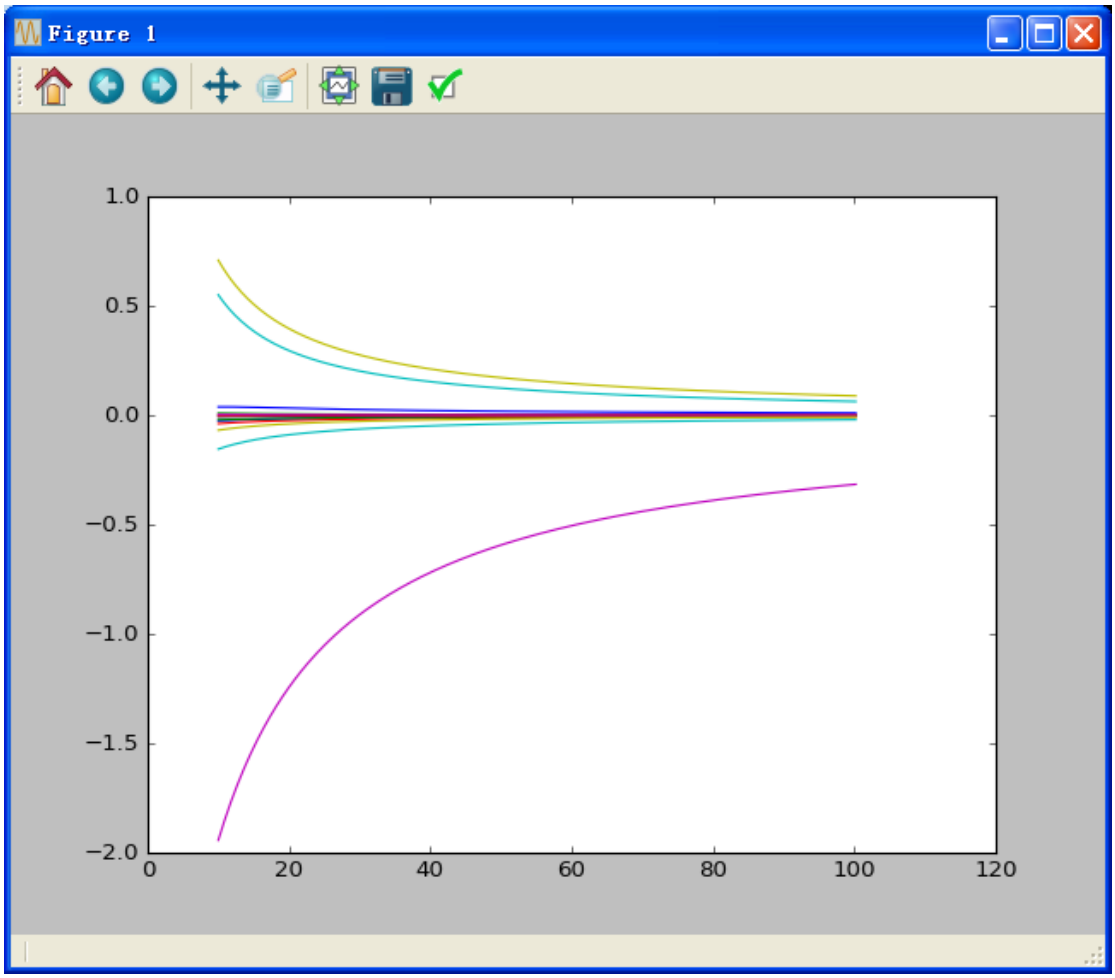
```
# 岭回归分析 boston 数据集

In [1]: import numpy as np

In [2]: import pandas as pd

In [3]: from sklearn.datasets import load_boston

In [4]: from sklearn.cross_validation import train_test_split

In [5]: from sklearn.linear_model import Ridge

In [6]: from sklearn.metrics import mean_squared_error

In [7]: import matplotlib.pyplot as plt

In [8]: from sklearn.preprocessing import PolynomialFeatures

In [52]: data = load_boston()

In [53]: X = data['data']

In [54]: y = data['target']

In [55]: X = X - np.mean(X,axis=0)   # 中心化

In [56]: X_train,X_test_all,y_train,y_test_all =
             train_test_split(X,y,test_size=0.3,random_state=9)

In [57]: X_dev,X_test,y_dev,y_test =
            train_test_split(X_test_all,y_test_all,test_size=0.3,
                             random_state=9)

In [58]: poly_features = PolynomialFeatures(interaction_only=True)

In [59]: poly_features.fit(X_train)

In [60]: X_train_poly = poly_features.transform(X_train)

In [61]: X_dev_poly = poly_features.transform(X_dev)

In [62]: X_test_poly = poly_features.transform(X_test)

In [63]: model = Ridge(normalize=True,alpha=0.015)
```

```
In [64]: model.fit(X_train_poly,y_train)

In [65]: y_predicted = model.predict(X_train_poly)

In [66]: mse = mean_squared_error(y_train,y_predicted)

In [67]: print "MSE = %0.2f" % mse
MSE = 6.85

In [68]: for i,coef in enumerate(model.coef_):
    ...:     print "Coefficient %d %0.3f" % (i+1,coef)
    ...:
Coefficient 1 0.000          # 几乎缩减为 0
Coefficient 2 -0.016
Coefficient 3 -0.003
Coefficient 4 0.006
Coefficient 5 4.285
......
