ML HW1

1 Linear Algebra Review

Machine Learning HW1

1.

1.1

$$2x_1 + 2x_2 + 3x_3 = 1$$
$$-3x_1 + 6x_2 + 3x_3 = 6$$
$$5x_1 - 4x_2 = -5$$
$$-5x_1 - 5x_2 = -5$$
$$x_2 = 0$$

$$5x_1 = -5 \Rightarrow x_1 = -1$$

$$1 + x_3 = 2 \Rightarrow x_3 = 1$$

$$x_1 = -1,\ x_2 = 0,\ x_3 = 1$$

1.2

$$\begin{bmatrix} 2 & 2 & 3 \\ 1 & -1 & 0 \\ -1 & 2 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix}$$

1.3 3

1.4

$$\left[\begin{array}{ccc|ccc} 2 & 2 & 3 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 & 1 & 0 \\ -1 & 2 & 1 & 0 & 0 & 1 \end{array}\right]$$

$$A^{-1} = \begin{bmatrix} 1 & -4 & -3 \\ 1 & -5 & -3 \\ -1 & 6 & 4 \end{bmatrix}$$

$$\det(A) = -2 + 6 - 3 - 0 - 2 = -1$$

$$= \left[\begin{array}{ccc|ccc} 5 & -4 & 0 & 1 & 0 & -3 \\ 1 & -1 & 0 & 0 & 1 & 0 \\ -1 & 2 & 1 & 0 & 0 & 1 \end{array}\right]$$

$$\rightarrow = \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 1 & -4 & -3 \\ 1 & -1 & 0 & 0 & 1 & 0 \\ -1 & 2 & 1 & 0 & 0 & 1 \end{array}\right] = \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 1 & -4 & -3 \\ 0 & 1 & 0 & 1 & -5 & -3 \\ 0 & 2 & 1 & 1 & -4 & -2 \end{array}\right] = \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 1 & -4 & -3 \\ 0 & 1 & 0 & 1 & -5 & -3 \\ 0 & 0 & 1 & -1 & 6 & 4 \end{array}\right]$$

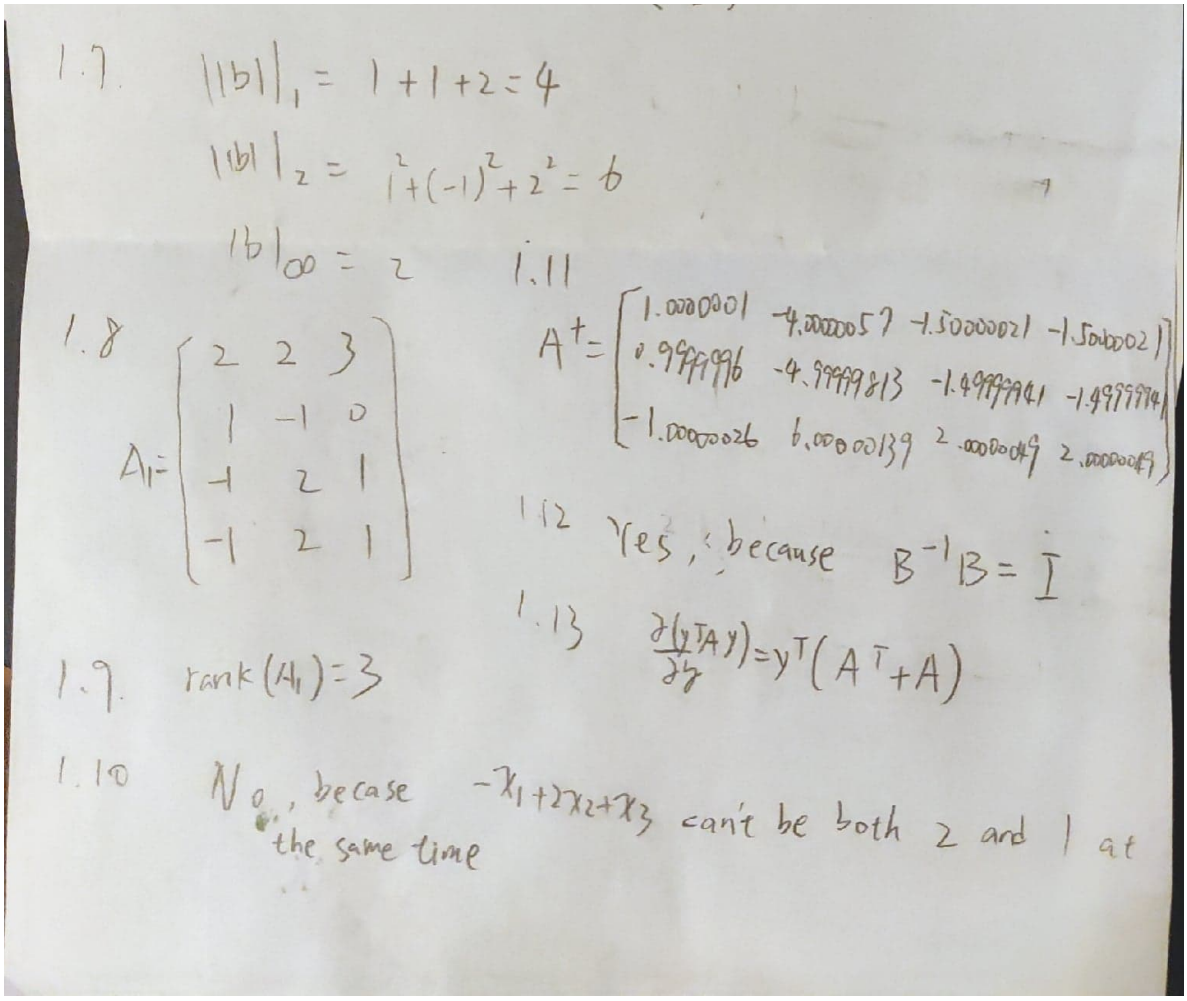
1.5

$$A \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix}$$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = A^{-1} \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 & -4 & -3 \\ 1 & -5 & -3 \\ -1 & 6 & 4 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$

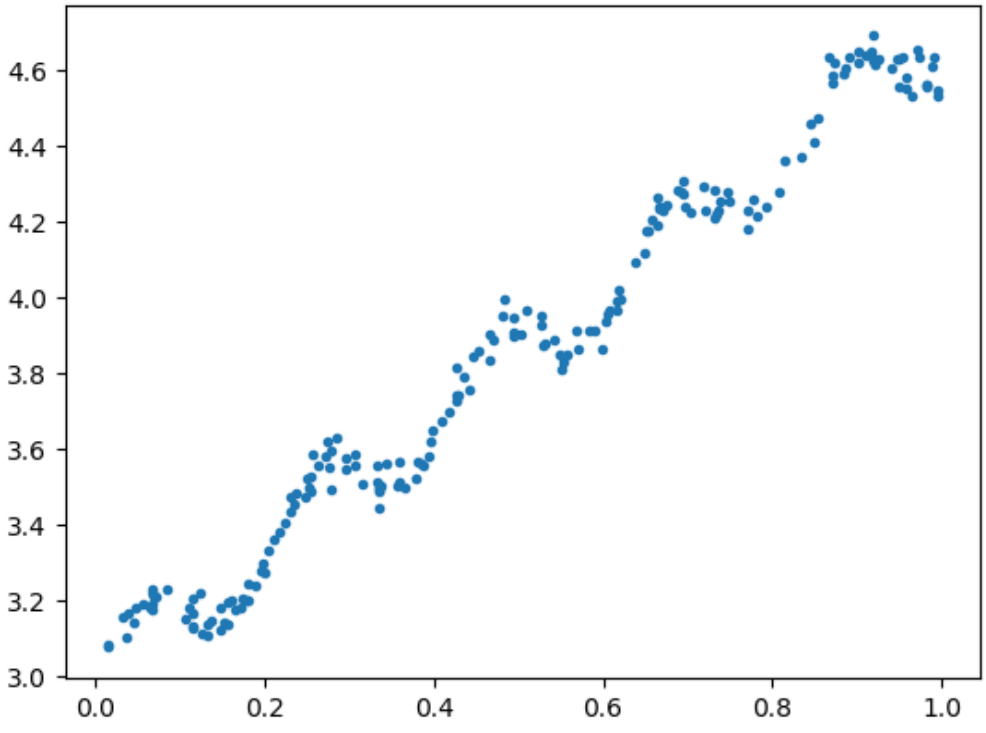
1.6.

