Ridge Regression

December 1, 2018

```
In [1]: %matplotlib inline

        import numpy as np
        import matplotlib.pyplot as plt
        import pandas as pd

        def main():
            x1 = np.random.uniform(low=-2, high=10, size=12)
            x_train = get_x_train(x1)
            y_train = get_y_train(x1)

            print_training_data(x1, y_train)

            w = linear_regression(x_train, y_train)
            best_r = determine_reg_param(x_train, y_train)
            w_reg = ridge_regression(x_train, y_train, regularizer=best_r)

            print_regression_eq('without reg', w, round(calculate_error(w, x_train, y_train), 
            print()
            print_regression_eq('with reg', w_reg, round(calculate_error(w_reg, x_train, y_tra

            plot_exp(x1, y_train, w, w_reg)


        def determine_reg_param(x_train, y_train):
            regularizers = np.array([0.1, 1, 10, 100])
            cv_error = np.zeros_like(regularizers)
            ein = np.zeros_like(regularizers)

            for i, r in enumerate(regularizers):
                w = ridge_regression(x_train, y_train, regularizer=r)
                ein[i] = calculate_error(w, x_train, y_train)
                cv_error[i] = cross_validation(x_train, y_train, 3, r)

            print_cross_validation_results(regularizers, ein, cv_error)

            return regularizers[np.argmin(cv_error)]
```

```
def print_cross_validation_results(regularizers, ein, cv_error):
    cv_summary = np.round(np.column_stack((regularizers, ein, cv_error)), 2)
    df = pd.DataFrame(cv_summary, columns=['lambda', 'in-sample error', 'cv error'])
    print(str(df))
    print('choose lambda = ' + str(regularizers[np.argmin(cv_error)]) + '\n')


def cross_validation(x_train, y_train, num_splits, regularizer):
    d = np.column_stack((x_train, y_train))
    cv_error = np.zeros(num_splits)

    for i, this_d in enumerate(np.split(d, num_splits)):
        x_train, y_train = this_d[:, :-1], this_d[:, -1]
        w = ridge_regression(x_train, y_train, regularizer)
        cv_error[i] = calculate_error(w, x_train, y_train)

    return np.average(cv_error)


def calculate_error(w, x_train, y_train):
    y = np.dot(x_train, w)
    return np.average(np.square(np.subtract(y, y_train)))


def ridge_regression(x_train, y_train, regularizer):
    I = np.identity(x_train.shape[1])
    xTx = np.dot(x_train.T, x_train)
    xTy = np.dot(x_train.T, y_train)
    return np.dot(np.linalg.pinv(np.add(xTx, regularizer * I)), xTy)


def linear_regression(x_train, y_train):
    return np.dot(np.linalg.pinv(x_train), y_train)


def get_x_train(x1):
    bias = np.ones_like(x1)
    return np.column_stack((bias, x1))


def get_y_train(x1):
    return np.square(x1) + 10


def print_training_data(x1, y_train):
    d = np.column_stack((x1, y_train))
    d_sorted = d[np.argsort(d[:, 0])]
    df = pd.DataFrame(np.round(d_sorted, 2), columns=['x', 'y'])
    print(str(df) + '\n')


def print_regression_eq(label, w, error):
    print('{0:12}: y = {1}x + {2}'.format(label, round(w[1], 2), round(w[0], 2)))
    print('{0:12}: {1}'.format('error', error))


def plot_exp(x1, y_train, w, w_reg):
    plt.style.use('seaborn-whitegrid')
    fig, ax = plt.subplots()

    ax.set(title='Ridge Regression')
    ax.scatter(x1, y_train, color='b', marker='x', label='training data')
    ax.plot(x1, line(w[1], x1, w[0]), color='g', label='regression line')
    ax.plot(x1, line(w_reg[1], x1, w_reg[0]), color='r', label='ridge regression line'

    ax.legend(facecolor='w', fancybox=True, frameon=True, edgecolor='black', borderpad
    plt.show()


def line(m, x, b):
    return m * x + b


if __name__ == '__main__':
    main()
```

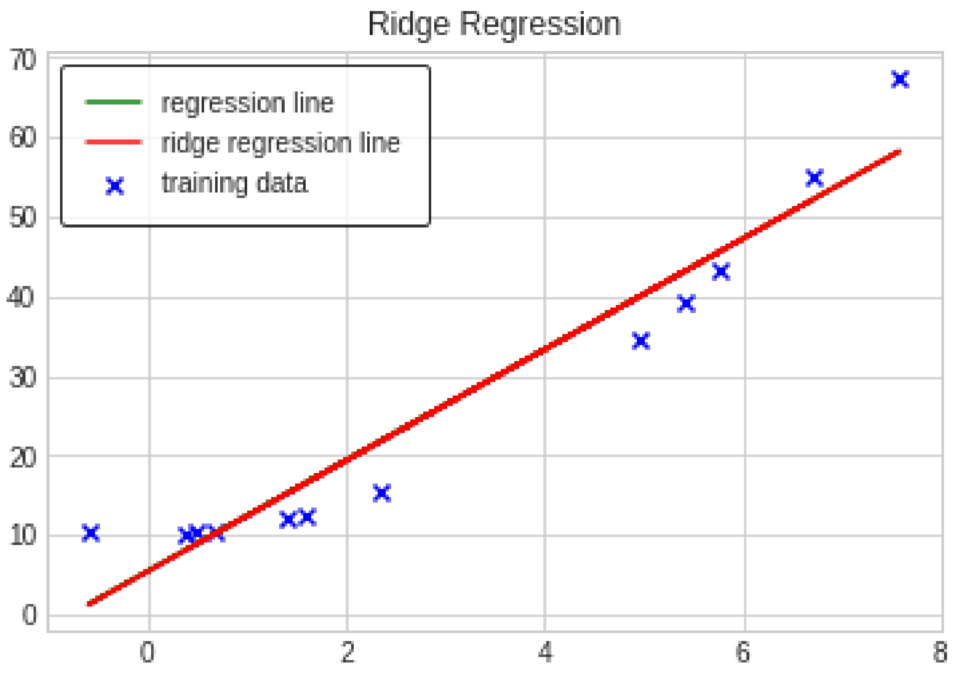
```
       x      y
0  -0.60  10.36
1   0.37  10.14
2   0.49  10.24
3   0.67  10.45
4   1.42  12.00
5   1.59  12.52
6   2.33  15.45
7   4.96  34.60
8   5.42  39.40
9   5.77  43.34
10  6.71  55.01
11  7.58  67.49

   lambda  in-sample error  cv error
0     0.1            24.81     15.63
1     1.0            25.02     17.03
```

```
2    10.0             30.18     40.94
3   100.0            143.70    379.07
choose lambda = 0.1

without reg : y = 6.95x + 5.47
error       : 24.8

with reg    : y = 6.97x + 5.39
error       : 24.81
```


Ridge Regression
regression line
ridge regression line
training data