Coefficient 90 -0.002
Coefficient 91 0.031
Coefficient 92 -0.001

In [69]: print "Intercept %0.3f" % (model.intercept_)
Intercept 21.027

In [70]: y_predicted = model.predict(X_dev_poly)

In [71]: print "MSE = %0.2f"
                % (mean_squared_error(y_dev,y_predicted))
MSE = 11.54

In [72]: y_predicted = model.predict(X_test_poly)

In [73]: print "MSE = %0.2f"
                % (mean_squared_error(y_test,y_predicted))
MSE = 9.46



# 演示模型对微小变化的敏感
In [56]: from sklearn.datasets import load_boston
```

```
In [57]: from sklearn.cross_validation import train_test_split

In [58]: from sklearn.linear_model import Ridge

In [59]: from sklearn.preprocessing import PolynomialFeatures

In [60]: data = load_boston()

In [61]: X = data['data']

In [62]: y = data['target']

In [63]: X = X - np.mean(X,axis=0)

In [64]: noise = np.random.normal(0,1,(X.shape))

In [69]: Xnoise = X + noise

In [70]: modelLR = LinearRegression()

In [71]: modelLR.fit(X,y)

In [72]: for i,coef in enumerate(modelLR.coef_):
   ...:     print "Coefficient %2d %-0.3f" % (i+1,coef)
   ...:

In []: print "Intercept %-0.3f" % (modelLR.intercept_)

In [73]: modelLR.fit(Xnoise,y)

In [74]: for i,coef in enumerate(modelLR.coef_):
   ...:     print "Coefficient %2d %-0.3f" % (i+1,coef)
   ...:
Coefficient  1 -0.107       -0.087
Coefficient  2 0.046        0.057
Coefficient  3 0.021        -0.071
Coefficient  4 2.689        -0.073
Coefficient  5 -17.796      0.001
Coefficient  6 3.805        0.984
Coefficient  7 0.001        0.020
Coefficient  8 -1.476       -0.877
Coefficient  9 0.306        0.304
Coefficient 10 -0.012       -0.016
Coefficient 11 -0.953       -0.722
```

```
Coefficient 12 0.009    0.008
Coefficient 13 -0.525       -0.727

Intercept 36.491            42.255


# 岭回归
In [80]: modelRidge = Ridge(normalize=True,alpha=0.015)

In [81]: modelRidge.fit(X,y)

In [82]: for i,coef in enumerate(modelRidge.coef_):
    ...:     print "Coefficient %2d %-0.3f" % (i+1,coef)
    ...:

In [83]: print "Intercept %-0.3f" % (modelRidge.intercept_)
Intercept 33.951

In [84]: modelRidge.fit(Xnoise,y)

In [85]: for i,coef in enumerate(modelRidge.coef_):
    ...:     print "Coefficient %2d %-0.3f" % (i+1,coef)
    ...:

# 加噪声前后的模型系数对比
Coefficient  1 -0.101    -0.083
Coefficient  2 0.042     0.053
Coefficient  3 -0.001    -0.088
Coefficient  4 2.772     -0.073
Coefficient  5 -16.139   -0.013
Coefficient  6 3.883     1.007
Coefficient  7 -0.001    0.017
Coefficient  8 -1.385    -0.855
Coefficient  9 0.254 0.256
Coefficient 10 -0.010    -0.013
Coefficient 11 -0.927    -0.718
Coefficient 12 0.009     0.008
Coefficient 13 -0.512    -0.709

Intercept 33.951          41.456
# 稳定性还不明显,是因为 alpha 太小

In [89]: coeffs = []
```

```
In [90]: for alpha in alpha_range:
    ...:     model = Ridge(normalize=True,alpha=alpha)
    ...:     model.fit(X,y)
    ...:     coeffs.append(model.coef_)
    ...:

In [95]: plt.plot(alpha_range,coeff)

In [97]: plt.show()
```



系数在 alpha=100 附近稳定

取 alpha = 50 再做一次 Ridge 对比

```
Coefficient  1 -0.007
Coefficient  2 0.003
Coefficient  3 -0.011
Coefficient  4 0.123
Coefficient  5 -0.594
Coefficient  6 0.171
```

```
Coefficient  7 -0.002
Coefficient  8 0.018
Coefficient  9 -0.007
Coefficient 10 -0.000
Coefficient 11 -0.040
Coefficient 12 0.001
Coefficient 13 -0.018

Intercept 22.975

Coefficient  1 -0.007
Coefficient  2 0.003
Coefficient  3 -0.012
Coefficient  4 -0.007
Coefficient  5 -0.007
Coefficient  6 0.055
Coefficient  7 -0.002
Coefficient  8 0.013
Coefficient  9 -0.007
Coefficient 10 -0.000
Coefficient 11 -0.033
Coefficient 12 0.001
Coefficient 13 -0.017

Intercept 23.287

# 基本稳定!
```