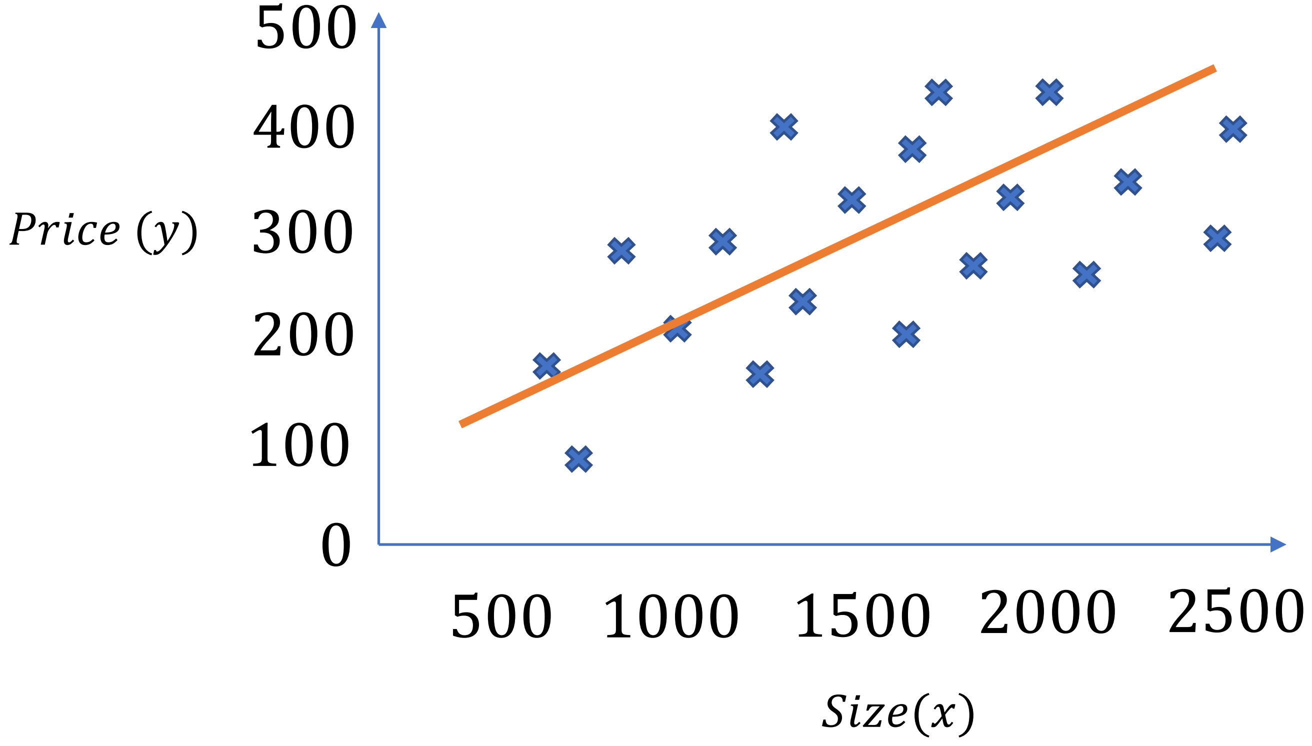
Basic Regression

By: Nur Rohman Widiyanto

Institut Teknologi Sepuluh Nopember
IEEE Student Branch

Simple Explaination



<u>Supervised Learning</u>

Given the “Right answer” for example in the data

<u>Regression Problem</u>

Predict real-valued output

Problem

- Build a Simple Linear Regression Model to predict sales based on the **money spent** on **TV advertising**.
  - Decide whether TV advertising is effective or not and if it is effective
  - How much money the company should spend on TV advertising to get a particular increase in sales.

- Let's inspect the datasets **@Notebook** #1 2_1_data_processing.ipynb →

Institut Teknologi Sepuluh Nopember
IEEE Student Branch

From Scratch

Dataset understanding using Exploratory Data Analysis (EDA)

- The goal is to understand our dataset.
- EDA techniques can be divided into two types:
  - Numerical techniques
  - Graphical techniques

- Let's inspect the datasets **@Notebook** #2 2_1_data_processing.ipynb →

Take the Conclusion

- Since there seems to be a ***linear relationship*** between TV advertising and sales.
  - We can build a linear regression model as our problem statement defines.
- What do you do if you see a non-linear relationship between the two variables?
  - You cannot build a linear regression model.

