**Lab 6: Ridge/LASSO Regression**

**Problem statement:**

* Develop Ridge Regression model and try to tune it with varying alpha values. Plot SSE against each value of alpha. [Download suitable dataset with enough features, refer to access R-preloaded dataset]
* Develop LASSO Regression model and try to tune it with varying alpha values. Plot SSE against each value of alpha.
* Demonstrate the program SPARSITY property of LASSO Regression.

**Source Code and Output:**

"""

Author: Ashish Upadhyay

Branch: Computer Science and Engineering

Semester: 6th

Dr. SP Mukherjee International Institute of Information Technology, Naya Raipur

Subject: Machine Learning Lab 7

Task: Ridge/LASSO Regression Implementation

"""

#Importing libraries.

import numpy as np

import pandas as pd

import random

import matplotlib.pyplot as plt

%matplotlib inline

from matplotlib.pylab import rcParams

rcParams['figure.figsize'] = 12, 10

#Define input array with angles from 60deg to 300deg converted to radians

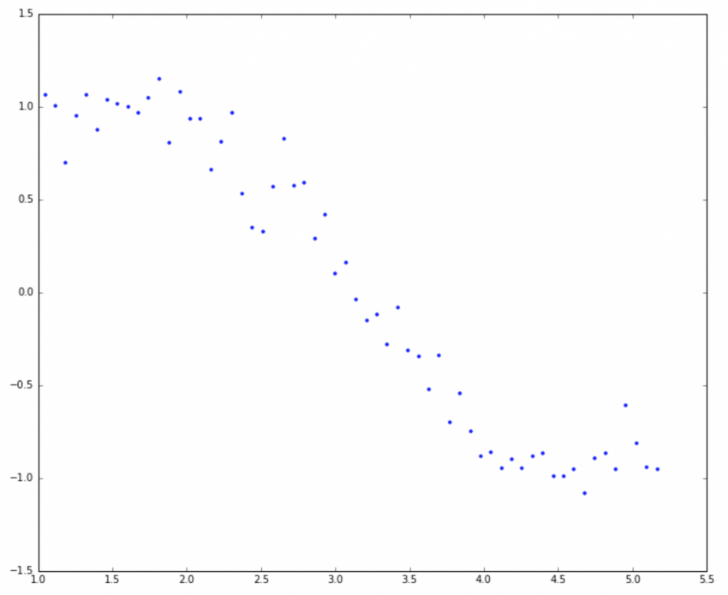
x = np.array([i\*np.pi/180 for i in range(60,300,4)])

np.random.seed(10) #Setting seed for reproducability

y = np.sin(x) + np.random.normal(0,0.15,len(x))

data = pd.DataFrame(np.column\_stack([x,y]),columns=['x','y'])

plt.plot(data['x'],data['y'],'.')

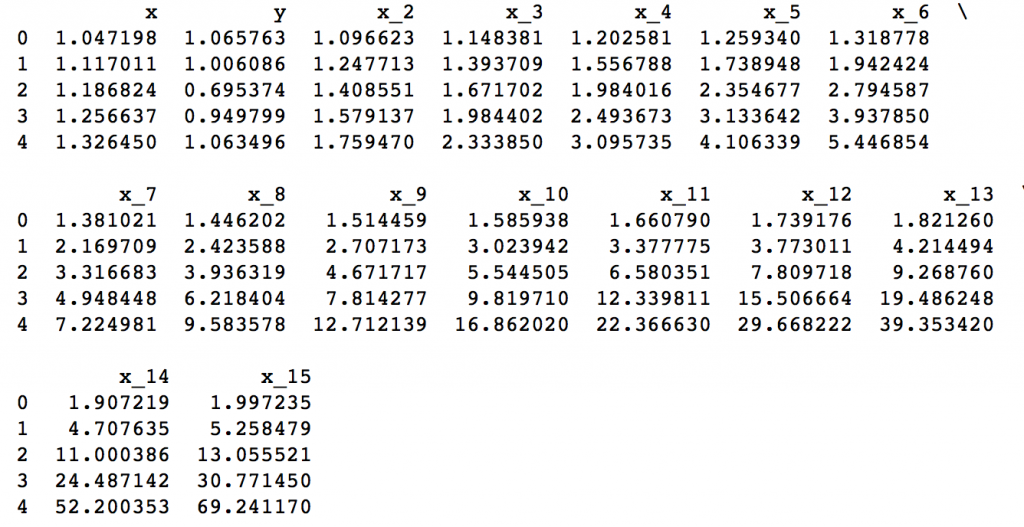


for i in range(2,16): #power of 1 is already there

colname = 'x\_%d'%i #new var will be x\_power

data[colname] = data['x']\*\*i

print data.head()