$$\langle x, b \rangle = x^T b = \begin{bmatrix} 1, -1, 2 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix} = 1 + 1 + 4 = 6$$



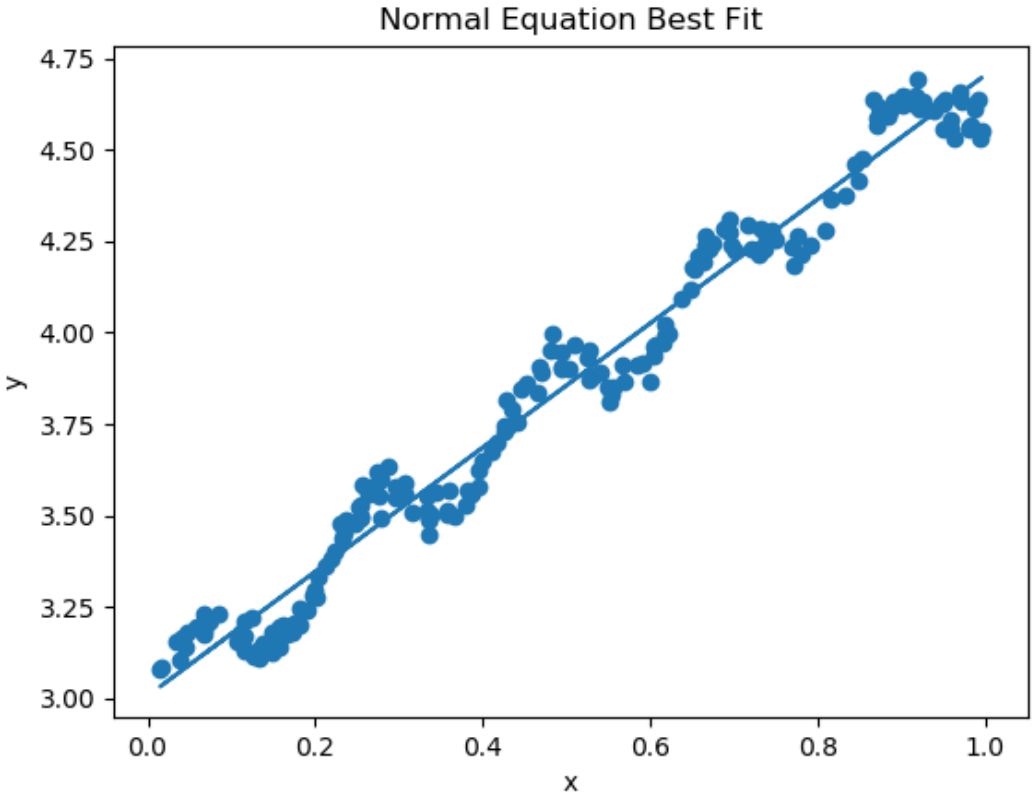
2 Linear Regression Model Fitting (Programming)

