Pattern Recognition and Machine Learning

Assignment - 01

July 21, 2025

Name : Rohan Baghel
Student ID: 202116011

About

In this,we will study Polynomial regression and implement it using Matlab on random X values and f(x) = sin(2*Pi*X).

In simple linear regression algorithm only works when the relationship between the data is linear But suppose if we have non-linear data then Linear regression will not capable to draw a best-fit line and It fails in such conditions. To over come this we use Polynomial regression Model

0.0.1 Polynomial regression Model

Polynomial regression is a form of Linear regression where only due to the Non-linear relationship between dependent and independent variables we add some polynomial terms to linear regression to convert it into Polynomial Regression.

File1

In this file we are generating X vector with random points and adding noise for the function f(x) = Sin(2*pi*X) with E N(0,0.25).
Taking

- Training points = 20
- Testing points = 50

And Estimate Least square polynomial regression mode for M =1,2,3,9
and finding Coefficient for each M
computing the RMSE for polynomial regression models for order M =1,2,3 and 9 .

```
clear
clc
%-------------------------------------------------------------------------%
%1
disp("Q1._")

X1 = linspace(0,1,20);
disp("X1 =")
disp(X1)
disp("...................")


%-------------------------------------------------------------------------%
%2
disp("Q2_")
len = length(X1);
Y1=zeros(1,len);

for i = 1:len
    C = (0.25-0).*rand(1,20) + 0;
    mean = sum(C,2)/2;
    SD = zeros(1, len);
    for j = length(C)
            SD(j) = (C(j)-mean)^2;
    end
    SD = (sum(SD,2)/(len-1))^(1/2);

    Y1(i) = sin(2*pi*X1(i)) + SD;
end
disp('Y1 =')
disp(Y1)

disp('...................')

%-------------------------------------------------------------------------%
```

```
%3rd
disp("Q3_")
X2 = (1-0).*rand(1,50) + 0;
X2 = sort(X2);
len1 = length(X2);
Y2=zeros(1,len1);

for i = 1:len1
    C = (0.25-0).*rand(1,20) + 0;
    mean = sum(C,2)/2;
    SD = zeros(1, len);
    for j = length(C)
            SD(j) = (C(j)-mean)^2;
    end
    SD = (sum(SD,2)/(len-1))^(1/2);

    Y2(i) = sin(2*pi*X2(i)) + SD;
end
disp('X2 =')
disp(X2)
disp('Y2 =')
disp(Y2)

disp("................")

%-------------------------------------------------------------------------%
%Q4 and Q5
fprintf("Q4, Q5. Estimate Least square polynomial regression mode for M =1,2,3,9...
    \n and Coffecient \n\n")

% We know eqution is AX = B
% Here
disp("For m = 1")
m = 1;
[coef1,Yn1,MES1] = matrix(m,X1,Y1,len);
disp("coefficient 1 = ")
disp(coef1)
disp("Yn 1 = ")
disp(Yn1)

disp("For m = 2")
m = 2;
[coef2,Yn2,MES2] = matrix(m,X1,Y1,len);
disp("coefficient 2 = ")
disp(coef2)
disp("Yn 2 = ")
disp(Yn2)

disp("For m = 3")
m = 3;
[coef3,Yn3,MES3] = matrix(m,X1,Y1,len);
disp("coefficient 3 = ")
disp(coef3)
disp("Yn 3 = ")
disp(Yn3)

disp("For m = 9")
m = 9;
[coef9,Yn9,MES9] = matrix(m,X1,Y1,len);
```

```
disp("coefficient 9 = ")
disp(coef9)

disp("Yn 9 = ")
disp(Yn9)

disp("___________________________________")
%-------------------------------------------------------------------------%
%Q6_
disp("Q6_")
% ^Y = b_o + b_1*X

fprintf("for m = 1 \n RMES = \n %d \n", MES1 )

fprintf("for m = 2 \n RMES = \n %d \n", MES2 )

fprintf("for m = 3 \n RMES = \n %d \n", MES3 )

fprintf("for m = 9 \n RMES = \n %d \n", MES9 )

RMES1 = [MES9,MES2,MES3,MES9];

save('Q3_','X2','Y2','RMES1')

%.........................................................................%
%Q_7

y = sin(2*pi*X1);

ax1 = nexttile;
plot(X1,y,X1,Y1,'o',X1,Yn1,'r'), legend('Sin(2 Pi X)','sin(2 Pi X) +  ¬N(0,0.25...
    )','y  ');
title(ax1,'M= 1')

ax2 = nexttile;
plot(X1,y,X1,Y1,'o',X1,Yn2,'r'), legend('Sin(2 Pi X)','sin(2 Pi X) +  ¬N(0,0.25...
    )','y  ');
title(ax2,'M= 2')

ax3 = nexttile;
plot(X1,y,X1,Y1,'o',X1,Yn3,'r'), legend('Sin(2    X)','sin(2 Pi X) +  ¬N(0,0.25...
    )','y  ');
title(ax3,'M= 3')

ax4 = nexttile;
plot(X1,y,X1,Y1,'o',X1,Yn9,'r'), legend('Sin(2 Pi X)','sin(2 Pi X) +  ¬N(0,0.25...
    )','y  ');
title(ax4,'M= 9')
```