#Ridge Regression

from sklearn.linear\_model import Ridge

def ridge\_regression(data, predictors, alpha, models\_to\_plot={}):

#Fit the model

ridgereg = Ridge(alpha=alpha,normalize=True)

ridgereg.fit(data[predictors],data['y'])

y\_pred = ridgereg.predict(data[predictors])

#Check if a plot is to be made for the entered alpha

if alpha in models\_to\_plot:

plt.subplot(models\_to\_plot[alpha])

plt.tight\_layout()

plt.plot(data['x'],y\_pred)

plt.plot(data['x'],data['y'],'.')

plt.title('Plot for alpha: %.3g'%alpha)

#Return the result in pre-defined format

rss = sum((y\_pred-data['y'])\*\*2)

ret = [rss]

ret.extend([ridgereg.intercept\_])

ret.extend(ridgereg.coef\_)

return ret

#Initialize predictors to be set of 15 powers of x

predictors=['x']

predictors.extend(['x\_%d'%i for i in range(2,16)])

#Set the different values of alpha to be tested

alpha\_ridge = [1e-15, 1e-10, 1e-8, 1e-4, 1e-3,1e-2, 1, 5, 10, 20]

#Initialize the dataframe for storing coefficients.

col = ['rss','intercept'] + ['coef\_x\_%d'%i for i in range(1,16)]

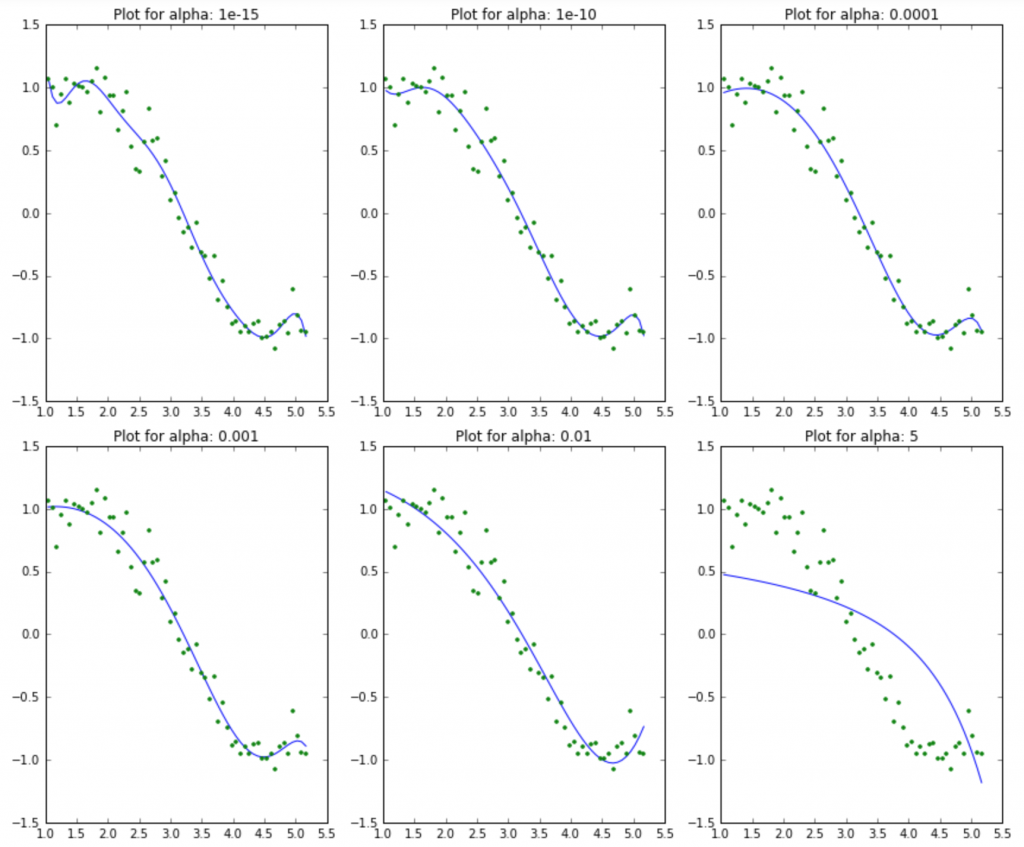
ind = ['alpha\_%.2g'%alpha\_ridge[i] for i in range(0,10)]