Data Set



Normal Equation

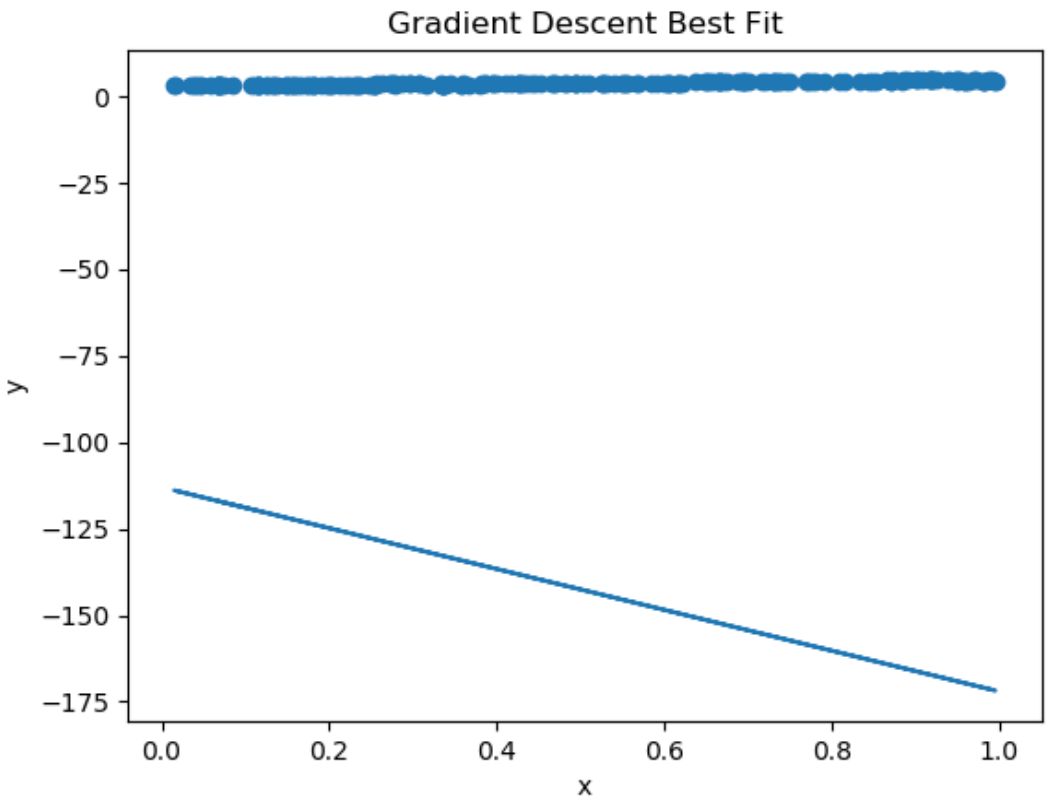
theta = `[[3.00774324] [1.69532264]]`

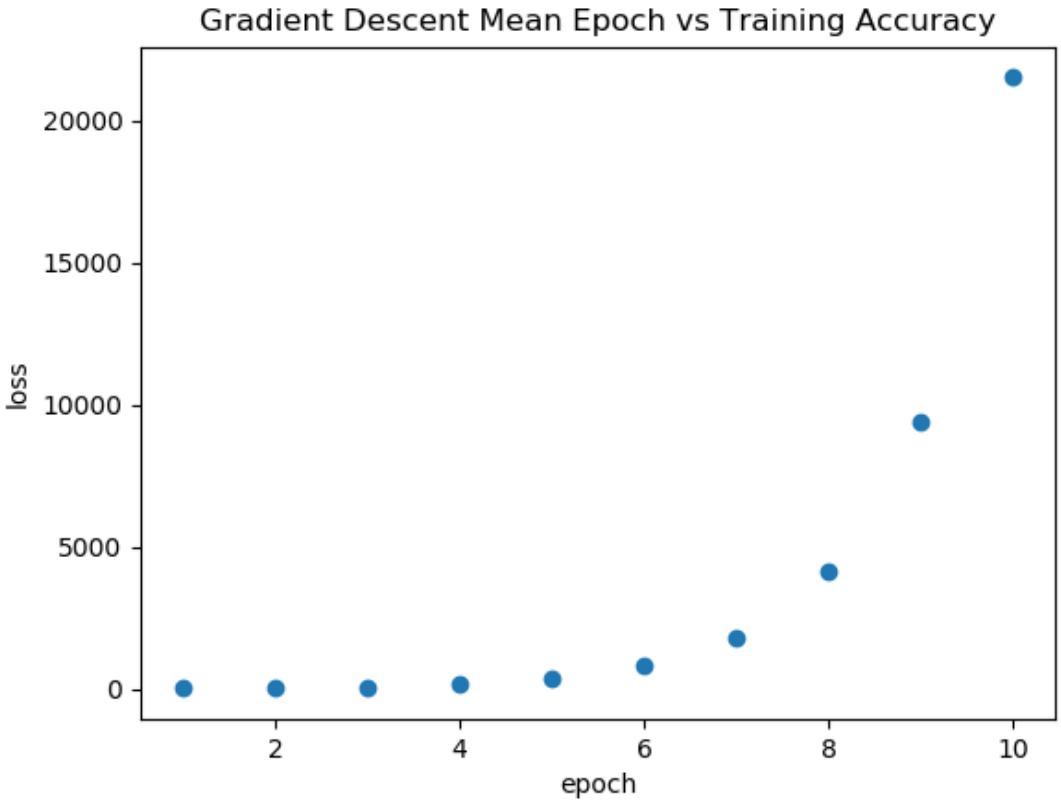


Gradient Descent

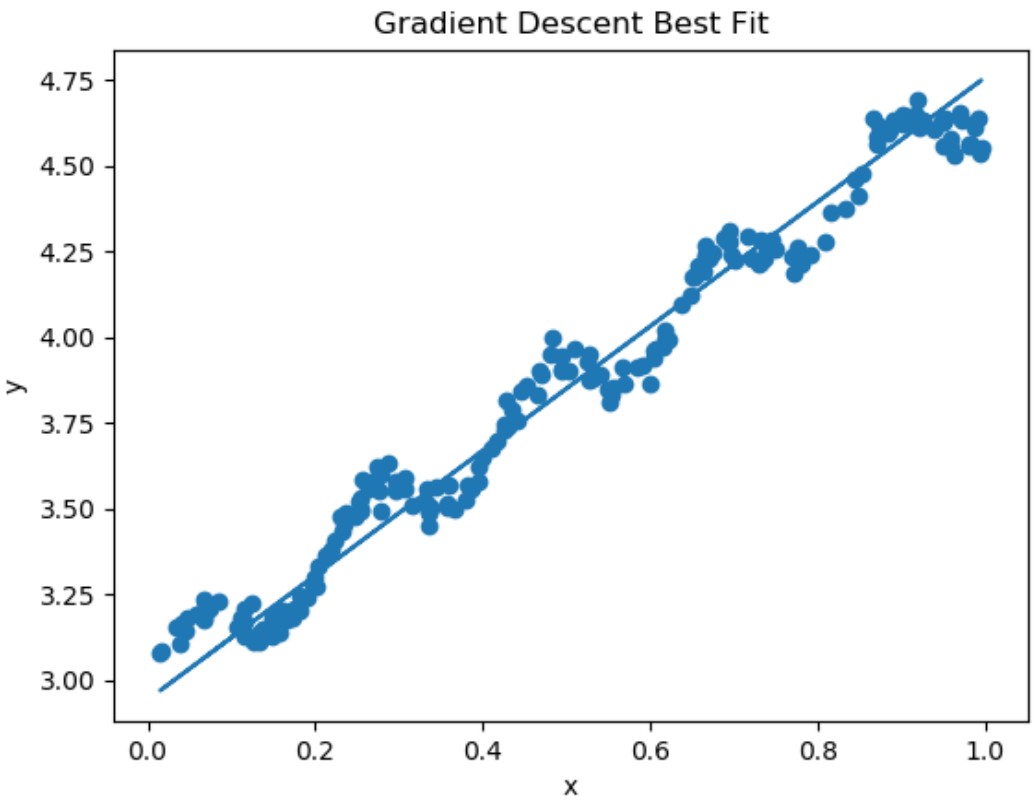
Results

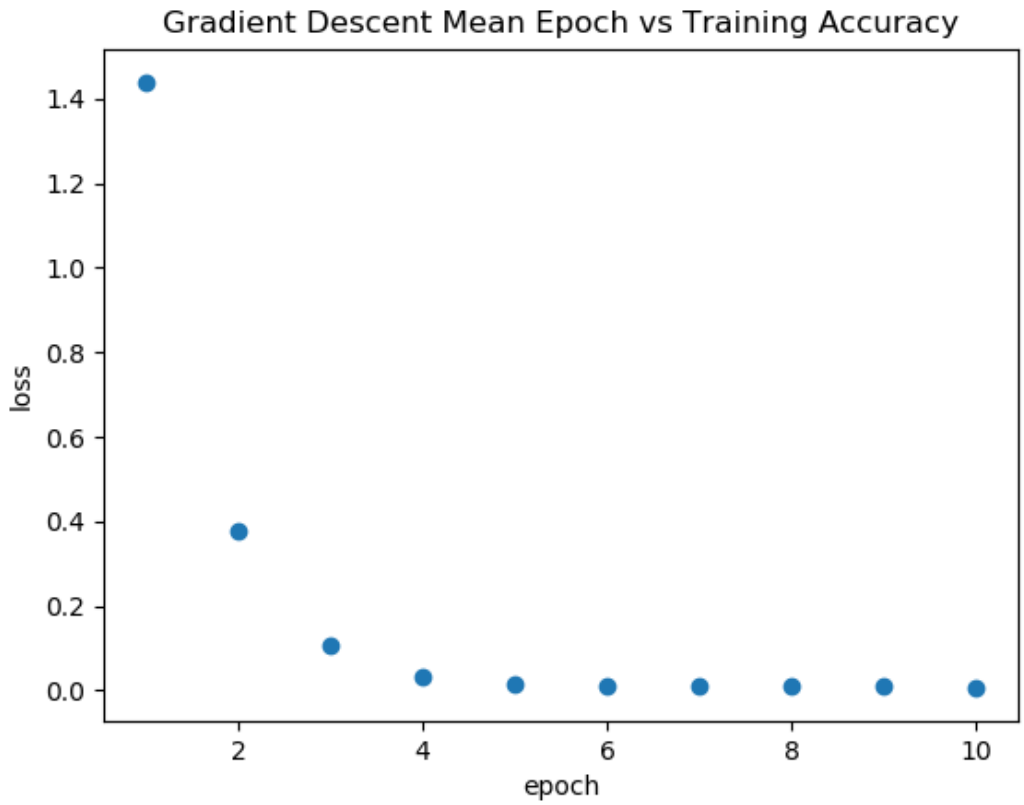
learning rate: 0.01 number of iterations: 10 theta = `array([[-113.02169208], [ -59.01472316]])`