File 2

Name of file :- **matrix.m**
This file contain **function** which calculates the matrix of Least square polynomial matrix and compute the coefficient for M = 1,2,3 and 9.
This function also calculates the root Mean Square Sequence for the above values.

```

function [coef,Yn,MES] = matrix(m,X1,Y1,len)
    mat = zeros(m+1);
        for i = 1:(m+1)
            for j = 1:(m+1)
                if i== 1 && j == 1
                   mat(i,j) = len;
                else
                    mult = ones(1,len);
                    for k = 1:(i+j-2)
                          mult = mult.*X1;
                          mat(i,j) = sum(mult,2);
                    end
                end
            end
        end
        mat2 = zeros(m+1,1);

        for i = 1:(m+1)
            if i == 1
                mat2(i) = sum(Y1,2);
            else
                mult = ones(1,len);
                    for j = 1:(i-1)
                          mult = mult.*X1;
                    end
                    mult = mult.*Y1;
                    mat2(i) = sum(mult,2);
            end
        end

        coef = mat^(-1)*mat2;
        yn = zeros(1,len);
        for i=1:len
            a = 0;
            for j = 1:m+1
                b = X1(i)^(j-1);
                a = a+ coef(j)*b;
            end
            yn(i) = a;
        end
        Yn = yn;
        n  = Y1-yn;
     MES = n*transpose(n)/len;
end
```

