Instruction

1. Gradient descent is a very common optimization method. I use the Concrete Compressive Strength dataset as my data. My goal is: use gradient descent to optimize linear regression, and solve the regression coefficients to complete the prediction of Concrete compressive strength(MPa, megapascals).
2. For uni-variate linear regression, I set learning rate is 0.000001. And iterators is 1000. And I could get right output. For multi-variate linear regression, I built a grad method for finding the gradient and then built a grad\_descent method for computing regression coefficients. The learning rate is 0.00001, and iterators is 100. The input x and y are both in dataFrame format. In addition to variables such as the basic learning rate, the variable error is added, which is used to limit the number of iterations. If the final error is less than error, the iteration stops.
3. {

def grad():

Substitute the formula for the sum of each point

Calculate the gradient, calculate the partial derivative of theta

Return gradient

end

}

{

def grad\_descent():

computing regression coefficients. Build a classifier.

Return the number of iterations, the error of the last iteration

end

}

{

Read data (see data distribution).

Split the data ensemble train and test sets.

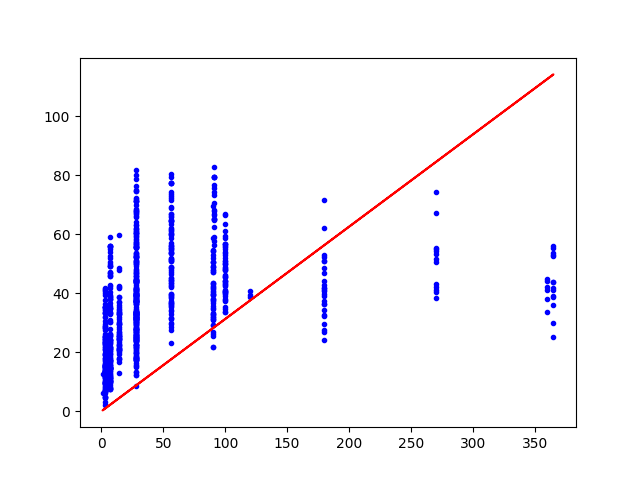
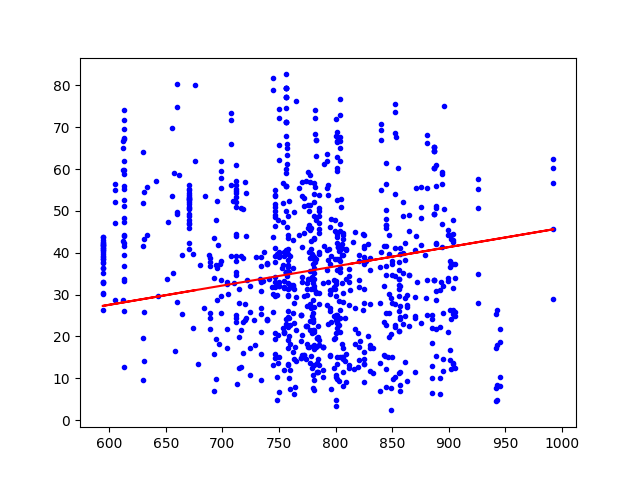
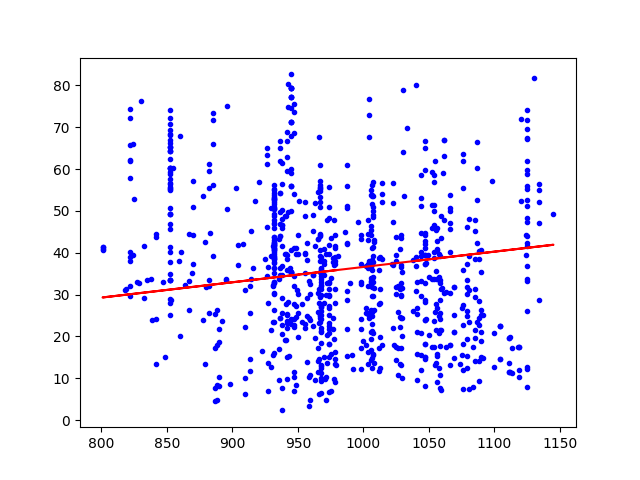
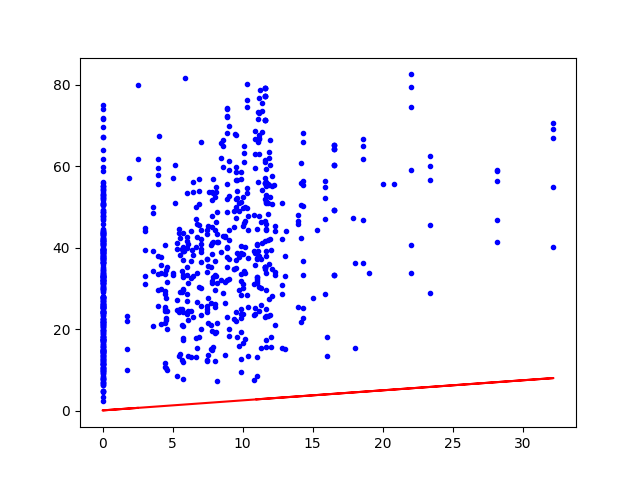
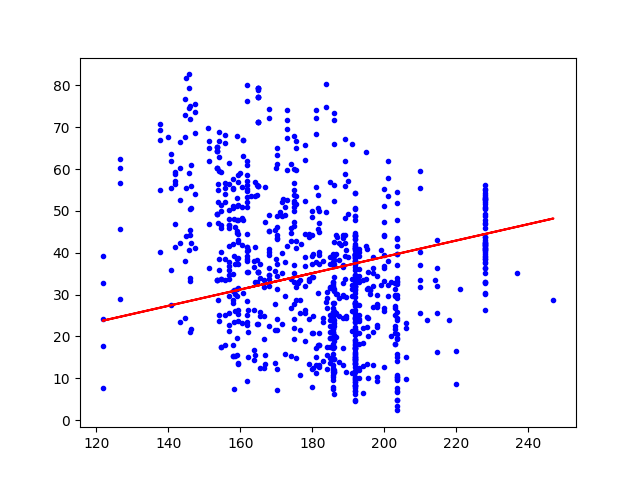
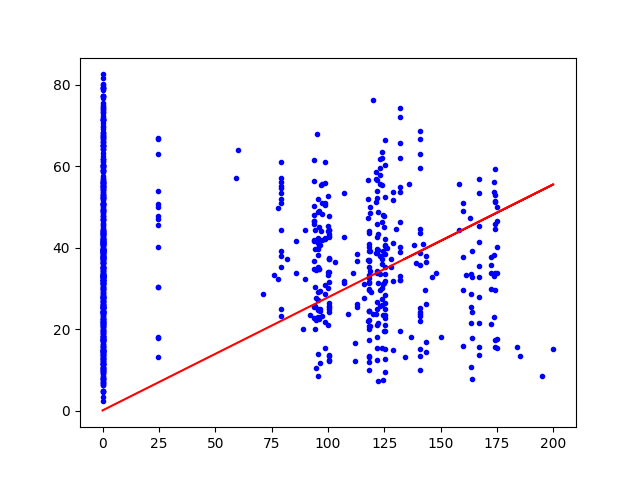
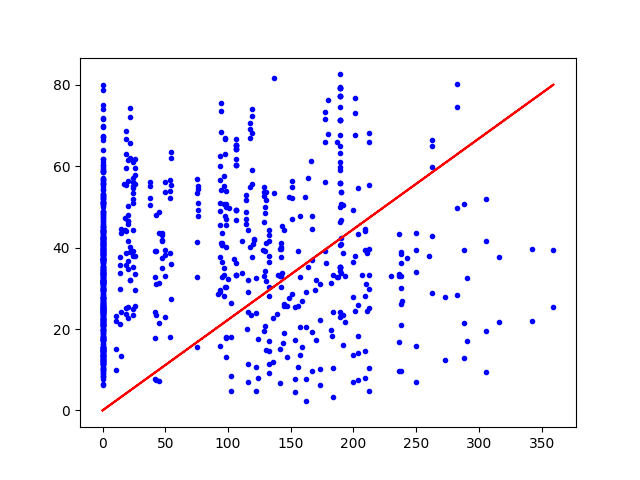
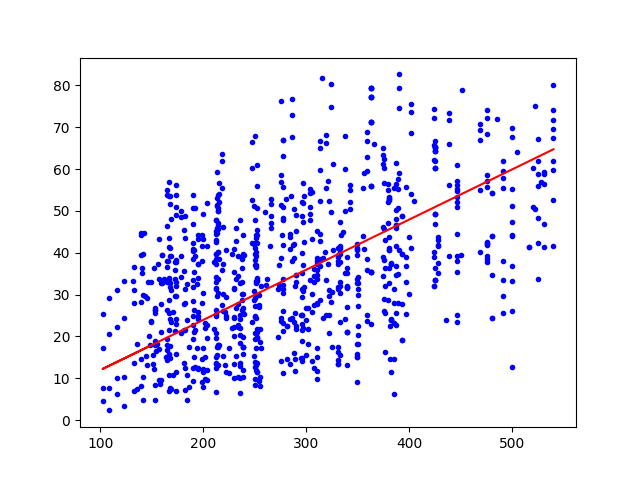
Set the threshold and complete the data result according to the threshold. Specify the parameters of the gradient algorithm and print the relevant parameters, and draw the gradient change pattern.