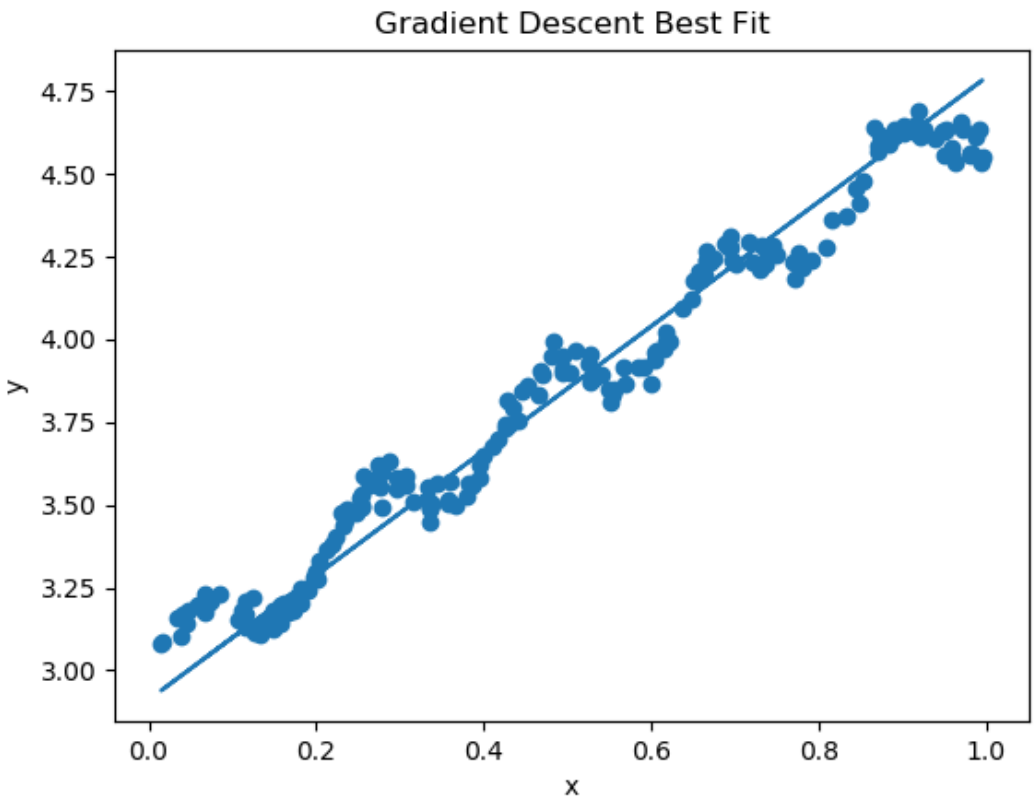


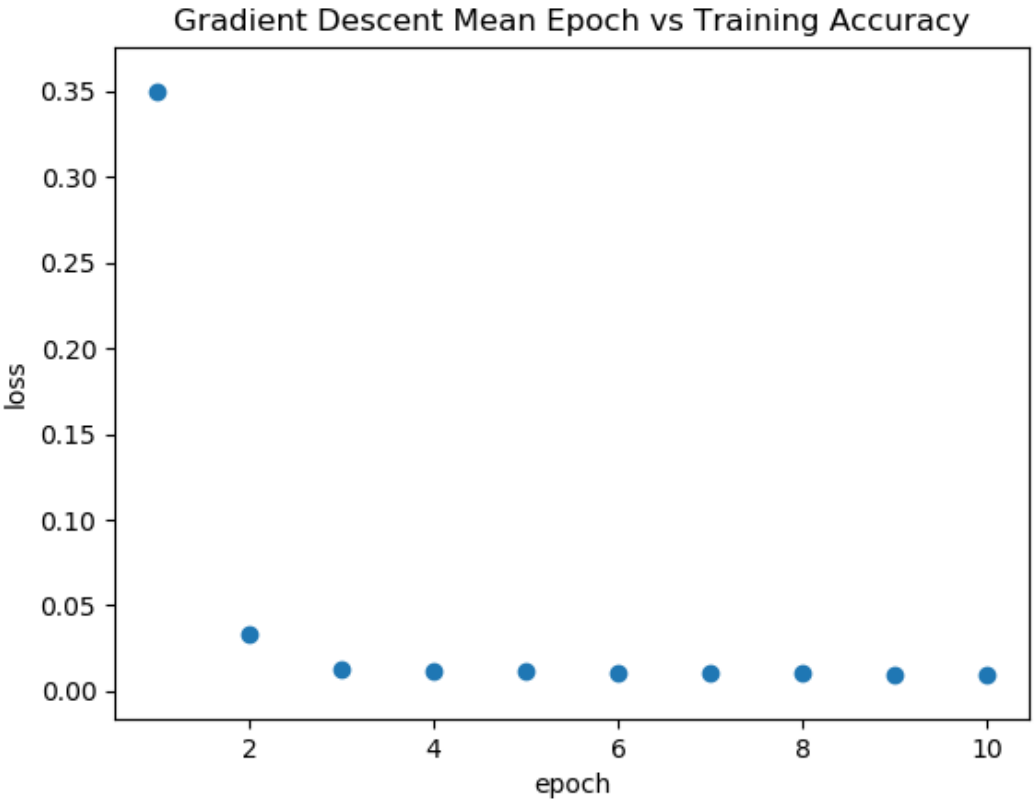
learning rate: 0.006 number of iterations: 10 theta = `array([[2.94306258], [1.81368982]])`





learning rate: 0.003 number of iterations: 10 theta = `array([[2.91107894], [1.87981306]])`



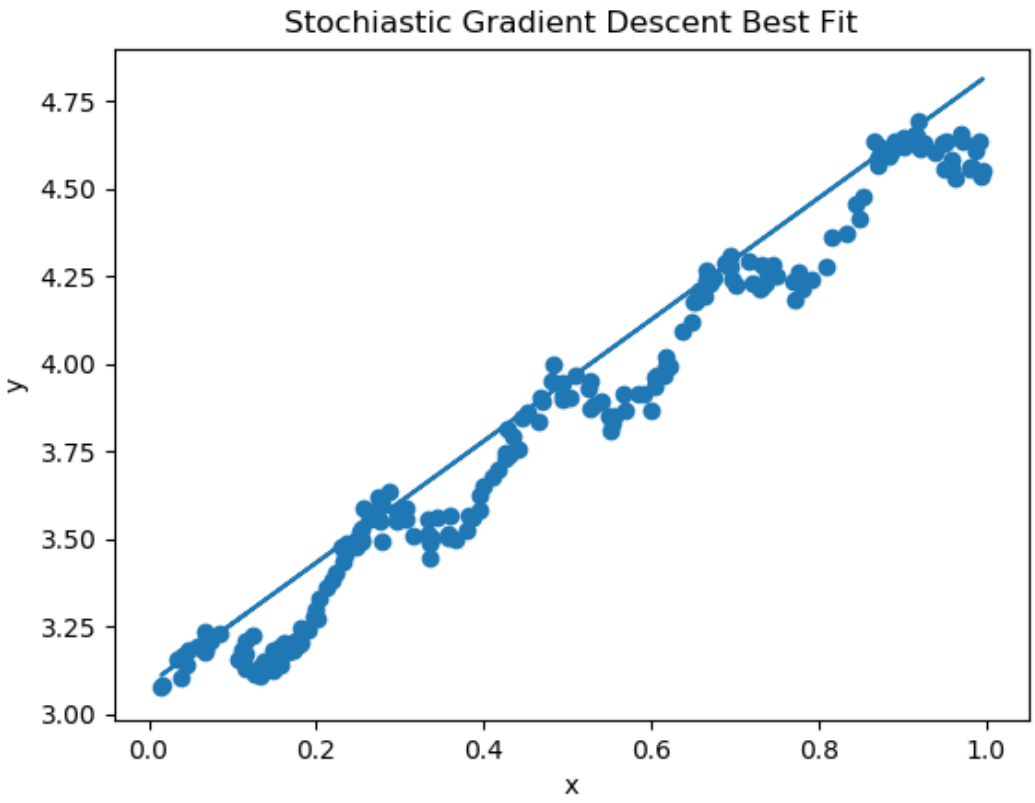


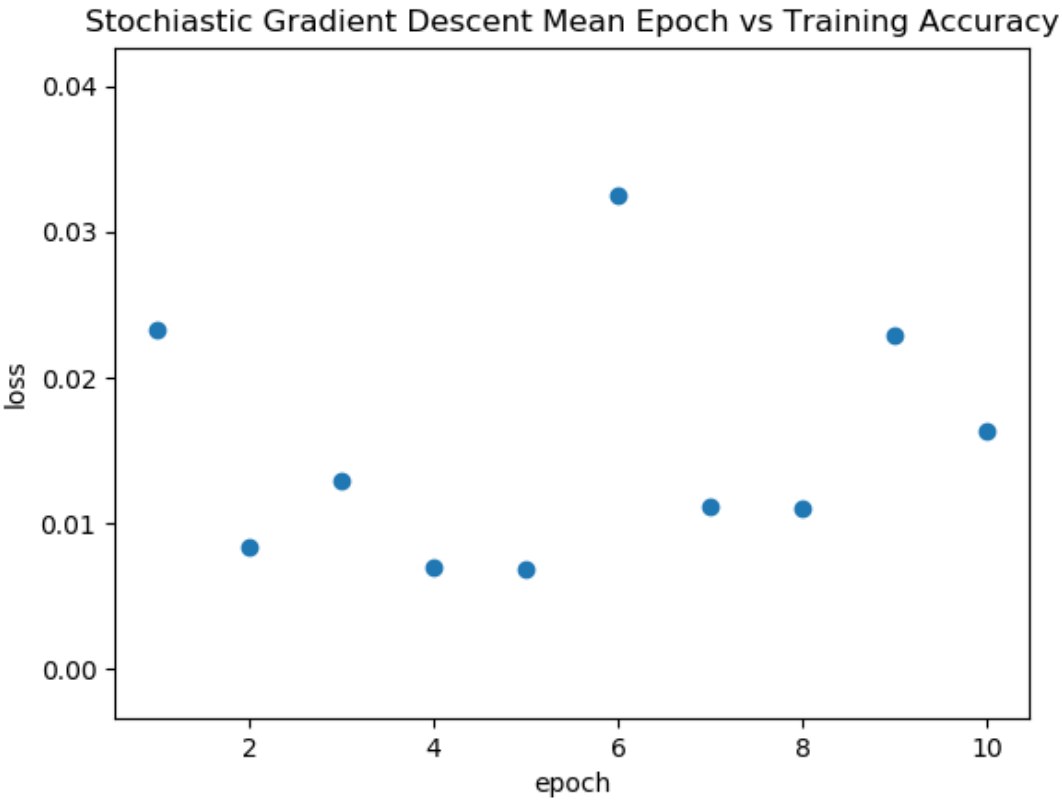
Observation

As shown above, too large a learning rate resulted in the loss function not converging, while learning rates of 0.003 and 0.006 were small enough for the loss function to converge.