Analyze the distribution of variables

- Let's inspect the datasets @Notebook #2 →
- **Histogram of TV Ad spending:**
  - The distribution of TV Ad spending is symmetric.
  - The **mean and median are approximately the same**.
  - TV Ad spending follows a uniform distribution with a mean of 147.
  - The center is close to 150. The typical deviation in TV Ad spending from the mean is about 85.9 units.

Analyze the distribution of variables

- Let's inspect the datasets @Notebook #2 →
- **Histogram of sales:**
  - The distribution of sales is single-peaked (unimodal) and symmetric.
  - The mean and median are approximately the same.
  - Sales approximately follow a normal distribution with a mean of 15.13.
  - The center is close to 16.
  - The typical deviation in sales from the mean is about 5.2 units.

Regression Model

- Simple Linear Regression

$$h_\theta(x) = \theta_0 + \theta_1 x_1$$

- θ_0 and θ_1: Model parameters
- $h_\theta(x)$: Response variable
- x_1 : Predictor

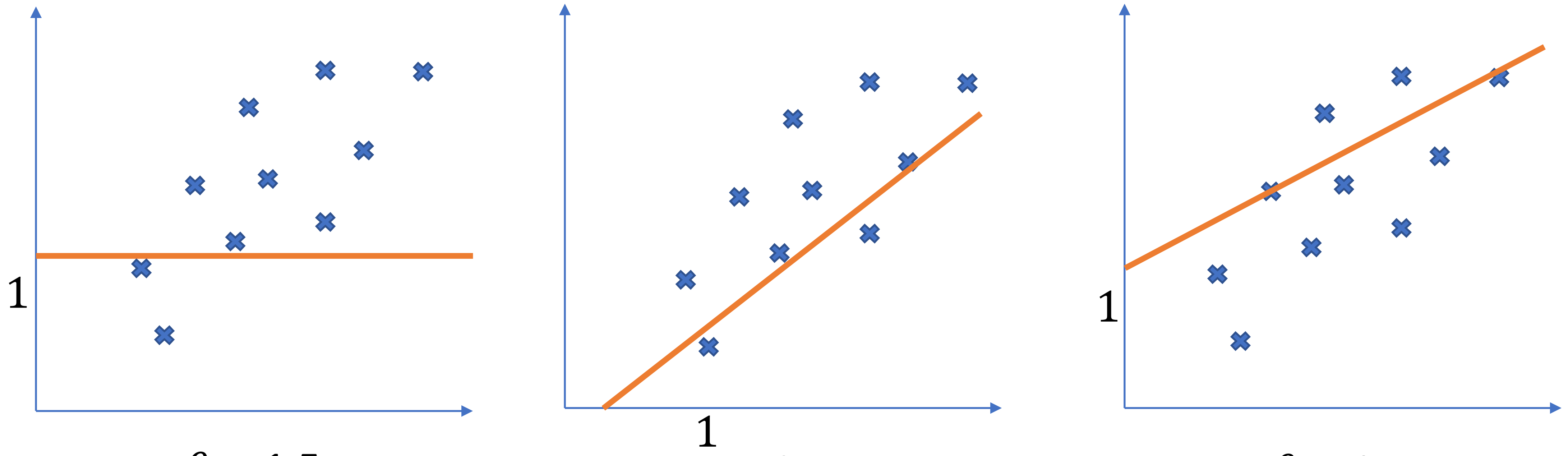
Cost Function

- Linear Regression

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

- m : Number of training examples
- $x(i)$: x-value of *i*-th training example
- $y(i)$: y-value of *i*-th training example
- $J(\theta_0, \theta_1)$: Cost function or mean squared error function

Determine the $\boldsymbol{\theta_0}$ and the $\boldsymbol{\theta_1}$



θ_0 : 1.5
θ_1 : 0

θ_0 : 0
θ_1 : 0.5

θ_0 : 1
θ_1 : 0.5

Code now **@Notebook** #1
2_2_linear_regression_gradient_descent_from scratch.ipynb →

Institut Teknologi Sepuluh Nopember
IEEE Student Branch

Gradient Descent

- The gradient descent algorithm is an optimization algorithm that can be used to minimize the above cost function and find the optimized values for the linear regression model parameters.

$$repeat\ until\ convergence\ \{$$

$$\theta := \theta_{\mathrm{j}} - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

$$(for\ j = 1\ and\ j = 0)$$

$$\}$$

https://math.stackexchange.com/questions/70728/partial-derivative-in-gradient-descent-for-two-variables

Gradient Descent

- **:=** notation: Assignment operator. In programming, it is just = notation.
- $\boldsymbol{\alpha}$**:** Learning rate. It is a fixed value.
- **Derivative term:** The partial derivative of the cost function $J(\theta_0, \theta_1)$ for J=*0* and 1.

