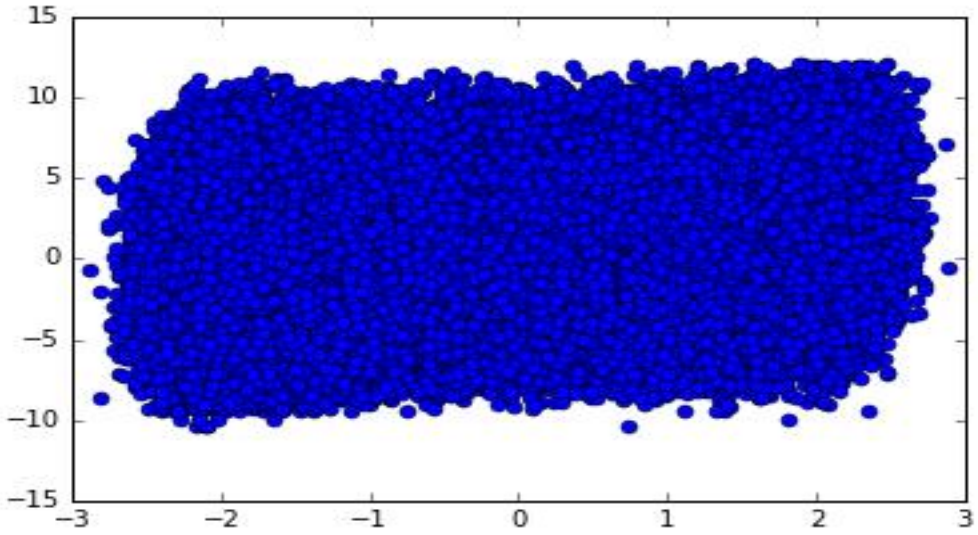
For mvar-set1.dat



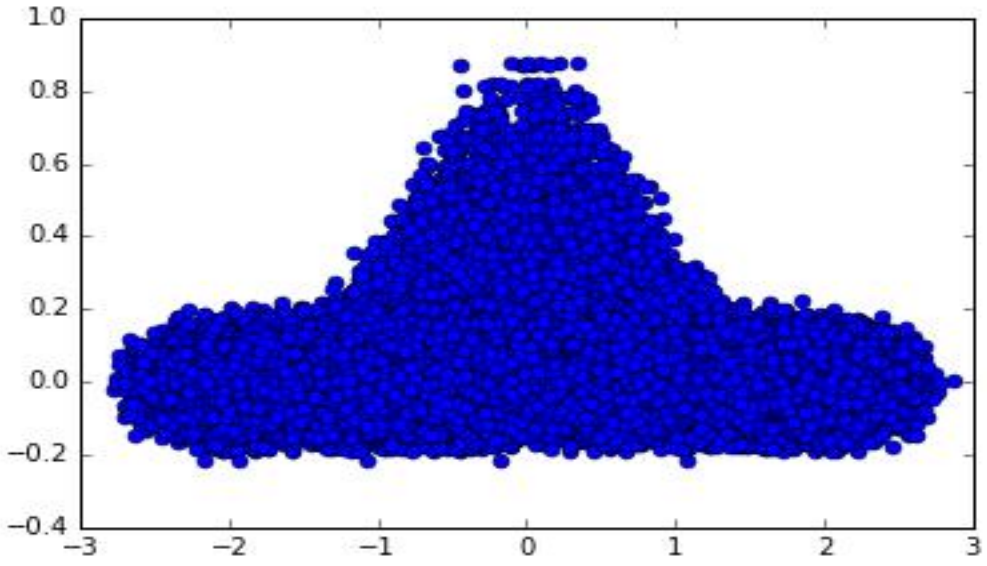
For mvar-set2.dat



For mvar-set3.dat



For mvar-set4.dat



2.2 Linear Regression:

For applying Linear regression we did split the data using 10 fold cross validation in the function k_fold and then fit_model function is applied on the training dataset. The parameters obtained from this are applied on the testing dataset.

