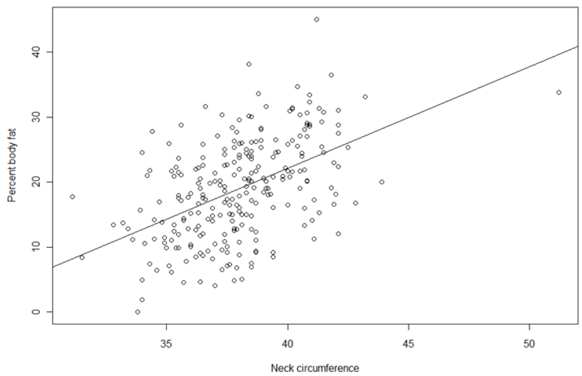
Simple Linear Regression (SLR)



Simple linear regression model

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

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

- **<u>Cost (loss) functions:</u>**
  - Squared error (SE) cost function:

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

  - Sum squared error (SSE) cost function:

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

  - Mean squared error (MSE) cost function:

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

  - 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)

- **<u>General gradient descent algorithm for simple linear regression:</u>**

  Repeat until convergence {

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

  }

- **<u>Gradient descent algorithm using SE loss function in SLR:</u>**

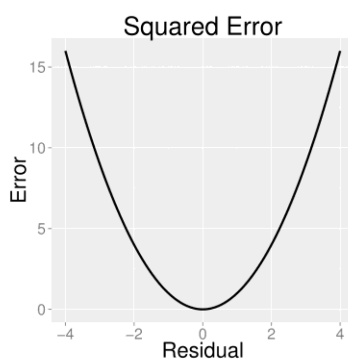
  Repeat until convergence {

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

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

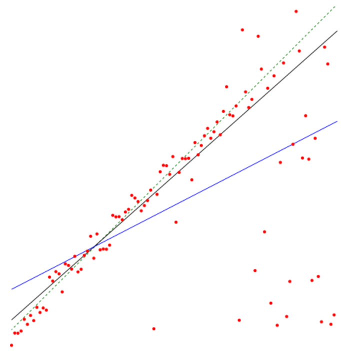
  }

- **<u>Note:</u>**
  - 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 θ_0, θ_1 values that are closest to the $\min J(\theta_0, \theta_1)$
  - The α value is called: learning rate.
    - It determines the aggressiveness of the learning algorithm
    - If α is too small → gradient descent can be slow
    - If α 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 (θ_0, θ_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 θ_0, θ_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):

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

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

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

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

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

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

- Before deciding whether we should apply SLR or not, if we compute the Pearson's R correlation coefficient and find that $|r| \geq \frac{2}{\sqrt{m}}$, then we can say that a linear relationship exists