CS584 Assignment 1 Report

Weilun Zhao; A20329942

Department of Computer Science

Illinois Institute of Technology

February 16, 2016

• Abstract

In this assignment, there were two parts of regression problems implemented by techniques for parametric regression with Python. In the first single regression part, data were plot and fitted to linear regression and polynomial models. To evaluate the result, using train data sets got the parameters of regression formula, then test data sets were fitted into the hypothesis formula to get difference with the true value. In the second multivariate regression part, loading multiple feature data sets, mapping them to higher dimensional feature space, and testing the result with different data sets.

• Single variable regression

a). Load data sets and plot the data

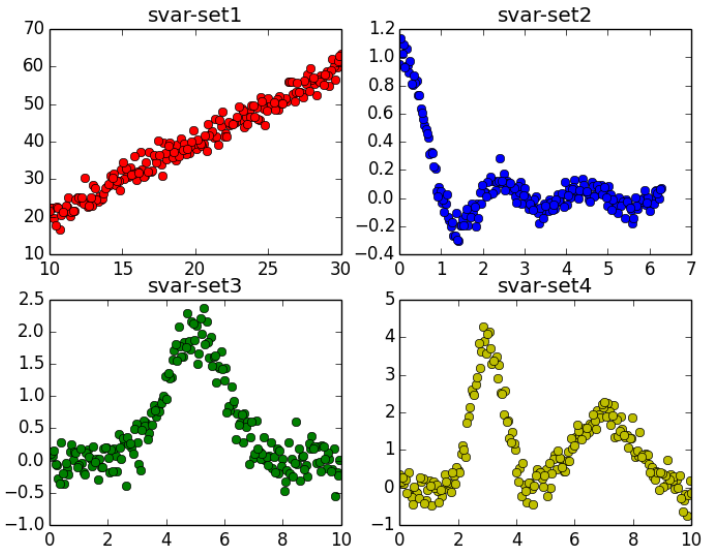


Figure 1

Analysis:

Data Set	Expect Degree	Analysis Idea
svar-set1	1	The data points is similar with a line
Svar-set2	3 or more	Data points neither form line or bow curve
Svar-set3	2	Data points look like a bow curve
Svar-set4	3 or more	Data points neither form a line or bow curve