Choosing α (Learning rate)

- **α** is too small, the gradient descent can be slow. It will require more iterations (hence time) to reach the minimum.
- **α** is too large, the gradient descent can overshoot the minimum. In that case, it may fail to converge or even diverge.

Gradient Descent

$$
\begin{aligned}
&\text{repeat until convergence } \{ \\
&\quad \theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right) \\
&\quad \theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right) \cdot x^{(i)} \\
&\}
\end{aligned}
$$

Code now **@Notebook** #2
2_2_linear_regression_gradient_descent.ipynb →

Plot the Convergence

- Plot the result of Gradient Descent Process to search θ_0 and θ_1 as cost function or $J(\theta_0, \theta_1)$.

Let's inspect the datasets **@Notebook** #3
2_2_linear_regression_gradient_descent.ipynb →

Train the Model

θ_0 = Di peroleh dari proses Gradient Descent

θ_1 = Di peroleh dari proses Gradient Descent

$$Sales = \theta_0 + \ \theta_1 \times TV$$

Let’s inspect the datasets **@Notebook** #4
2_2_linear_regression_gradient_descent.ipynb →

Institut Teknologi Sepuluh Nopember
IEEE Student Branch

Evaluate the Model Performance

$$RSME = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(yi\ - \hat{y})^2}$$

$$R^2 = Corelation(x)^2$$

Let’s inspect the datasets **@Notebook** #5
2_2_linear_regression_gradient_descent.ipynb →

Institut Teknologi Sepuluh Nopember
IEEE Student Branch

Evaluate the Model Performance

- Plot the **Predicted Sales** compared with the **Real Sales**.
- Calculate the Residual and Analyze the Distribution.

$$R^2 = Corelation(x)^2$$

- Plot the **Predicted Sales** compared with the **Residual**.

Let’s inspect the datasets **@Notebook** #5
2_2_linear_regression_gradient_descent.ipynb →

Institut Teknologi Sepuluh Nopember
IEEE Student Branch

With Scikit

Steps

- Define *x* and *y* **#1**
- Split the dataset into a Training Set and a Testing Set **#2**
- Create and fit (train) the Model **#3**
- Get the Gradient and Intercept of the Linear Regression Line **#4**
- Make Predictions **#5**
- Evaluate the Model Performance **#6**
- Let's inspect the datasets @Notebook 2_3_linear_regression_with_gradient_descent_with_scikit →

With Normal Equation

Steps

$$\theta = (x^T x)^{-1}\, x^T y$$

- **θ**: Parameter matrix which contains optimized values.
- **x**: Feature matrix. It is represented as a 2d NumPy array. The shape is (m, n+1) where n is the number of predictors and m is is the number of training examples (rows/observations).
- **y**: Target vector. It is represented as a 2d NumPy array to match the dimension. The shape is (m, 1) where m is the number of training examples (rows/observations).
- Let’s inspect the datasets @Notebook 2_4_linear_regression_with_normal_equation.ipynb →

Gradient Descent vs Normal Equation

Gradient Descent

- Need to choose α
- Need many iterations
- Works well even when ***n*** or the features is large

Normal Equation

- No need to choose α
- Don't need to iterate
- Need to Compute $(X^T X)^{-1}$
- Slow if ***n*** or the features is very large