For linear regression formula used is:

Y_hat = theta0 + theta1*x

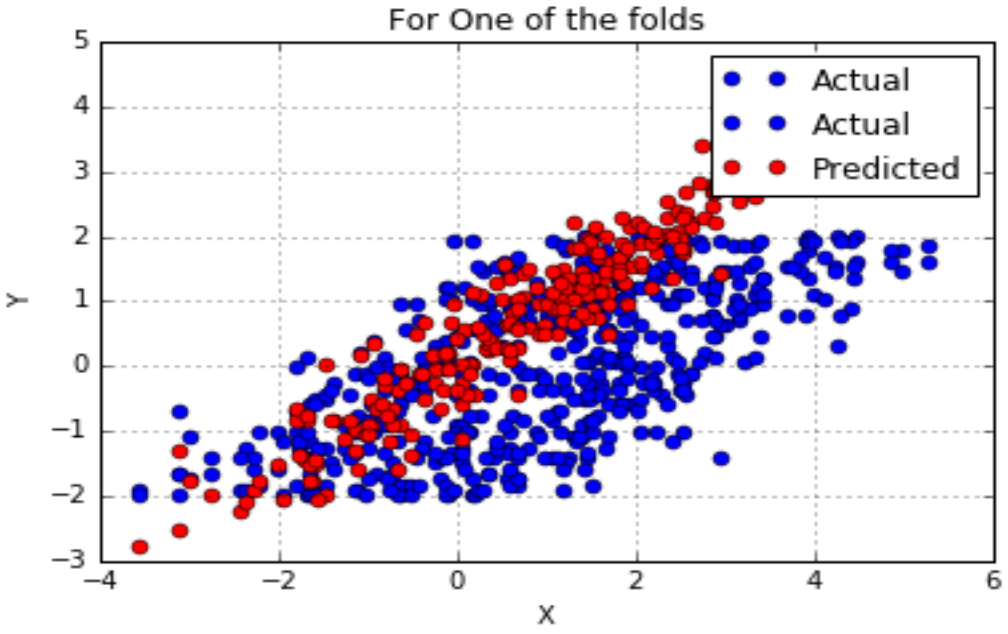
For Mean square error:

MSE= (y_hat-y)^2

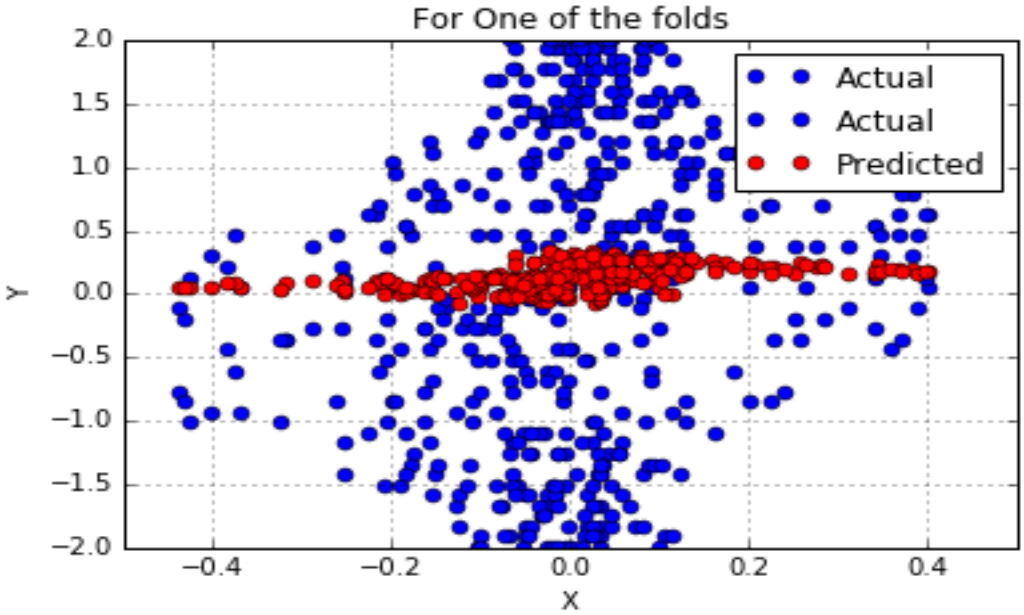
Here, y_hat is the prediction.