b) Plot train set data and Calculate MSE

- Test set data plot

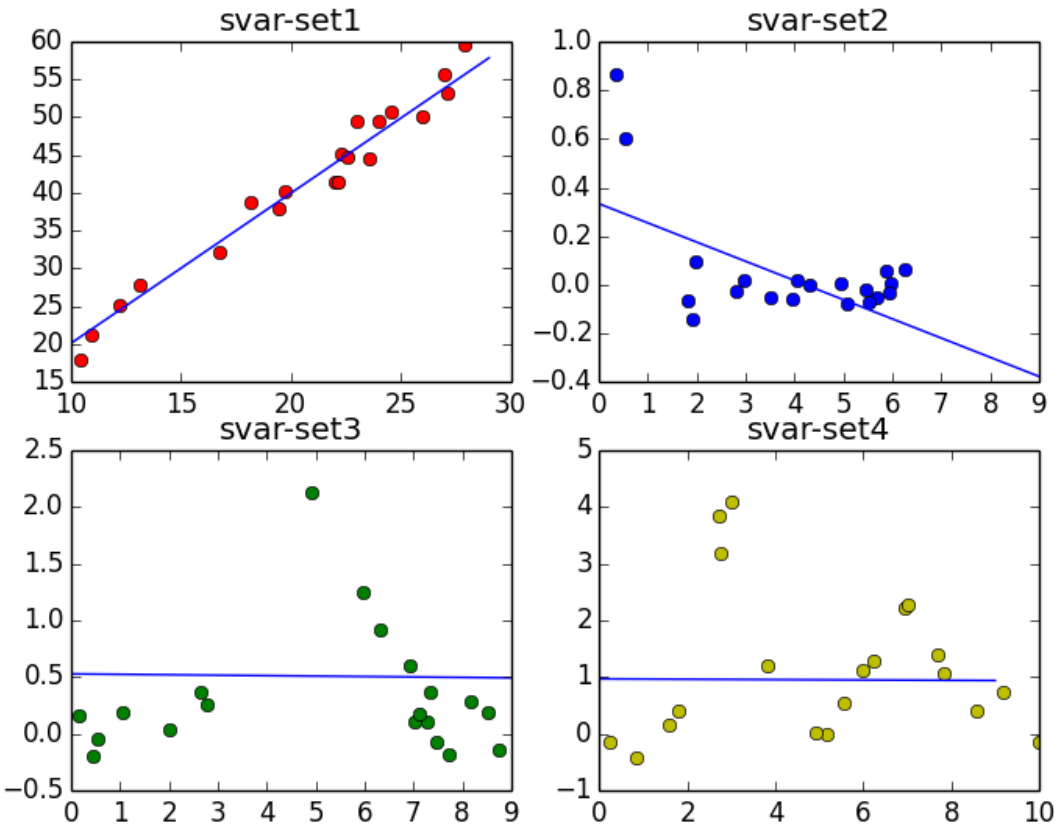


Figure 2

- Train and testing set error in the linear model

The general formula of linear regression: y = a0 * x + a1

The train data size is 180; the test data size is 20

Data Set	A0	A1	MSE in train set	MSE in test set
Svar-set1	1.97734961	0.42373779	4.23343919	4.2521169
Svar-set2	-0.07891878	0.33283915	0.06196014	0.03825547
Svar-set3	-0.0038674	0.52697054	0.51830252	0.32597774
Svar-set4	-0.00315079	0.96590264	1.14315024	1.72445474

c) Compare my results with the results from ready made Python function

My function theory:

The hypothesis formula is :

$$\hat{y}_i = h_\theta(x_i) = \theta_0 + \theta_1 * x_i$$

The residual sum of squares:

$$\mathcal{J} = \sum_{i=1}^{m} (y_i - (\theta_0 + \theta_1 * x_i))^2$$

To get the minimum RSS, derivative the RSS formula:

$$\nabla \mathcal{J} = 0$$

$$\frac{\partial \mathcal{J}}{\partial \theta_0} = \sum_{i=1}^{m} (y_i - (\theta_0 + \theta_1 * x_i)) * 1$$

$$\frac{\partial \mathcal{J}}{\partial \theta_1} = \sum_{i=1}^{m} (y_i - (\theta_0 + \theta_1 * x_i)) * x_i$$

Expand the formula above:

$$m\theta_0 + \sum_{i=1}^{m} x_i * \theta_1 - \sum_{i=1}^{m} y_i = 0$$

$$\sum_{i=1}^{m} x_i * \theta_0 + \sum_{i=1}^{m} (x_i)^2 * \theta_1 - \sum_{i=1}^{m} x_i * y_i = 0$$

Basing on the svar-set data, I can get the $\sum_{i=1}^{m} x_i$, $\sum_{i=1}^{m} y_i$, $\sum_{i=1}^{m} (x_i)^2$, and $\sum_{i=1}^{m} x_i * y_i$. Then solve the Expand formula of partial derivative of RSS and get coefficient of the linear regression.

The coefficient of the linear regression:

Data set	Coefficient
Svar-set1	[1.97734961 0.42373779]
Svar-set2	[-0.07891878 0.33283915]
Svar-set3	[-0.0038674 0.52697054]
Svar-set4	[-0.00315079 0.96590264]

*comment: in the coefficient array, the coefficient[0] is θ_1; coefficient[1] is θ_0

The python function:

```
import statsmodels.api as sm

def data_Anaylsis(xdata, ydata):
        x = sm.add_constant(xdata)
        est = sm.OLS(ydata, x).fit()
        print est.summary()
```

Using the summary to get coefficient for the linear regression formula with different data sets.

