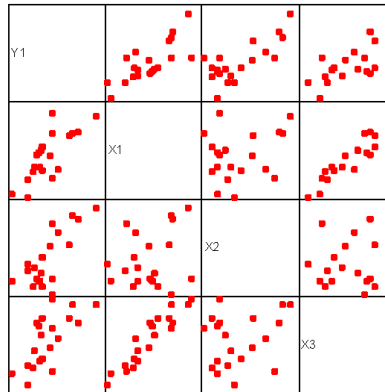


Multivariate Regression



1. Notation

- m : number of training examples
- n : number of features
- x : input variable
- y : output variable
- $x^{(i)}$: input features of i^{th} training example
- $x_j^{(i)}$: value of feature j in i^{th} training example
- $y^{(i)}$: output of i^{th} training example
- $(x^{(i)}, y^{(i)})$: i^{th} training example

x_0	x_1 : size	x_2 : # of bedrooms	x_3 : # of floors	x_4 : age of house	y
1	2104	5	1	45	460
1	1416	3	2	40	232

- This example shows a training data with:
 - 2 training examples
 - 4 features (by convention, there will always be feature 0 & it will always be 1)
 - $x_3^{(2)}$: feature 3 of the second training example, which is 2

2. Hypothesis

- Multivariate linear regression equation:

$$h_{\theta}(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n$$

- Note: we can write $h_{\theta}(x)$ as following:

$$x = \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{pmatrix} \in \mathbb{R}^{n+1} \quad \theta = \begin{pmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{pmatrix} \in \mathbb{R}^{n+1}$$

Multivariate Regression

$$h_{\theta}(x) = (\theta)^T x = (\theta_0 \quad \theta_1 \quad \dots \quad \theta_n) \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{pmatrix}$$

- Polynomial regression equation (some examples):

$$h_{\theta}(x) = \theta_0 x_0 + \theta_1 x_1^1 + \theta_2 x_2^2 + \dots + \theta_n x_n^n$$

$$h_{\theta}(x) = \theta_0 x_0 + \theta_1 x_1^{\frac{1}{2}} + \theta_2 x_2^{\frac{1}{4}} + \dots + \theta_n x_n^{\frac{1}{2n}}$$

3. Cost Functions

- Squared error (SE) cost function:

$$\begin{aligned} J(\theta_0, \theta_1) &= \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2 \\ &= \frac{1}{2m} \sum_{i=1}^m ((\theta)^T x^{(i)} - y^{(i)})^2 \\ &= \frac{1}{2m} \sum_{i=1}^m \left(\left(\sum_{j=0}^n \theta_j x_j^{(i)} \right) - y^{(i)} \right)^2 \end{aligned}$$

4. Gradient Descent Algorithm Using SE Loss Function for Multivariate Regression

Repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)}) (x_j^{(i)}) \quad (for \ j = 0, \dots, n)$$

}

- Note:

$$\frac{\partial}{\partial \theta_j} J(\theta) = \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)}) (x_j^{(i)})$$

5. Feature Scaling

- Make sure that the features are on similar scale (i.e. Data normalization)
- Feature scaling helps making the contour plot of the cost function less skewed. Thus, helping the gradient descent move through a much direct path
- Here are some formulas used for features scaling:

Multivariate Regression

$$x_i = \frac{x_i - \bar{x}}{\max(x)}$$

$$x_i = \frac{x_i - \bar{x}}{\max(x) - \min(x)}$$

$$x_i = \frac{x_i - \bar{x}}{s}$$

- Note: we don't apply feature scaling to feature 0 (i.e. x_0) because by convention, it will always be 1

6. Normal Equation

- A method to solve for θ analytically
- We can apply the following formula to solve for θ :

$$\theta = (X^T X)^{-1} X^T Y$$

- Example:

x_0	x_1 : size	x_2 : # of bedrooms	x_3 : # of floors	x_4 : age of house	y
1	2104	5	1	45	460
1	1416	3	2	40	232
1	1534	3	2	30	315
1	852	2	1	36	178

Construct a matrix X and Y from the data:

$$X = \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix} \quad Y = \begin{bmatrix} 460 \\ 232 \\ 315 \\ 178 \end{bmatrix}$$

From here, we can solve for θ ...

- Note:
 - Using normal equation method does not require us to do feature scaling.
 - There are some pros and cons of normal equation method

Gradient descent	Normal equation
+ Need to choose α	+ No need to choose α
+ Need many iterations	+ Don't need to iterate
+ Works well even when n is large	+ Slow when n is large because it needs to compute the inverse of $X^T X$ ($\sim O(n^3)$)
	+ $X^T X$ may not be invertible

Multivariate Regression

- If the matrix is not invertible, we can try:
 - Pseudoinverse the matrix (?)
 - Remove redundant features (linearly dependent features)
 - Have fewer features (Could affect the correctness)

7. Note

- In multivariate regression, two problems may arise:
 - Overfitting: caused by adding too many independent variables; they account for more variance but add nothing to the model
 - Multicollinearity: happens when some/all of the independent variables (x_1, x_2, \dots) are correlated with each other → we want to eliminate multicollinearity.