}

Results

1. For uni-variate regression, loss is 124.68332372136797, 509.8145618030323, 567.2471714947116, 179.0959927455514, 747.4091635524044, 161.1727551657253, 166.71789201708057, 474.21881733581586. Variance(y)=278, so variance explained is 0.55, -0.83, - 1.03, 0.36, -1.68, 0.42, 0.40, -0.70.

For multi-variate regression, last iteration error is 2.985407300826452e+92. (With np.random.randint method to randomly generate theta). Varience is 0.98.

1. For uni-variate regression, loss is 40.66801278184359, 173.46040094477024, 339.6669471599731, 80.79593257489091, 510.07378580288787, 81.05391515290735, 80.78910075319833, 178.064695848207. Variance is 0.85, 0.37, -0.21, 0.71, -0.83, 0.70, 0.70, 0.35.
2. 

Discussion

1. Compared with BDG, MBGD has a faster convergence speed on the training set; it also avoids the problem of premature convergence and large fluctuation of the SGD algorithm. No. The data can be random and noisy. So we cannot make sure that it also work well on the testing data.
2. Multiple linear regression analysis prediction method: refers to the method of establishing a prediction model for prediction through the correlation analysis of two or more independent variables and a dependent variable. So it failed to predict the coefficients in the multi-variate model. Because we only get one independent variable.
3. Cement (component 1)(kg in a m^3 mixture). Because it has the lowest loss.