Multiple Linear Regression

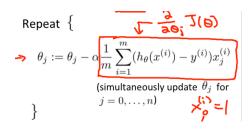
- Algorithm implemented in Octave
- -- Simple data were applied with the algorithm
- -- Same data were analyzed in R
- -- Same data were analyzed in Python

Algorithm:

Hypothesis:
$$\underline{h_{\theta}(x) = \theta^T x = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n}$$
 Parameters:
$$\underline{\theta_0, \theta_1, \dots, \theta_n}$$

$$\underline{\theta_0, \theta_1, \dots, \theta_n}$$

$$\underline{h+t-divestad}$$
 Vector Cost function:
$$\underline{J(\theta_0, \theta_1, \dots, \theta_n)} = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$



Gradient descent in practice I: Feature Scaling

Idea: Make sure features are on a similar scale.

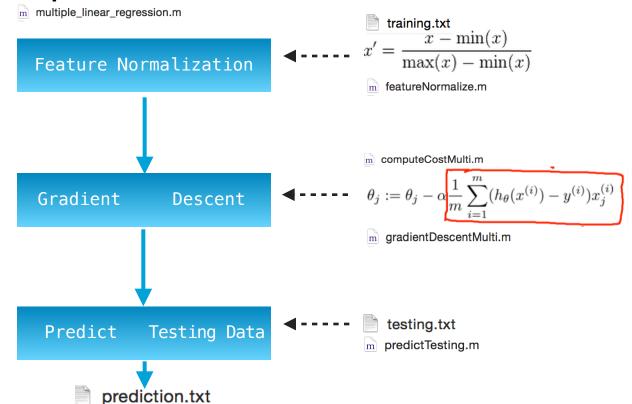
Gradient descent in practice II: Learning rate

- If α is too small: slow convergence.
- If α is too large: $J(\theta)$ may not decrease on every iteration; may not converge. (Slow)

To choose α , try

$$\dots,\underbrace{0.001}_{\checkmark}, \underbrace{\circ \circ \circ}_{\checkmark}, \underbrace{0.01}_{\checkmark}, \underbrace{\circ \circ \circ}_{\checkmark}, \underbrace{0.1}_{\checkmark}, \underbrace{\circ \circ}_{\checkmark}, \underbrace{1}_{\checkmark}, \dots$$

Implemented in Octave



Data Analyzed in R

File Name:

multiple_linear_regression.R

Usage:

/usr/bin/Rscript multiple_linear_regression.R training.txt testing.txt prediction.txt

Core Functions{Packages} Used:

build model
Im{stats}
make prediction on testing data

Data Analyzed in Python

File Name:

multiple_linear_regression.py

Usage:

python multiple_linear_regression.py training.txt testing.txt prediction.txt

Core Functions (Modules) Used:

build model linear_model.LinearRegression().fit(){sklearn} # make prediction on testing data