0.0.2 Output of training set:

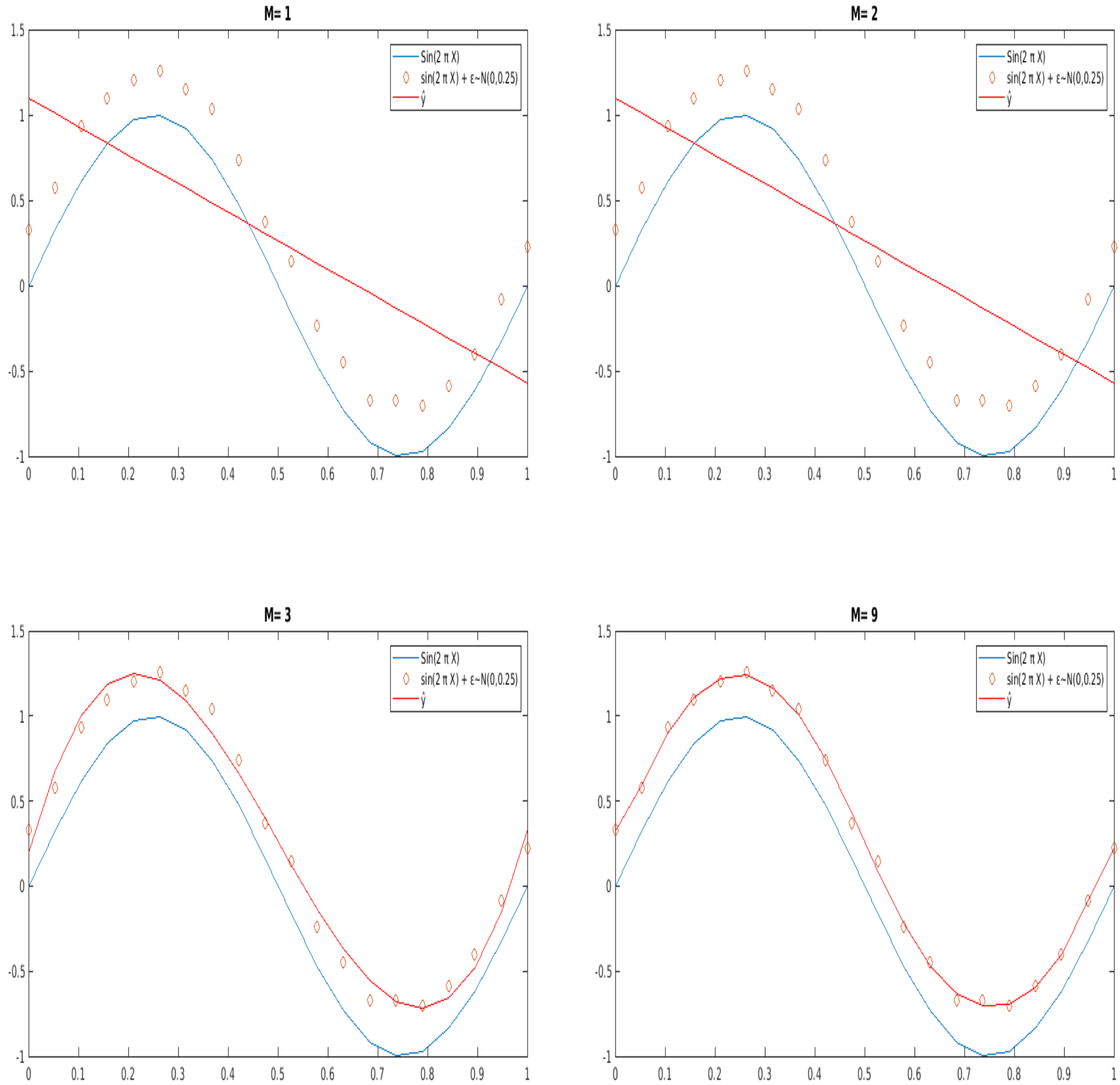

Figure 1: for Training set

File 3

This file contains Testing set with X as random numbers with 50 points and calculates the Least square polynomial and RMSE for M = 1,2,3 and 9.

```

%Q8_

load('Q3_.mat')

y = sin(2*pi*X2);
len = length(X2);

m = 1;
[coef1,Yn1,MES1] = matrix(m,X2,Y2,len);
m = 2;
[coef2,Yn2,MES2] = matrix(m,X2,Y2,len);
m = 3;
[coef3,Yn3,MES3] = matrix(m,X2,Y2,len);
m = 9;
[coef9,Yn9,MES9] = matrix(m,X2,Y2,len);

fprintf("for m = 1 \n RMES = \n %d \n", MES1 )
fprintf("for m = 2 \n RMES = \n %d \n", MES2 )
fprintf("for m = 3 \n RMES = \n %d \n", MES3 )
fprintf("for m = 9 \n RMES = \n %d \n", MES9 )

M = [1,2,3,9];
RMES2 = [MES1,MES2,MES3,MES9];

ax1 = nexttile;
plot(X2,y,X2,Y2,'o',X2,Yn1), legend('Sin(2 Pi X)','sin(2 Pi X) +  ¬N(0,0.25)','...
    y ');
title(ax1,'M= 1')

ax2 = nexttile;
plot(X2,y,X2,Y2,'o',X2,Yn2), legend('Sin(2 Pi X)','sin(2 Pi X) +  ¬N(0,0.25)','...
    y ');
title(ax2,'M= 2')

ax3 = nexttile;
plot(X2,y,X2,Y2,'o',X2,Yn3), legend('Sin(2 Pi X)','sin(2 Pi X) +  ¬N(0,0.25)','...
    y ');
title(ax3,'M= 3')

ax4 = nexttile;
plot(X2,y,X2,Y2,'o',X2,Yn9), legend('Sin(2 Pi X)','sin(2 Pi X) +  ¬N(0,0.25)','...
    y ');
title(ax4,'M= 9')

ax4 = nexttile;
plot(M,RMES1,M,RMES2), legend('RMSE for training', 'RMSE for Testing');
title(ax4,'M= 9')
```