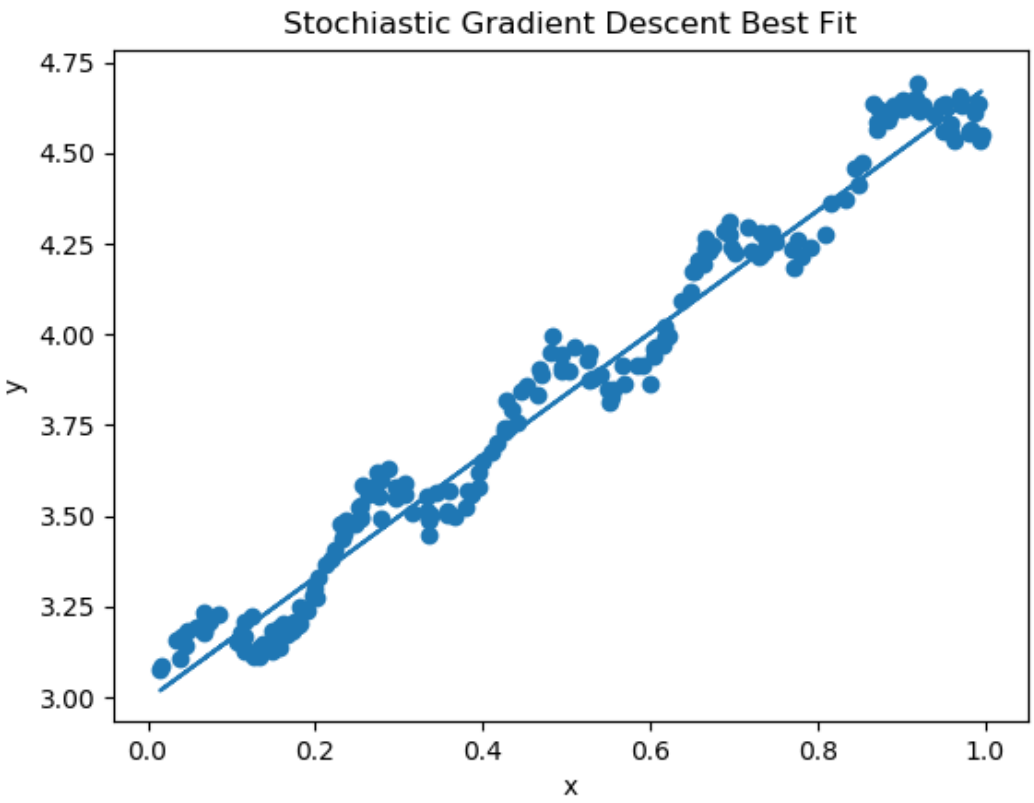
Stochastic Gradient Descent

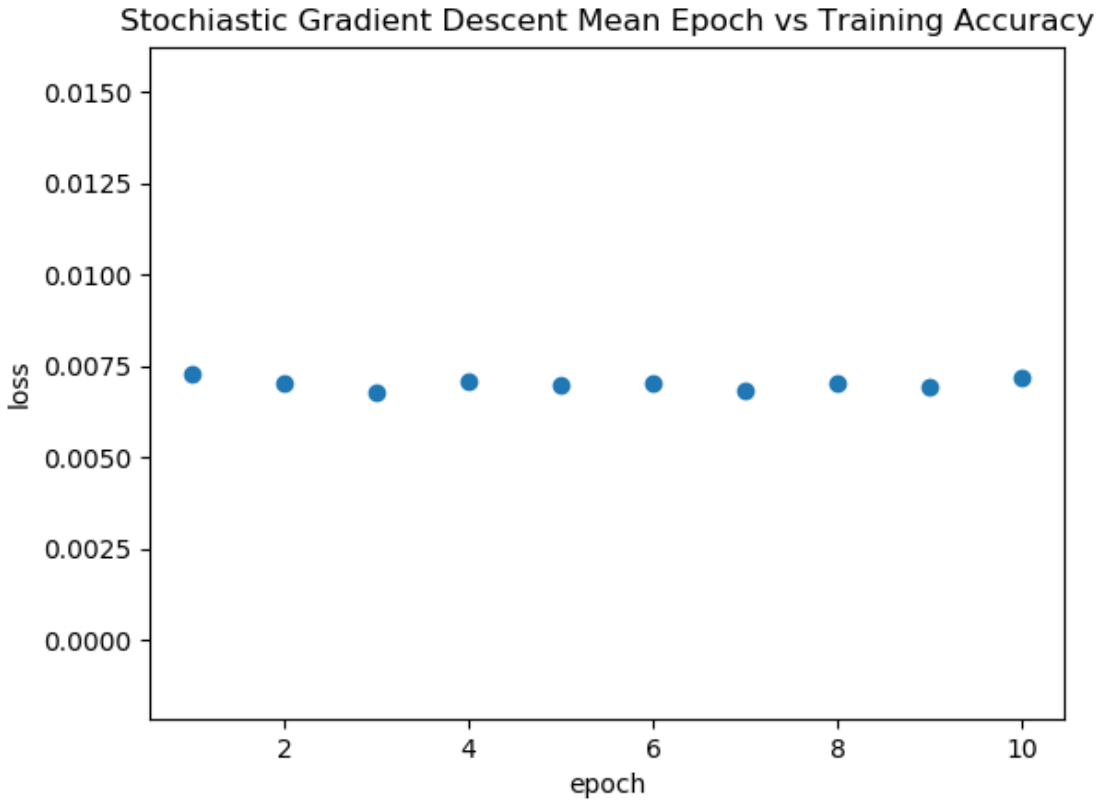
learning rate: 1 number of iterations: 10 theta = `array([[3.08500241], [1.73524859]])`