The coefficient result of my theory is same ready made python function.

d) Test different polynomial models

In this part, the data sets are fitted into different degree models and test the MSE to evaluate the result of fitness

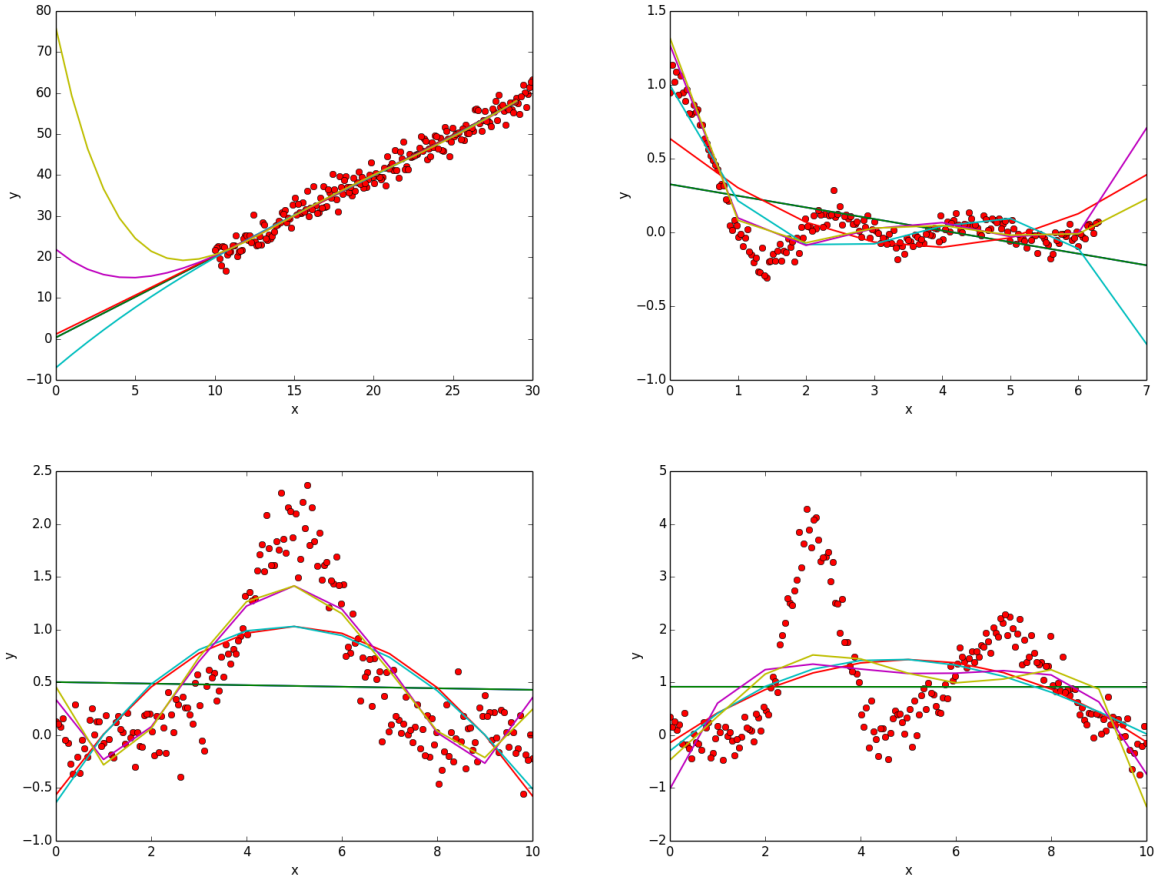
- Data plant

Data set	Degree	MSE of train set data	MSE of test set data
Svar-set1	1	4.23343919	4.2521169
	2	4.2331114	4.24934548
	3	4.17233278	4.11927854
	4	4.10154024	4.53085946
	5	4.09383144	4.54046111
Svar-set2	1	0.06196014	0.03825547
	2	0.041162	0.01675475
	3	0.02126954	0.01268473
	4	0.0120708	0.00662936
	5	0.01162945	0.00647401
Svar-set3	1	0.51830252	0.32597774
	2	0.25512431	0.24631473
	3	0.25508276	0.24752208
	4	0.12957968	0.10322223
	5	0.12943639	0.10140347
Svar-set4	1	1.14315024	1.72445474
	2	0.87498088	1.41351243
	3	0.87024017	1.37029609
	4	0.80784537	1.24375014
	5	0.76140198	1.20543009

svar-set1: MSE of train and test set data l is least data in the 3 degree polynomial model
svar-set2: MSE of train and test set data l is least data in the 5 degree polynomial model
svar-set3: MSE of train and test set data l is least data in the 5 degree polynomial model

svar-set4: MSE of train and test set data l is least data in the 5 degree polynomial model

- The plot graph



comment: Green like is one degree; red is for two degree; cyan is for 3 degree; purple is for 4 degree; yellow is for 5 degree