Plots obtained after applying model on training data and predicting on test data:

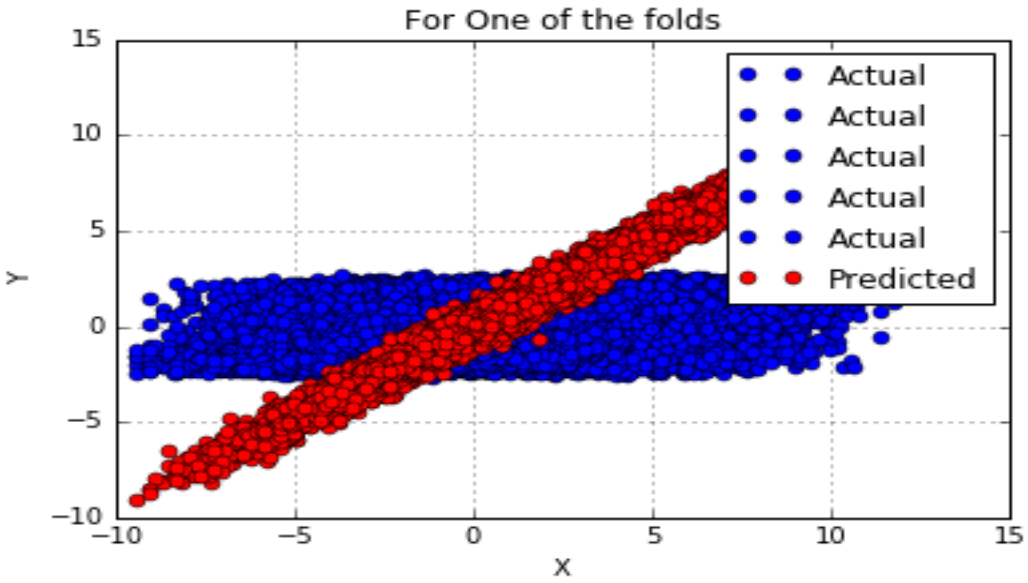
For mvar-set1.dat



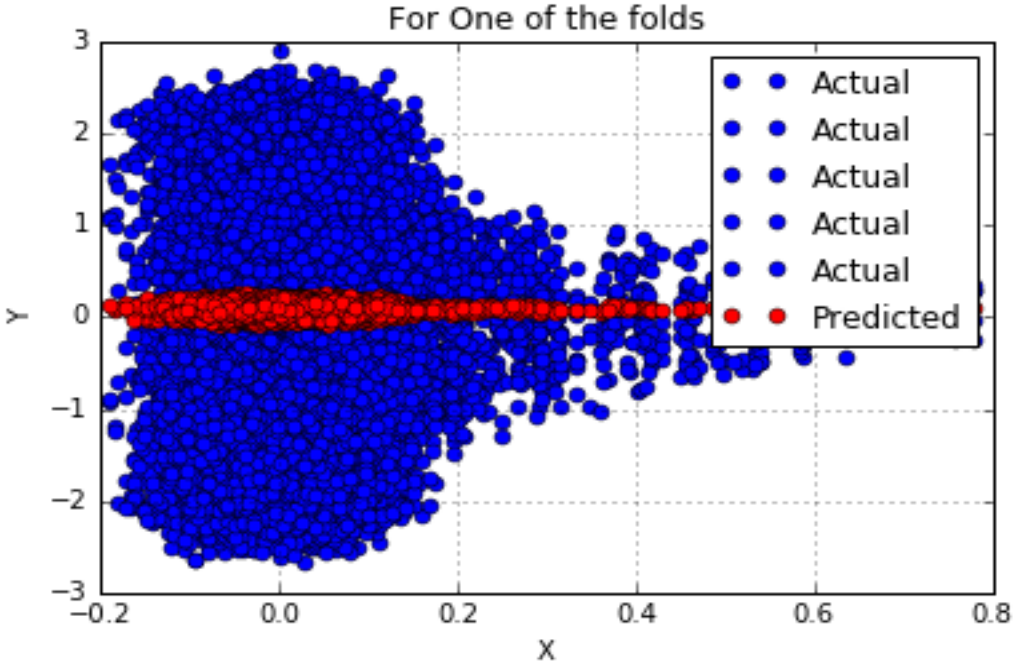
For mvar-set2.dat



For mvar-set3.dat



For mvar-set4.dat



Comparison of applying model on training data and predicting on training and testing datasets

