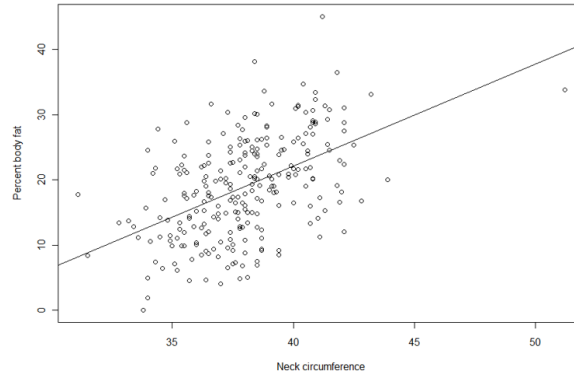# Simple Linear Regression (SLR)



Simple linear regression model

- **Notation:**
    - m: number of training examples
    - x: input variable
    - y: output variable
    - $x^i$: $i^{th}$ input variable
    - $y^i$: $i^{th}$ output variable
    - $(x^i, y^i)$: $i^{th}$ training example
- **Hypothesis:**

$$h_\theta(x) = \theta_0 + \theta_1 x$$

- **Cost (loss) functions:**
    - Squared error (SE) cost function:

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left( h_\theta(x^{(i)}) - y^{(i)} \right)^2$$

    - Sum squared error (SSE) cost function:

$$J(\theta_0, \theta_1) = \sum_{i=1}^{m} \left( h_\theta(x^{(i)}) - y^{(i)} \right)^2$$

    - Mean squared error (MSE) cost function:

$$J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} \left( h_\theta(x^{(i)}) - y^{(i)} \right)^2$$

    - Root mean squared error (RMSE) cost function:
        - For classification problem, RMSE is not a good choice for loss function

$$J(\theta_0, \theta_1) = \sqrt{\frac{\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2}{m}}$$

# Simple Linear Regression (SLR)

- **General gradient descent algorithm for simple linear regression:**

  Repeat until convergence {

  $$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \; ; \; (for \; j = 0, 1)$$

  }

- **Gradient descent algorithm using SE loss function in SLR:**

  Repeat until convergence {

  $$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} [(1) \times (h_\theta(x^{(i)}) - y^{(i)})]$$

  $$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} [(x^i) \times (h_\theta(x^{(i)}) - y^{(i)})]$$

  }

- **Note:**
  - We have a cost function and we want to minimize the cost function to obtain an optimal solution for the hypothesis function → Use gradient descent algorithm to continuously finding $\theta_0, \theta_1$ values that are closest to the $\min J(\theta_0, \theta_1)$
  - The $\alpha$ value is called: learning rate.
    - It determines the aggressiveness of the learning algorithm
    - If $\alpha$ is too small → gradient descent can be slow
    - If $\alpha$ is too big → gradient descent can overshoot the minimum. It may fail to converge, or even diverge
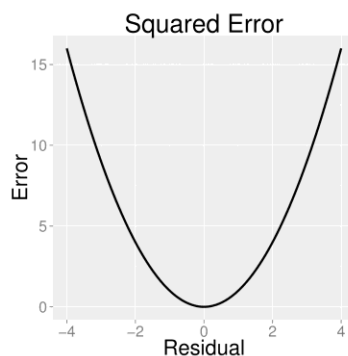


Example of the squared-error loss function

# Simple Linear Regression (SLR)
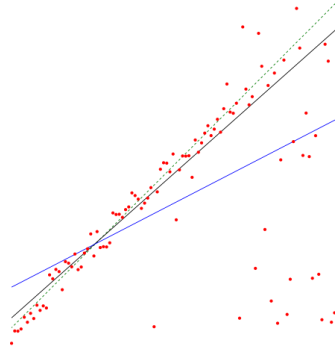


Example of different hypothesis base on different set of $(\theta_0, \theta_1)$

- For SLR, when we plot the cost function $J(\theta_0, \theta_1)$, the resulting graph is a convex function (has only 1 global minimum). Because of this, another approach from using gradient descent is to set the partial derivative with respect to $\theta_0, \theta_1$ to be $0$ in order to find the minimum.
- Another approach to find the hypothesis function would be using Pearson's R correlation coefficient (PCC):

$$\theta_0 = \overline{y} - \theta_1 \overline{x}$$

$$\theta_1 = r \left( \frac{s_y}{s_x} \right)$$

$$r = \frac{\sum_{i=1}^{m} \left( z_{x^{(i)}} \right) \left( z_{y^{(i)}} \right)}{m - 1}$$

$$s_x = \sqrt{\frac{\sum_{i=1}^{m} (x^{(i)} - \overline{x})^2}{m - 1}} \qquad s_y = \sqrt{\frac{\sum_{i=1}^{m} (y^{(i)} - \overline{y})^2}{m - 1}}$$

$$z_{x^{(i)}} = \frac{x^{(i)} - \overline{x}}{s_x} \qquad z_{y^{(i)}} = \frac{y^{(i)} - \overline{y}}{s_y}$$

$$\overline{x} = \frac{\sum_{i=1}^{m} x^{(i)}}{m} \qquad \overline{y} = \frac{\sum_{i=1}^{m} y^{(i)}}{m}$$