e) Reduce the amount of training data and observe the effect of performance

The train data size is 150; the test data size is 50

- Data plant

Data set	Degree	MSE of train set data	MSE of test set data
Svar-set1	1	4.22038643	4.29856341
	2	4.21587319	4.34355935
	3	4.17893242	4.21425175
	4	4.14667685	4.17825535
	5	4.1386093	4.17958162
Svar-set2	1	0.06182678	0.05311341
	2	0.04134559	0.03135766

	3	0.0211801	0.01896487
	4	0.01179757	0.0108183
	5	0.01153065	0.00994813
Svar-set3	1	0.47699844	0.56679837
	2	0.23802506	0.29984929
	3	0.23721062	0.30493155
	4	0.12312189	0.14093139
	5	0.12145707	0.14646803
Svar-set4	1	1.0816912	1.57115126
	2	0.85272863	1.18469433
	3	0.84918374	1.16235166
	4	0.78452564	1.07263819
	5	0.75065473	0.99141371

- Observe

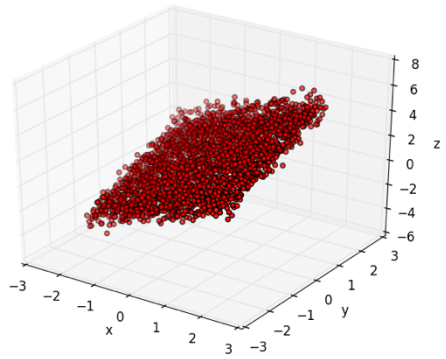
There is no apparent difference of MSE, if reduce the amount of train data. However, MSE of test set data increase in a small certain degree based on the observation of comparison with original MSE of data set

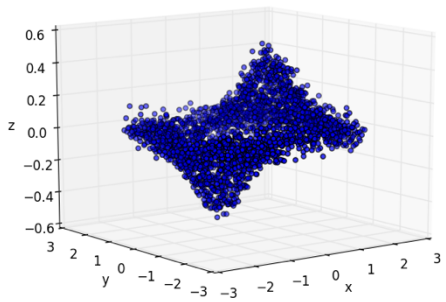
• Multivariate regression

a) Load the multiple feature data sets, and map into higher dimension

- Data load

The data of mvar-set1 and mvar-set2 can be plot into 3 dimension graph, but mvar-set3 and mvar-set4 have 6 columns data and cannot display in graph.