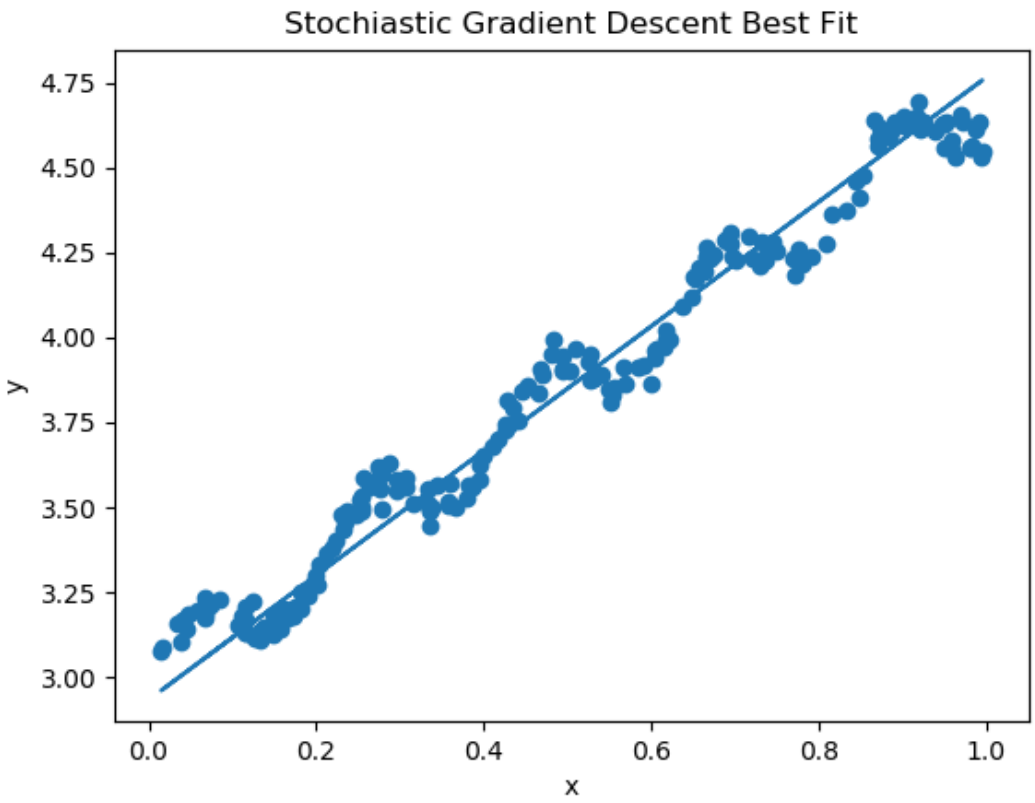


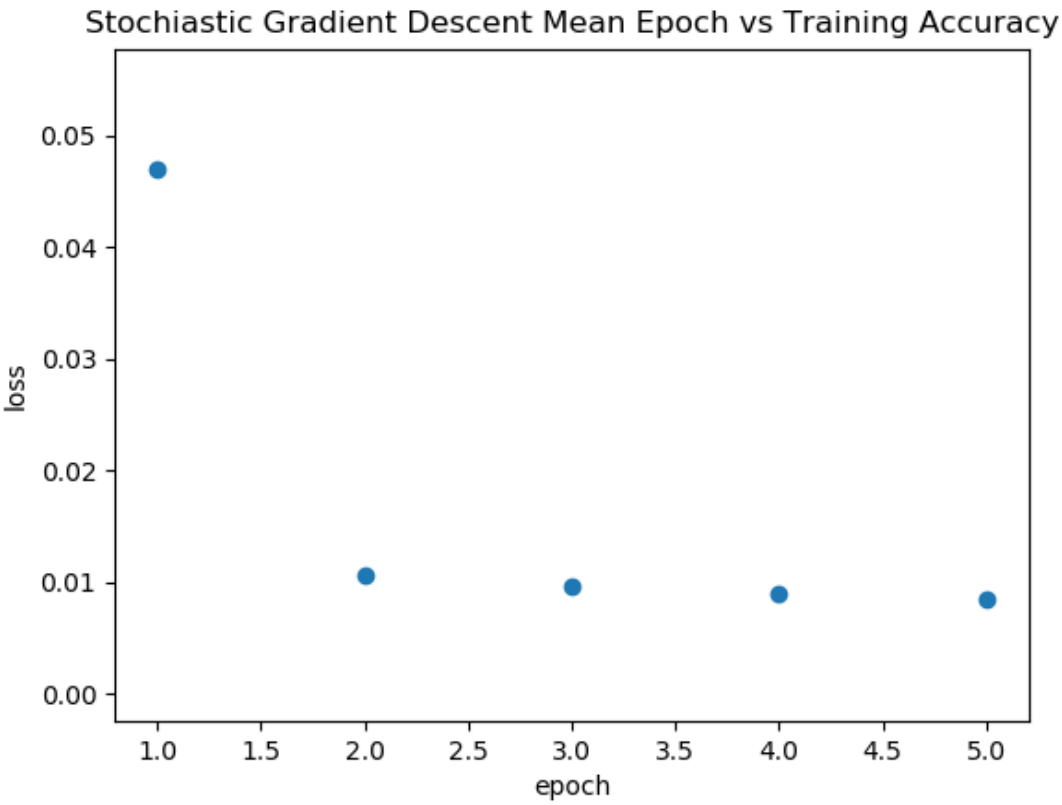
learning rate: 0.1 number of iterations: 10 theta = `array([[2.9940233 ], [1.68317826]])`





learning rate: 0.01 number of iterations: 5 theta = `array([[2.93335447], [1.83177765]])`





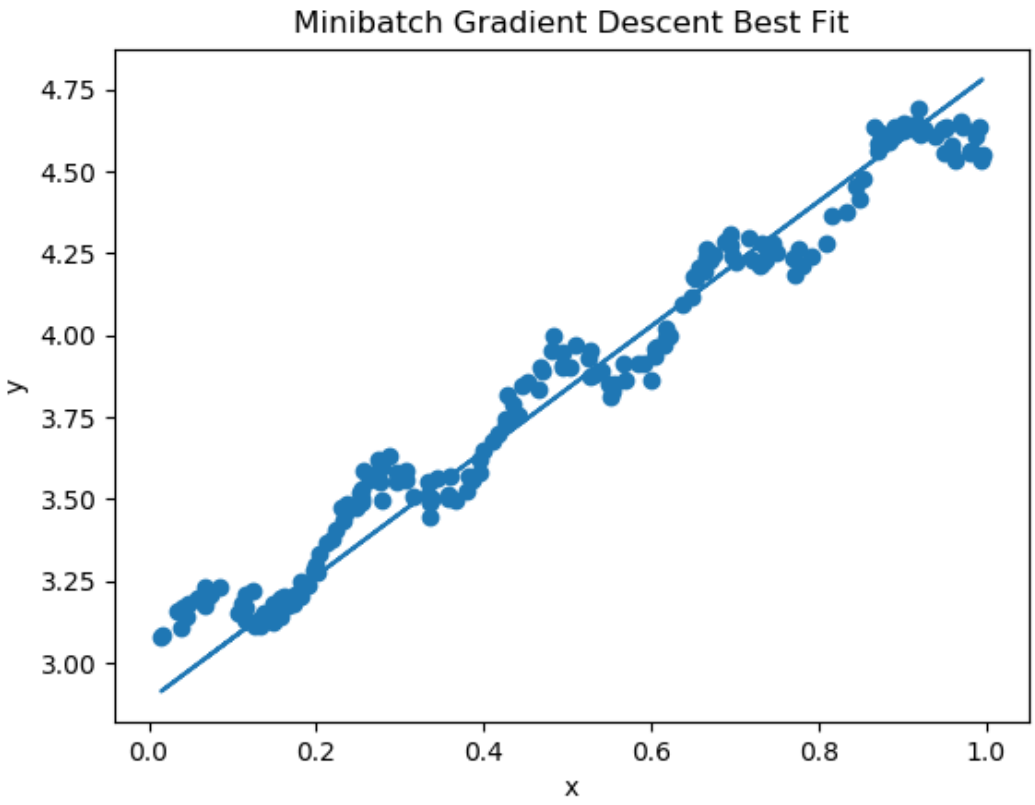
Observation

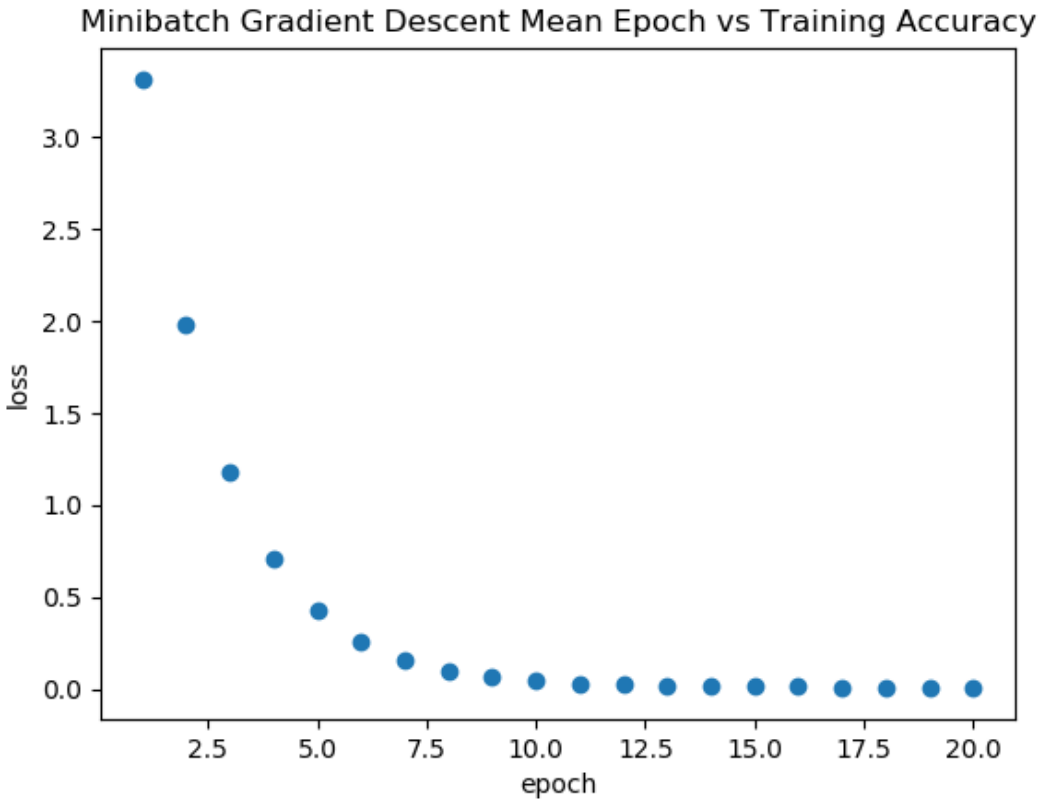
As we can see in the above figure, even with a learning rate of 1, SGD still managed to fit pretty well to the data points. But if we look at the loss function, we can see that does not increase as epoch increases, so 1 is still too large for the loss function to converge. When the learning rate is brought down to 0.1 or 0.01, the loss function converges and no longer jumps up and down.