coef\_matrix\_ridge = pd.DataFrame(index=ind, columns=col)

models\_to\_plot = {1e-15:231, 1e-10:232, 1e-4:233, 1e-3:234, 1e-2:235, 5:236}

for i in range(10):

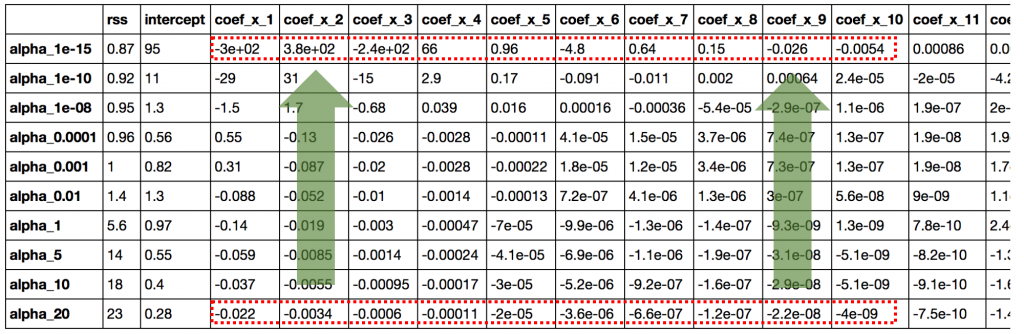
coef\_matrix\_ridge.iloc[i,] = ridge\_regression(data, predictors, alpha\_ridge[i], models\_to\_plot)



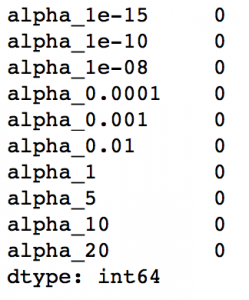
#Set the display format to be scientific for ease of analysis

pd.options.display.float\_format = '{:,.2g}'.format

coef\_matrix\_ridge



coef\_matrix\_ridge.apply(lambda x: sum(x.values==0),axis=1)



#LASSO Rigression

from sklearn.linear\_model import Lasso

def lasso\_regression(data, predictors, alpha, models\_to\_plot={}):

#Fit the model

lassoreg = Lasso(alpha=alpha,normalize=True, max\_iter=1e5)

lassoreg.fit(data[predictors],data['y'])

y\_pred = lassoreg.predict(data[predictors])

#Check if a plot is to be made for the entered alpha

if alpha in models\_to\_plot:

plt.subplot(models\_to\_plot[alpha])

plt.tight\_layout()

plt.plot(data['x'],y\_pred)

plt.plot(data['x'],data['y'],'.')

plt.title('Plot for alpha: %.3g'%alpha)

#Return the result in pre-defined format

rss = sum((y\_pred-data['y'])\*\*2)

ret = [rss]

ret.extend([lassoreg.intercept\_])

ret.extend(lassoreg.coef\_)

return ret

#Initialize predictors to all 15 powers of x

predictors=['x']

predictors.extend(['x\_%d'%i for i in range(2,16)])

#Define the alpha values to test

alpha\_lasso = [1e-15, 1e-10, 1e-8, 1e-5,1e-4, 1e-3,1e-2, 1, 5, 10]

#Initialize the dataframe to store coefficients

col = ['rss','intercept'] + ['coef\_x\_%d'%i for i in range(1,16)]

ind = ['alpha\_%.2g'%alpha\_lasso[i] for i in range(0,10)]

coef\_matrix\_lasso = pd.DataFrame(index=ind, columns=col)