Advancing Regression

https://youtube.com/playlist?list=PLLssT5z_DsK-h9vYZkQkYNWcItqhlRJLN

Institut Teknologi Sepuluh Nopember
IEEE Student Branch

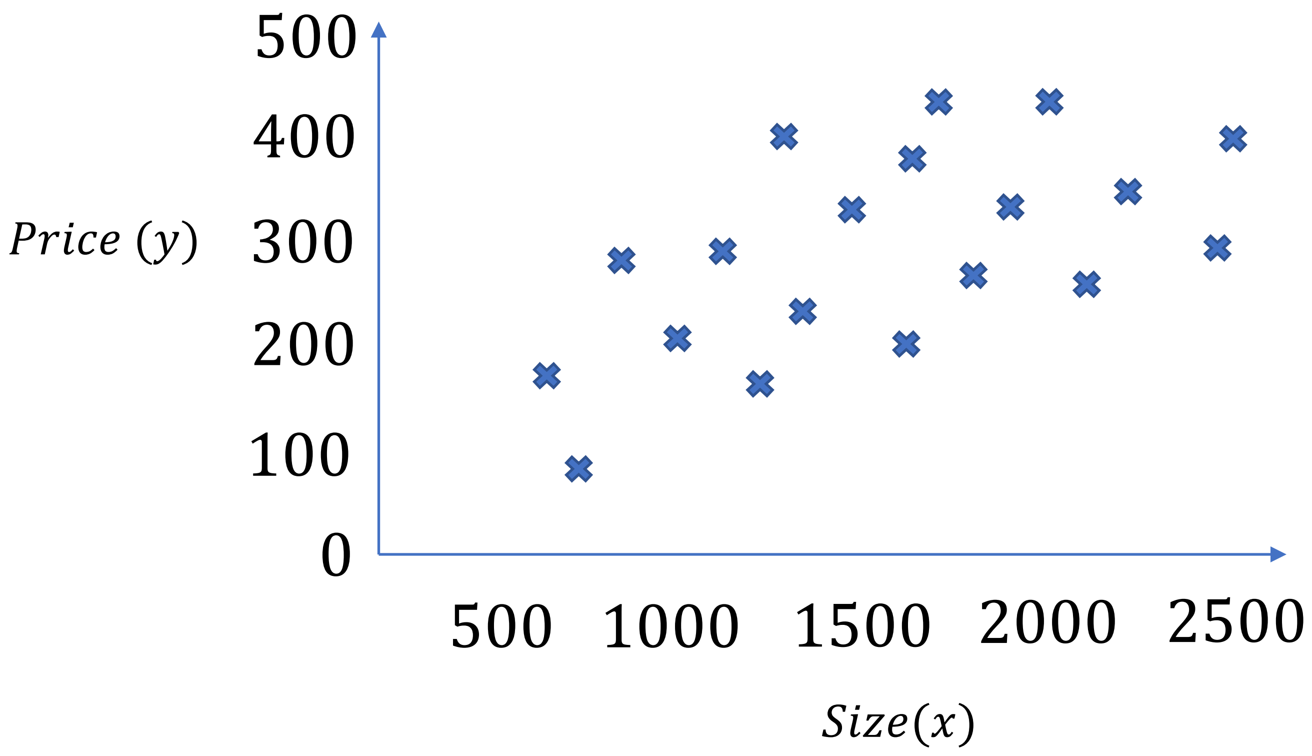
Gradient Descent with Multiple Features

Size (Feet) **x1**	Number of Bed **x2**	Number of Floor **x3**	Years **x4**	Price **y**
2104	5	1	45	460
1416	4	2	40	232
1534	3	2	30	315
…	…	…	…	…

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n$$

$$x = \begin{bmatrix} x_0 \\ x_1 \\ \dots \\ x_n \end{bmatrix} \qquad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \dots \\ \theta_n \end{bmatrix} \Rightarrow \qquad h_\theta(x) = \theta^T x$$

Polynominal Linear Regression



$$= \theta_0 + \theta_1 x + \theta_2 x^2$$

$$= \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$$

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_n x_n$$

$$= \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$$

Institut Teknologi Sepuluh Nopember
IEEE Student Branch

Feature Scaling

Size (Feet) **x1**	Number of Bed **x2**	Number of Floor **x3**	Years **x4**	Price **y**
2104	5	1	45	460
1416	4	2	40	232
1534	3	2	30	315
…	…	…	…	…

$$x_1 = Size\ (0\ - 2000)$$

$$x_2 = number\ of\ bedrooms$$

- Mean Normalization

$$x = \frac{x\ -\ \mu}{S}$$

$$0\ \leq x\ \leq 1$$

LATIHAN

@Notebook
2_1_data_processing_life_expectancy.ipynb →