	Predicting on training	Predicting on testing
Dataset1	0.258679377423	0.259149055236
Dataset2	0.0199095055921	0.0199567817456
Dataset3	0.250741123643	0.250781409474
Dataset4	0.00418915379927	0.00418950268543

Comparison of readymade model and my model

	Readymade	My model
Dataset1	0.259149055236	0.259149055236
Dataset2	0.0199567817456	0.0199567817456
Dataset3	0.250781409474	0.250781409474
Dataset4	0.00418915379927	0.00418950268543

Conclusion:

We can see that error after predicting model on training dataset is almost equal to predicting it on test dataset. We can say that, model fits better on both data sets.

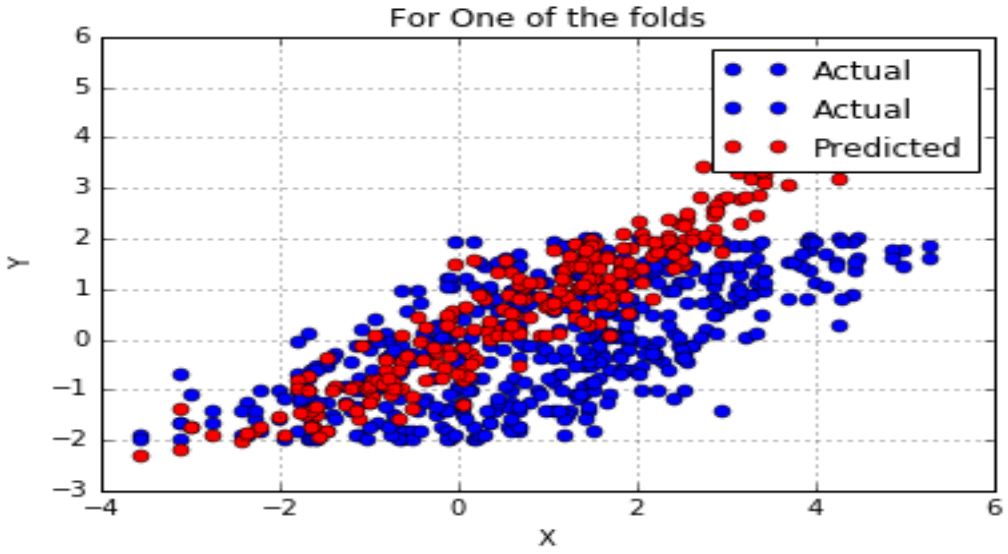
2.3 Testing on different polynomial model:

Polynomial regression using formula:

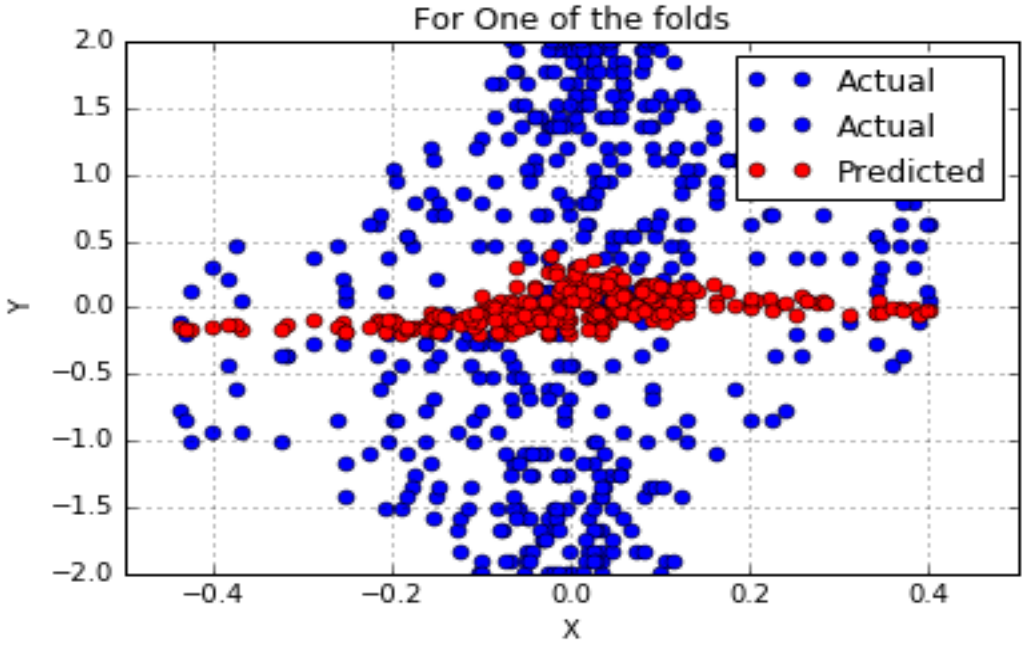
Y_hat= theta0 + theta1*x + theta2 * x^2+ ……..+ theta_n*x_n^2