- Data linear regression

Mvar-set1 and mvar-set2 linear hypothesis formula:

$$y = a_0 + a_1 * x_0 + a_2 * x_1$$

Mvar-set3 and mvar-set4 linear hypothesis formula:

$$y = a_0 + a_1 * x_0 + a_2 * x_1 + a_3 * x_2 + a_4 * x_3 + a_5 * x_4$$

- Mapping data into higher dimension

Mvar-set1 and mvar-set2 higher dimensional hypothesis formula:

$$y = a_0 + a_1 * x_0 + a_2 * x_1 + a_3 * x_0^2 + a_4 * x_0 * x_1 + a_5 * x_1^2$$

Mvar-set3 and mvar-set4 higher dimensional hypothesis formula:

$$y = a_0 + a_1 * x_0 + a_2 * x_1 + a_3 * x_2 + a_4 * x_3 + a_5 * x_4 + a_6 * x_0^2 + a_7 * x_1^2 + a_8 * x_2^2 + a_9 * x_3^2 + a_{10} * x_4^2$$

b) Perform linear regression in the higher dimensional space

- Linear regression coefficient and MSE

In the linear regression model of mvar-set1 and mvar-set2:

$$y = a_0 + a_1 * x_0 + a_2 * x_1$$

Data Set	A0	A1	A2	MSE
Mvar-set1	0.9959	0.9975	0.9905	0.258702892151
Mvar-set2	0.0009	0.0646	-0.0006	0.0199118768693

In the linear regression model of mvar-set3 and mvar-set4:

$$y = a_0 + a_1 * x_0 + a_2 * x_1 + a_3 * x_2 + a_4 * x_3 + a_5 * x_4$$

Data Set	A0	A1	A2	A3	A4	A5	MSE
Mvar-set3	0.9989	0.9979	1.0008	0.9988	-0.0012	1.9998	0.250743148719
Mvar-set4	0.0101	4.668e-05	0.0001	3.952e-05	0.0002	-1.837e-06	0.00418917439966

- Higher regression result with MSE

Mapping data into higher dimensional follow the formula below:

- Mvar-sert1, Mvar-set2: $y = a_0 + a_1 * x_0 + a_2 * x_1 + a_3 * x_0^2 + a_4 * x_0 * x_1 + a_5 * x_1^2$
- Mvar-sert3, Mvar-set4:$y = a_0 + a_1 * x_0 + a_2 * x_1 + a_3 * x_2 + a_4 * x_3 + a_5 * x_4 + a_6 * x_0^2 + a_7 * x_1^2 + a_8 * x_2^2 + a_9 * x_3^2 + a_{10} * x_4^2$

Data Set	Amount of train data	Amount of test data	Train set MSE	Test set MSE
Mvar-set1	2450	50	0.25800120516	0.281907095552
Mvar-set2	2450	50	0.0200158719878	0.0147231540775

Mvar-set3	99900	100	14.4838115942	12.3854983595
Mvar-set4	99900	100	0.00395142847316	0.00319633351941

c) Solve the regression problem using explicit solution and iterative solution

- Explicit solution

The Explicit Solution Theory:

The hypothesis formula is :

$$\hat{y}_i = h_\theta(x_i) = \theta^T Z_i$$

The residual sum of squares:

$$\mathcal{J} = \sum_{i=1}^{m} (\theta^T Z_i - y_i)^2$$

To get the minimum RSS, we need figure out the coefficient of parameter

$$\theta^* = \arg min\, \mathcal{J}(\theta)$$

$$\mathcal{J} = (\mathrm{Z}\theta - Y)^T (Z\theta - Y)$$

Partial derivative for the RSS to get the minimum value of RSS:

$$\nabla \mathcal{J}(\theta) = 0$$

$$2\mathrm{Z}^T(\mathrm{Z}\theta - Y) = 0$$
$$\mathrm{Z}^T\mathrm{Z}\theta = \mathrm{Z}^T Y$$
$$\theta = (\mathrm{Z}^T\mathrm{Z})^{-1}\mathrm{Z}^T Y$$

The Explicit Solution Result

Mvar-set1 and Mvar-set2 hypothesis formula:

$$y = a_0 * x_0 + a_1 * x_1$$

Data Set	A0	A1	MSE_train	MSE_test
Mvar-set1	0.99752589	0.99048461	1.2504733	1.25109853
Mvar-set2	0.06458313	-0.00060792	0.01991267	0.01995297