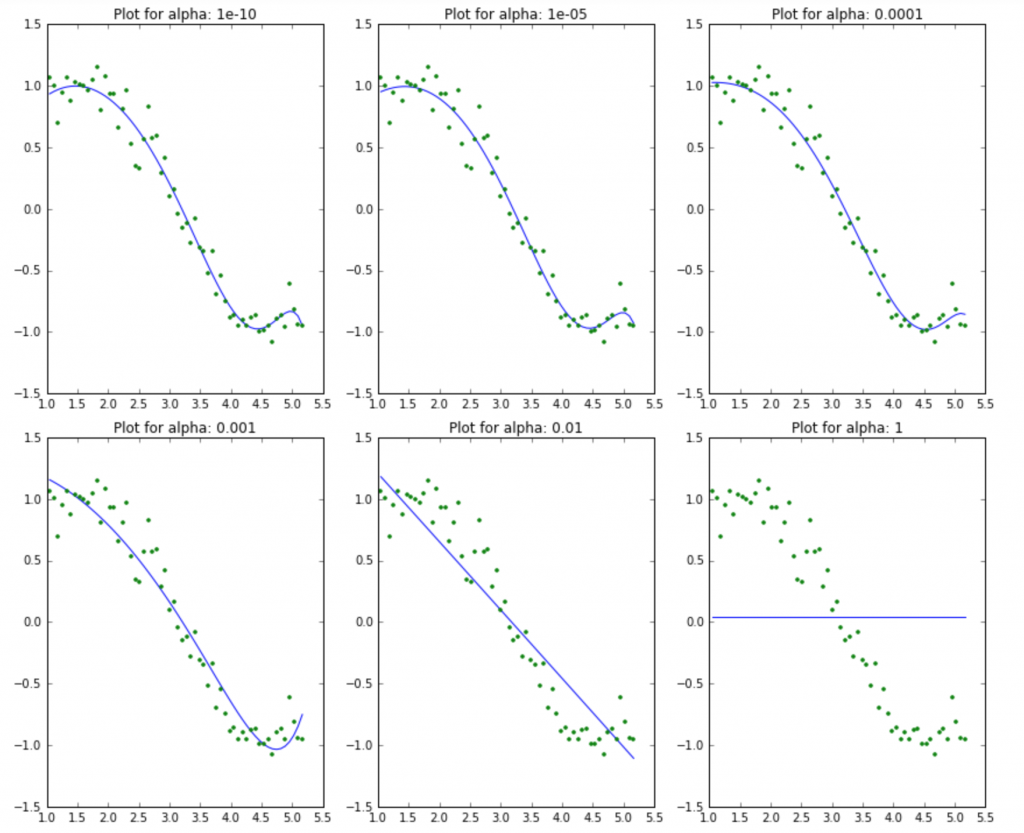
#Define the models to plot

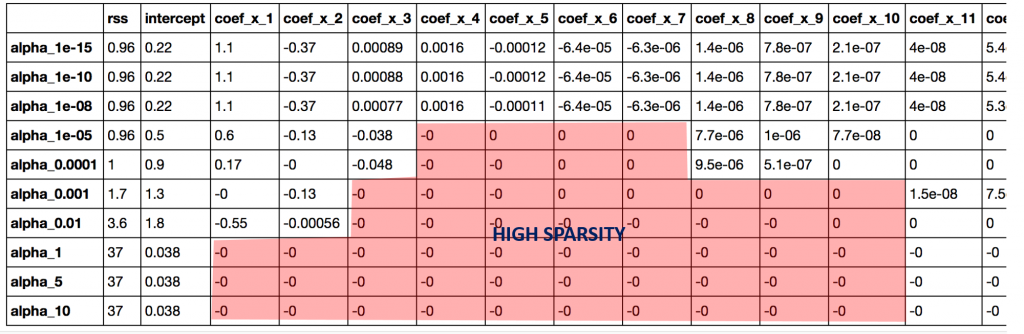
models\_to\_plot = {1e-10:231, 1e-5:232,1e-4:233, 1e-3:234, 1e-2:235, 1:236}

#Iterate over the 10 alpha values:

for i in range(10):

coef\_matrix\_lasso.iloc[i,] = lasso\_regression(data, predictors, alpha\_lasso[i], models\_to\_plot)





coef\_matrix\_lasso.apply(lambda x: sum(x.values==0),axis=1)