Degree=2

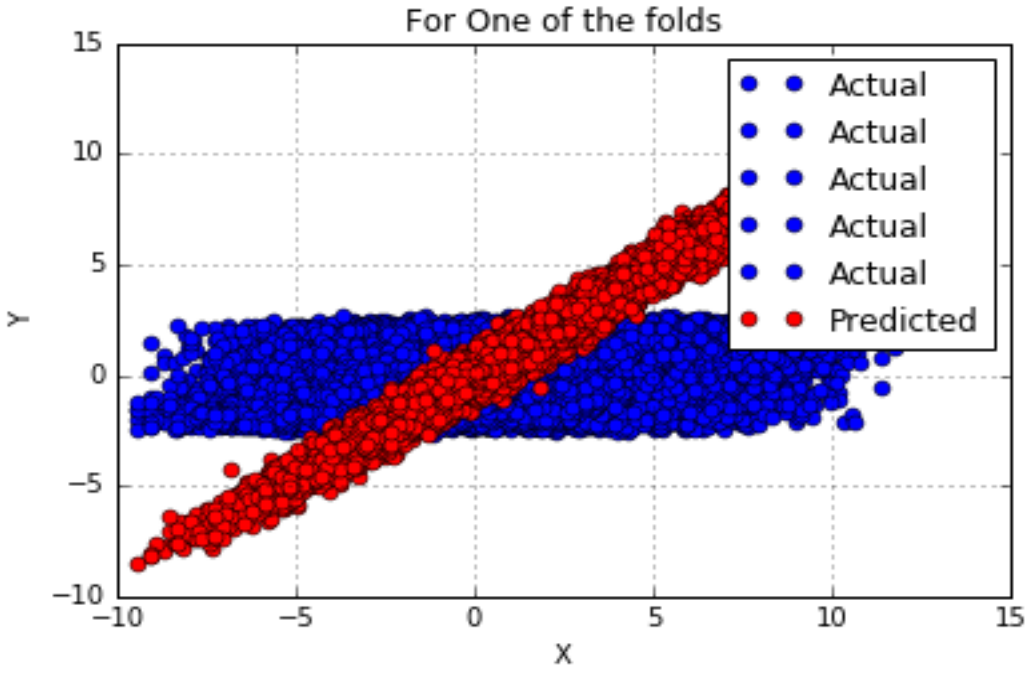
Plots after applying model on test data:
For mvar-set1.dat