Mvar-set3 and mvar-set4 hypothesis formula:

$$y = a_0 * x_0 + a_1 * x_1 + a_2 * x_2 + a_3 * x_3 + a_4 * x_4$$

Data Set	A0	A1	A2	A3	A4	MES_train	MSE_test
Mvarset3	9.98888180e-01	1.00066063e+00	9.98744225e-01	-1.59114109e-03	1.99987699e+00	1.24861541	1.24851603

Mvar-set4	5.29559018e-05	1.10912497e-04	3.81903871e-05	2.38354477e-04	-3.53383038e-06	0.00429159	0.00429166

- Iterative solution

Iterative solution theory: (gradient descent)

Using iterative equation to decrease the MSE at each step to get the approximate result which is very close the actual result.

The residual sum of squares:

$$\mathcal{J} = \sum_{i=1}^{m} (\theta^T Z_i - y_i)^2 = (\mathrm{Z}\theta - Y)^T (Z\theta - Y)$$

$$\nabla \mathcal{J}(\theta) = Z^T (Z\theta - Y)$$

In the gradient descent:

$$\theta_i = \theta_{i-1} - \eta \left(Z^T (Z\theta - Y) \right)$$; η is the length of step

Iterative solution result

Mvar-set1 and mvar-set2 linear hypothesis formula:

$$y = a_0 + a_1 * x_0 + a_2 * x_1$$

Data Set	Num_iteration	learning rate	A0	A1	A2	MSE
Mvar-set1	1000	0.001	0.86136909	0.93560181	0.92899764	0.28736327
Mvar-set2	1000	0.000001	0.00198976	0.00276481	0.00274529	0.025232

Mvar-set3 and mvar-set4 linear hypothesis formula:

$$y = a_0 + a_1 * x_0 + a_2 * x_1 + a_3 * x_2 + a_4 * x_3 + a_5 * x_4$$

Data Set	Num Iteration	Learning rate	A0	A1	A2	A3	A4	A5	MSE
Mvar-set3	100	0.1	0.99909956	0.99955968	1.00165288	0.99814789	-0.00136831	0.66673759	3.21724785
Mvar set4	100	0.01	0.00877825	4.67960439e-05	0.00010606	3.78883215e-05	0.00023086	0.00022281	0.00419106

d) Gaussian kernel function

- Gaussian kernel Theory:

The Gaussian density formula:

$$K(X,Y) = \exp\left(-\frac{||x-y||^2}{2r^2}\right)$$

Matrix G is Gaussian kernel Gram matrix:

$$G = \begin{bmatrix} K(X_1,X_1) & \cdots & K(X_1,X_m) \\ \vdots & \ddots & \vdots \\ K(X_m,X_1) & \cdots & K(X_m,X_m) \end{bmatrix}$$

$$\alpha = G^{-1}Y$$

$$h_\alpha(X) = \sum_{i=1}^{m} \alpha K(X,X^i)$$

- Gaussian kernel result:

Data Set	Train set MSE	Test set MSE
Mvar-set1	-0.000592494783072	0.114285660424
Mvar-set2	0.000329163130249	-0.0163264909216
Mvar-set3	-1.19652527498e-05	-0.654729722302
Mvar-set4	-2.46044821317e-05	0.22851579077

Conclude

Test set MSE is increase with the data set degree and the train set MSE is extremely small than the model above. As a result, Gaussian kernel regression have a relative good result with less MSE in all the model. And The set data can be unpredictable, but it still fit the linear and polynomial regression formula got from the train set data.

Reference:

http://docs.scipy.org/doc/numpy-1.10.1/index.html
http://docs.scipy.org/doc/scipy-0.16.0/reference/generated/scipy.stats.norm.html
http://stackoverflow.com/questions/29731726/how-to-calculate-a-gaussian-kernel-matrix-efficiently-in-numpy