Minibatch Gradient Descent

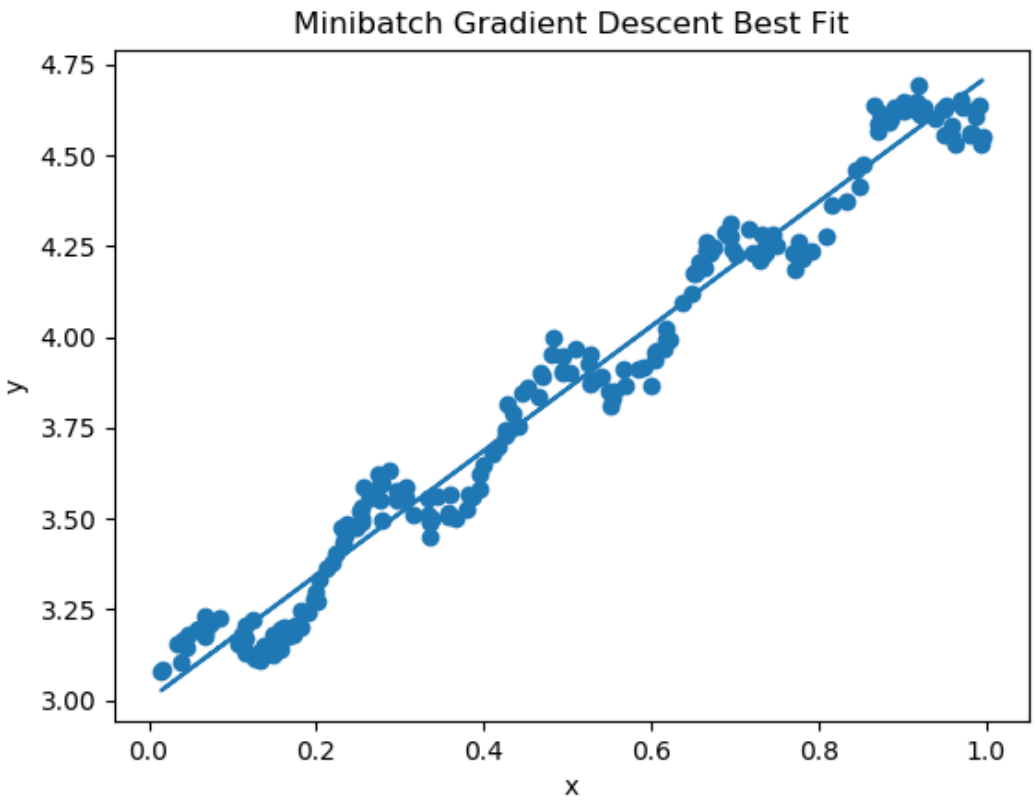
Result

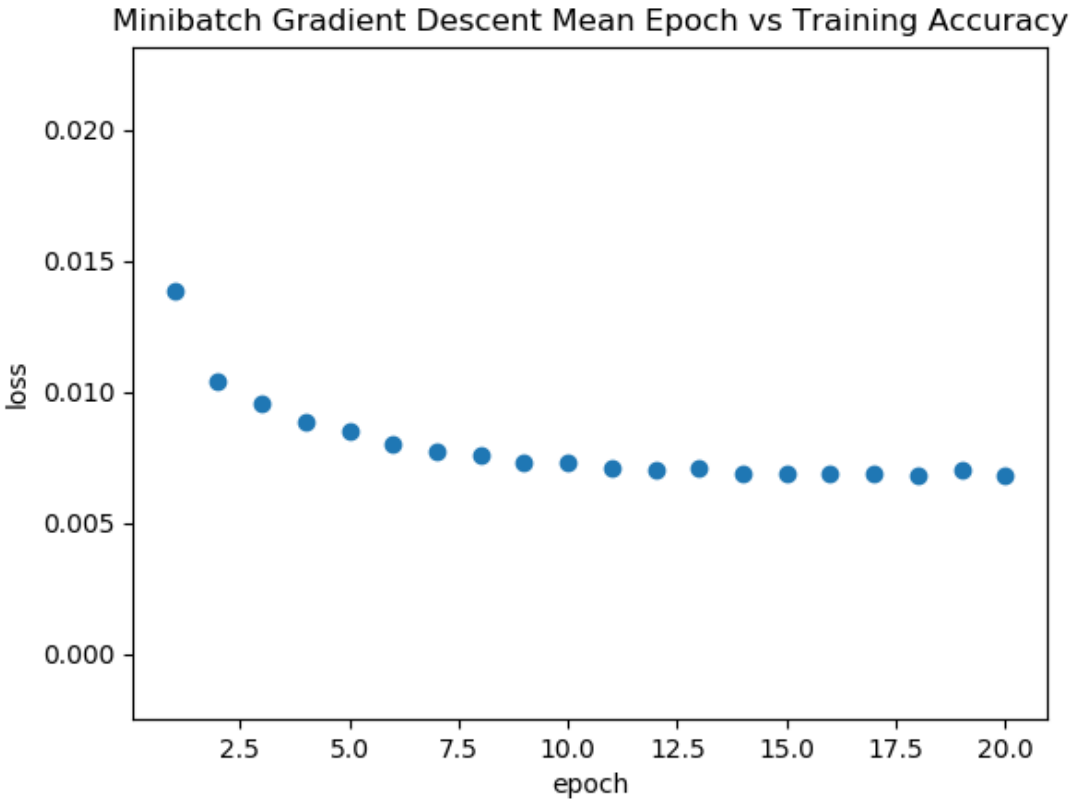
learning rate: 0.001, number of iterations: 20, batch size: 50, theta = `array([[2.88485029], [1.90411717]])`





learning rate: 0.01 number of iterations: 20, batch size: 50 theta = `array([[3.00127283], [1.71374226]])`





Observation

Converged faster when the learning rate was larger.

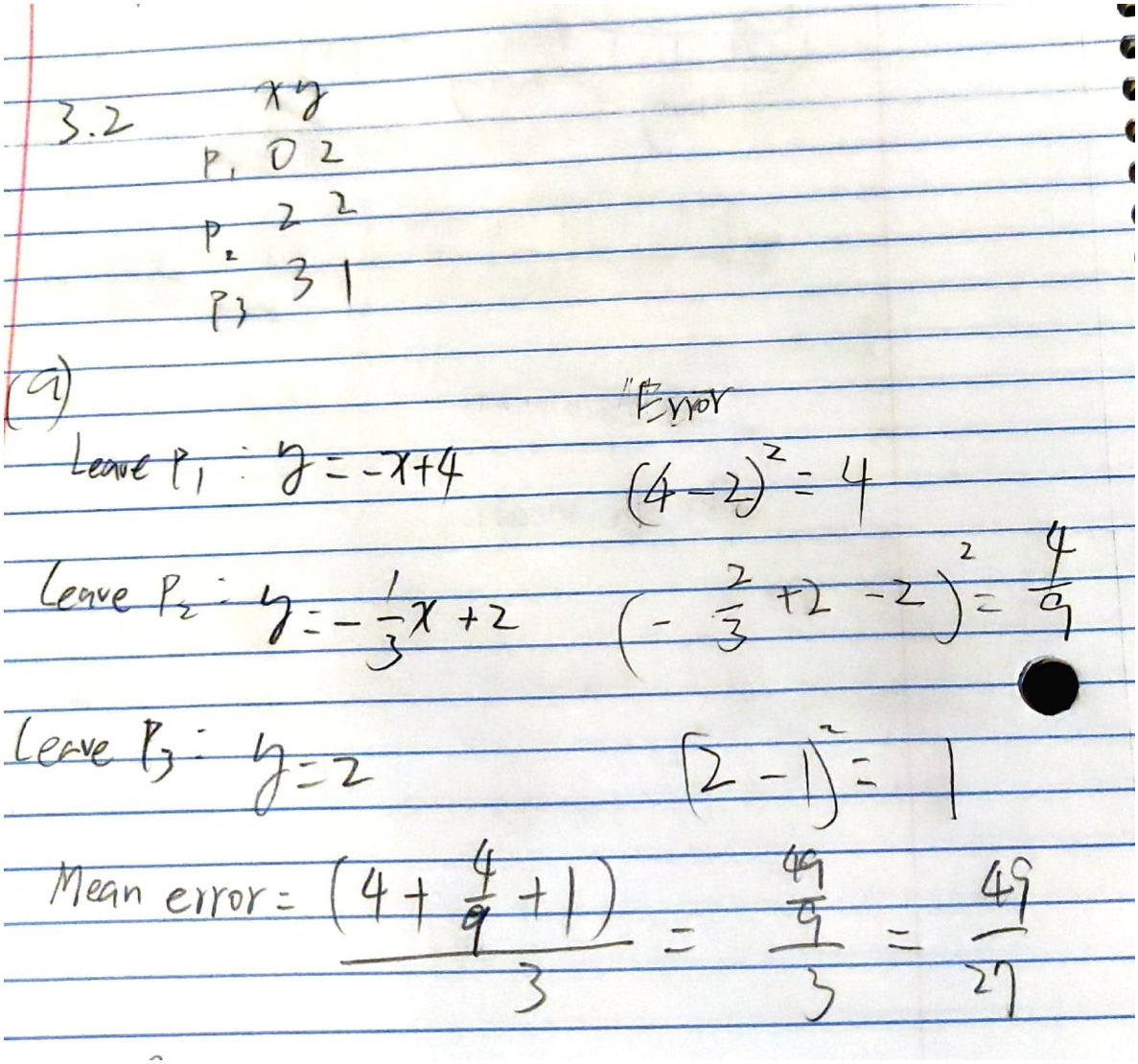
3 Sample Exam Questions:

3.1

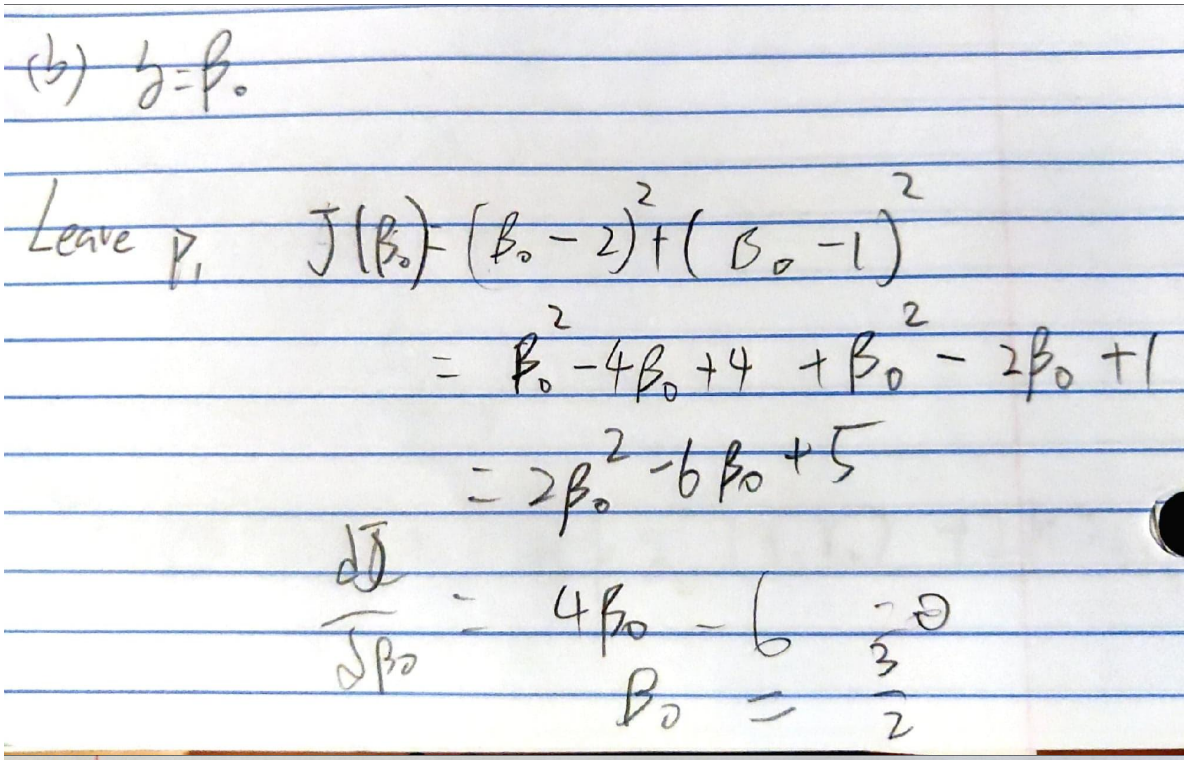
The MSE will be 0 as every point in the data set sits perfectly on the same line, so we can use linear regression to find a line to fit the data set perfectly.

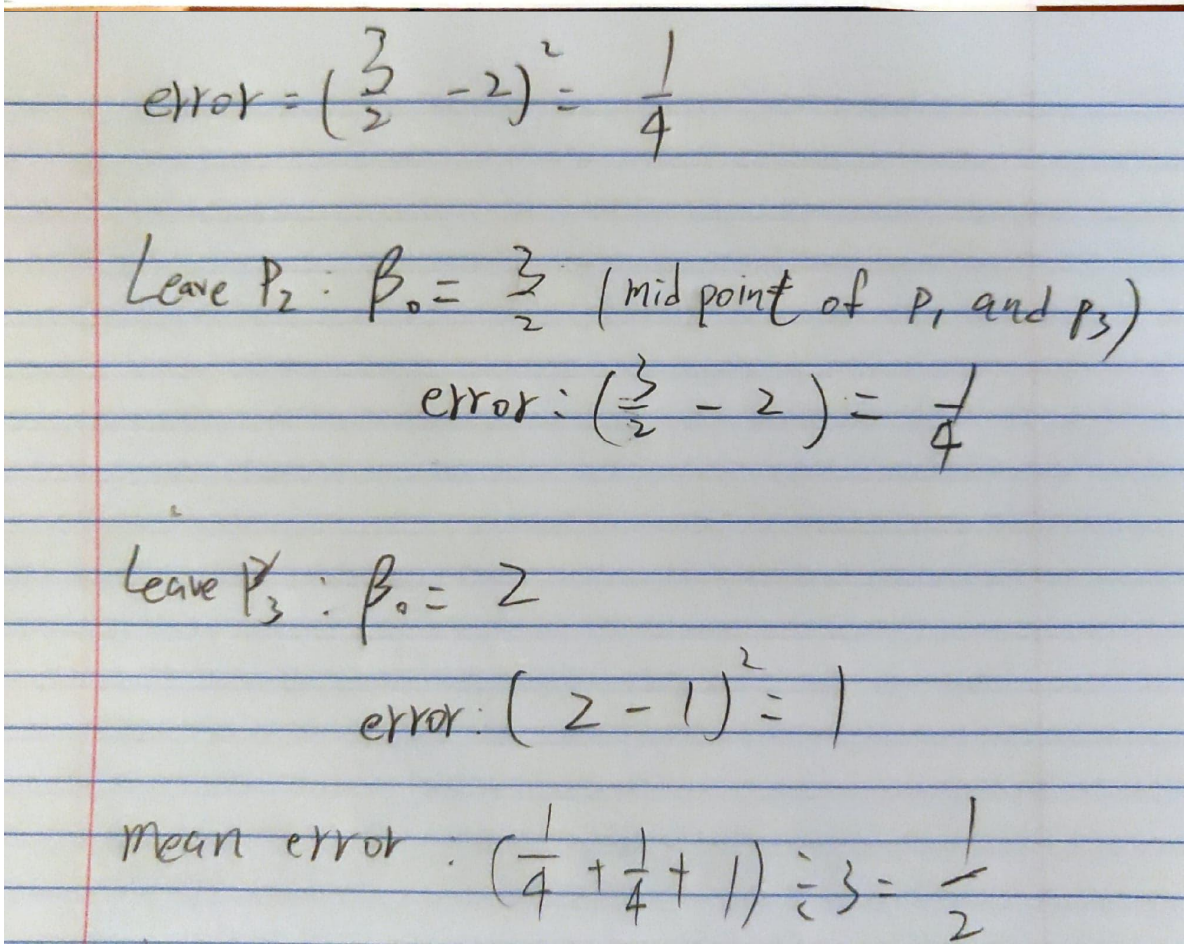
3.2

(a)



(b)





(c) The second model, since it produced a smaller error

3.3

(e) A, because the smaller the training data set is, the easier it is for us to find a line that fits the data points. Two points, for example, can be perfectly fit with 0 training error.

(f) B, because as we increase the size of the training data set, the model should generalize better and be able to make better predictions on testing data.