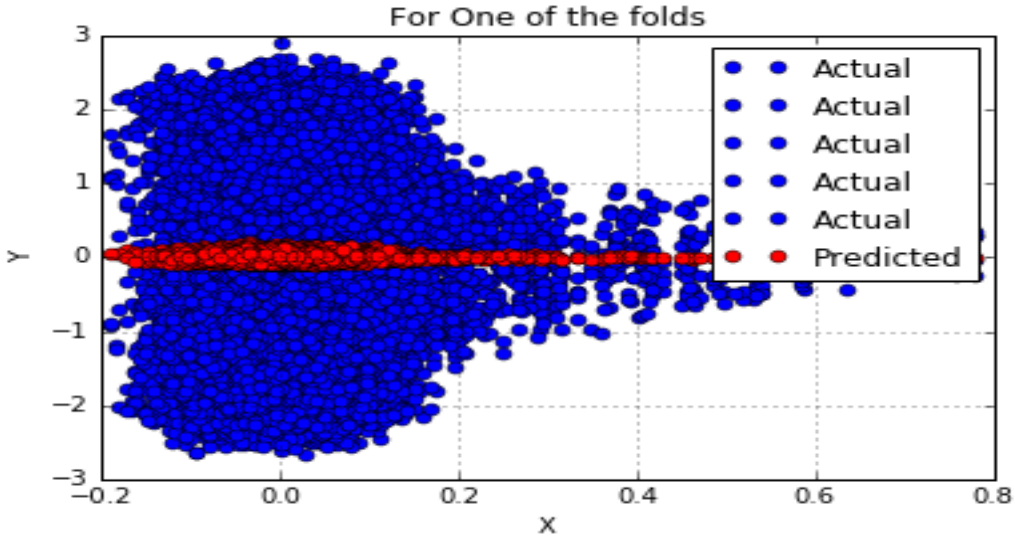
For mvar-set2.dat



For mvar-set3.dat



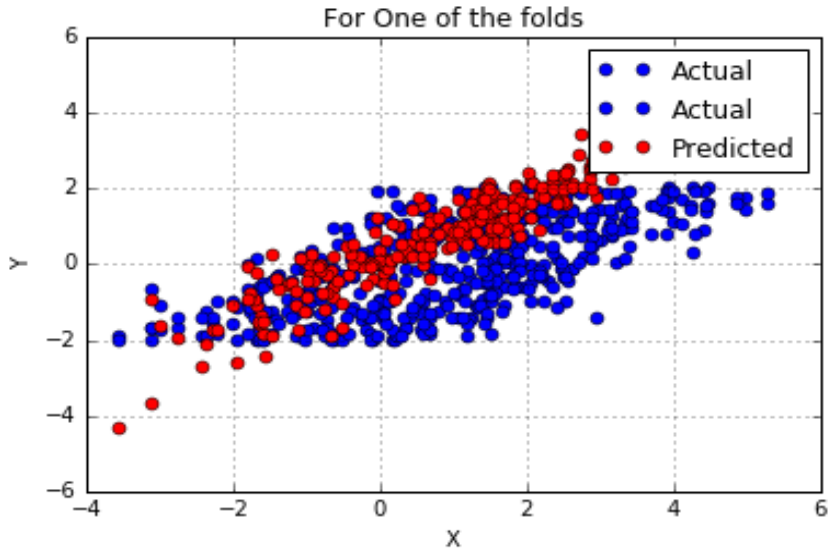
For mvar-set4.dat



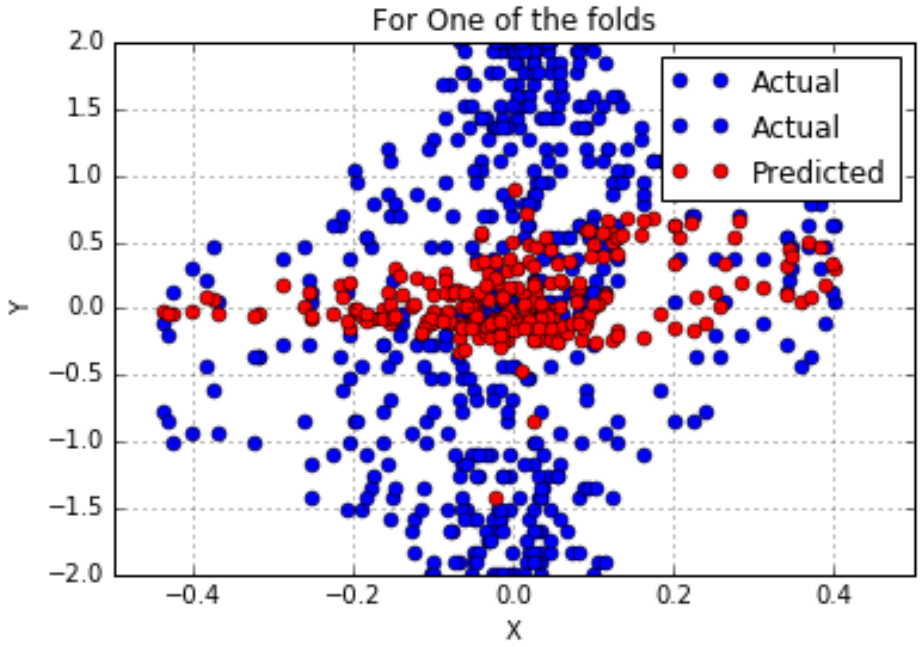
Degree=4

Plots after applying model on test data:

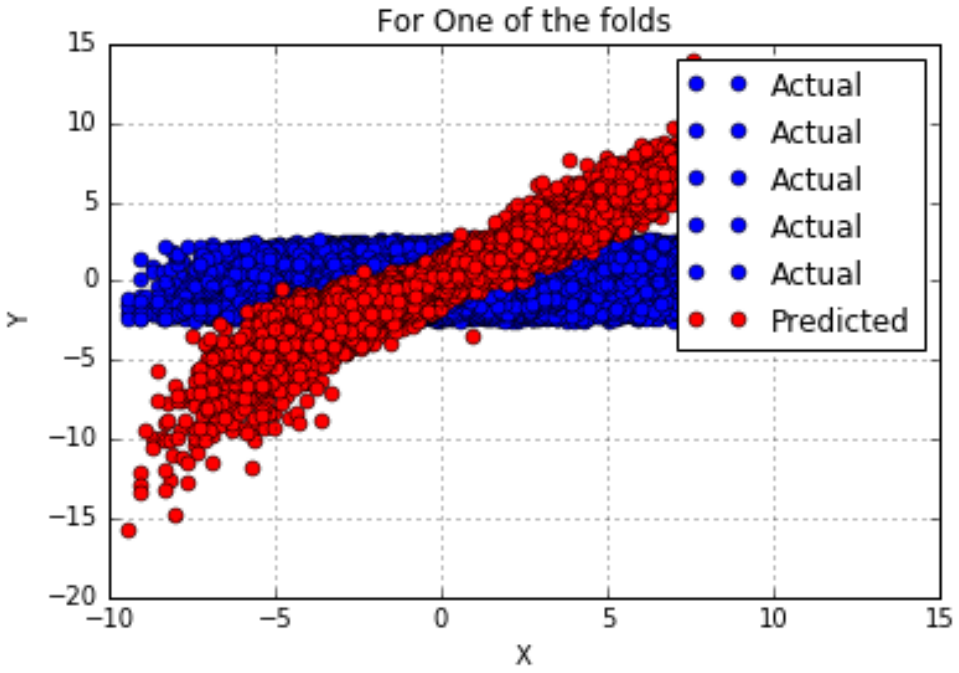
For mvar-set1.dat



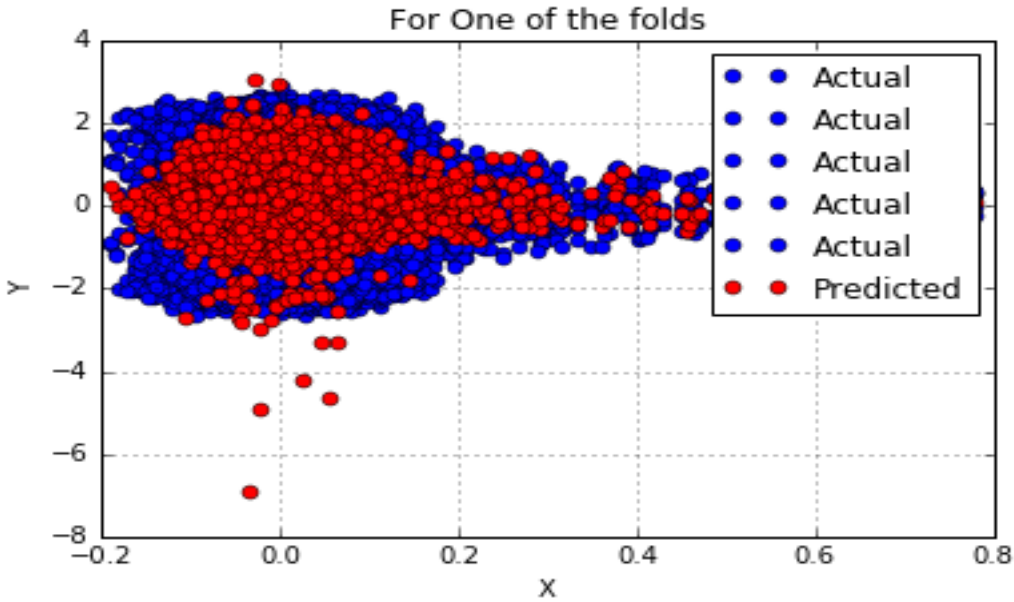
For mvar-set2.dat



For mvar-set3.dat



For mvar-set4.dat



Errors on test sets with **degree 2**

1 mvar-set1.dat

```
MSE with defined algorithm: 0.259616633312 (Explicit)
MSE With Readymade Function:  0.259149055236
MTE with defined algorithm:  0.258184561507
MTestE with defined algorithm:  0.252187065009
MSE with gradient Descent: 0.313139158337 (Iterative solution)
```

2 mvar-set2.dat

```
MSE with defined algorithm: 0.0200375957335 (Explicit)
MSE With Readymade Function:  0.0199567817456
MTE with defined algorithm:  0.0199027676401
MTestE with defined algorithm:  0.0193547757543
MSE with gradient Descent: 0.0396302130736 (Iterative solution)
```

3 mvar-set3.dat

```
MSE with defined algorithm: 0.25083633472 (Explicit)
MSE With Readymade Function:  0.250781409474
MTE with defined algorithm:  0.250704832521
MTestE with defined algorithm:  0.250174806566
MSE with gradient Descent: 0.312546044998 (Iterative solution)
```

4 mvar-set4.dat

```
MSE with defined algorithm: 0.00388831077649 (Explicit)
MSE With Readymade Function:  0.00418950268543
MTE with defined algorithm:  0.00388690298932
MTestE with defined algorithm:  0.0038812760517
MSE with gradient Descent: 0.0205443615418 (Iterative solution)
```

Errors on test sets with **degree 4**

5. mvar-set1.dat

```
MSE with defined algorithm: 0.260305477104 (Explicit)
MSE With Readymade Function:  0.259149055236