0.0.3 Output of testing set

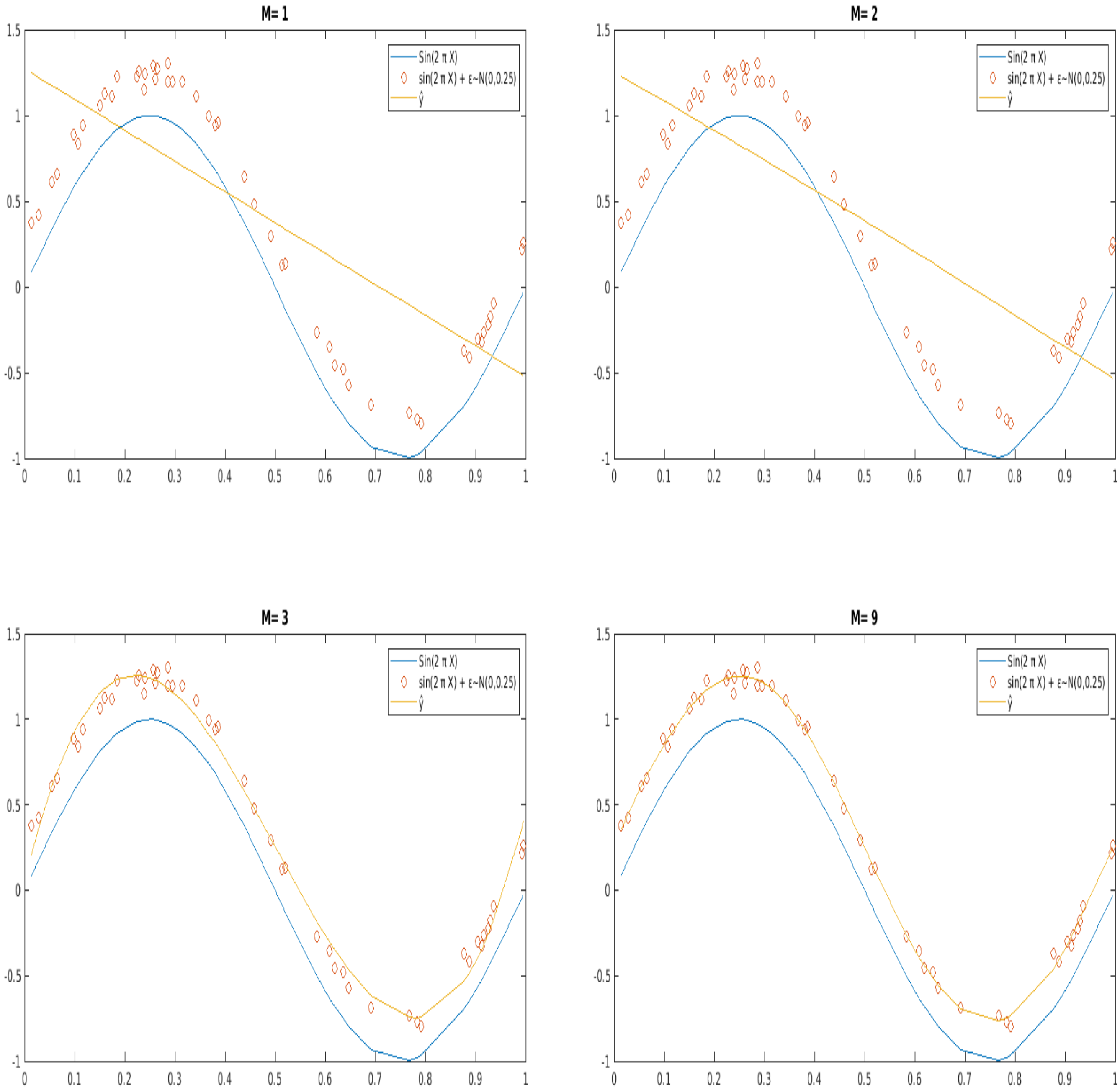

Figure 2: for Training set

Output of RMSE

For both testing and training points

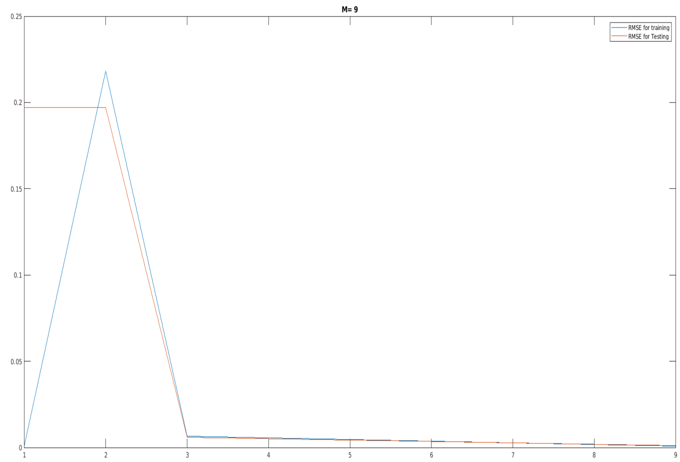

Figure 3: for Training set

Observation

We take,
Order of polynomial model m = 1,2,3 and 9

for order m = 1,
we observe that a linear line is formed in graph which will try to split the generated points

for order m = 2,
we observe that it form a quadratic line which will be little bent in shape trying to join the random points

for order m = 3,
we observe that forms a wave type graph which will try to connect the points but all the points are not connected, as some error will be present.

for order m = 9,
we observe that it will connect to all the random points, which will have more bends in its function

We can say that when we increase the order of polynomial model more bends will be formed.
More bands reduces the chance the function will be monotone.

0.1 statistical reason

When we look after the RMSE(Root mean Square error) values
For Training set with 20 pints when we increase the order of polynomial then RMSE value reduces
for order m = 3 RMSE value is less but for m = 9 RMSE value gradually increases.

For Testing set with 50 pints when we increase the order of polynomial then RMSE value reduces
in this for m = 9, RMSE value is less, as shows in the RMSE graph.

We can say that

- Small amount of data will restrict the maximum order of polynomial
- More the number of data less RMS eror will occur with higher order of polynomial
- Given n observations with no replicates, a model of order n-1 will fit perfectly, but would severely over fit the model and cause poor predictive accuracy.