MTE with defined algorithm:  0.256294582906
MTestE with defined algorithm:  0.239732765129
MSE with gradient Descent: 0.43592806027 (Iterative solution)
```

6. mvar-set2.dat

```
MSE with defined algorithm: 0.0104392679204 (Explicit)
MSE With Readymade Function:  0.0199567817456
MTE with defined algorithm:  0.0102871597183
MTestE with defined algorithm:  0.00963201044629
MSE with gradient Descent: 0.0862672289189 (Iterative solution)
```

7. mvar-set3.dat

```
MSE with defined algorithm: 0.251115458921 (Explicit)
MSE With Readymade Function:  0.250781409474
MTE with defined algorithm:  0.25042640807
MTestE with defined algorithm:  0.24765029002
MSE with gradient Descent: 0.822261917279 (Iterative solution)
```

8. mvar-set4.dat

```
MSE with defined algorithm: 0.00346825417012 (Explicit)
MSE With Readymade Function:  0.00418950268543
MTE with defined algorithm:  0.00345954239452
MTestE with defined algorithm:  0.00342503024142
MSE with gradient Descent: 0.314932840745 (Iterative solution)
```

Conclusion:

For dataset1 and dataset2, mse with degree 4 is more than degree 2. We can say that model fits better with degree 2 for these datasets.

For dataset3 and dataset4, mse with all degrees is almost equal. We can say that model fits good for all degrees on these datasets.

For all four datasets, with all degrees 1, 2 and 4 mse with iterative solution is more than the explicit solution. Hence, we can say that explicit solution is better than iterative solution.