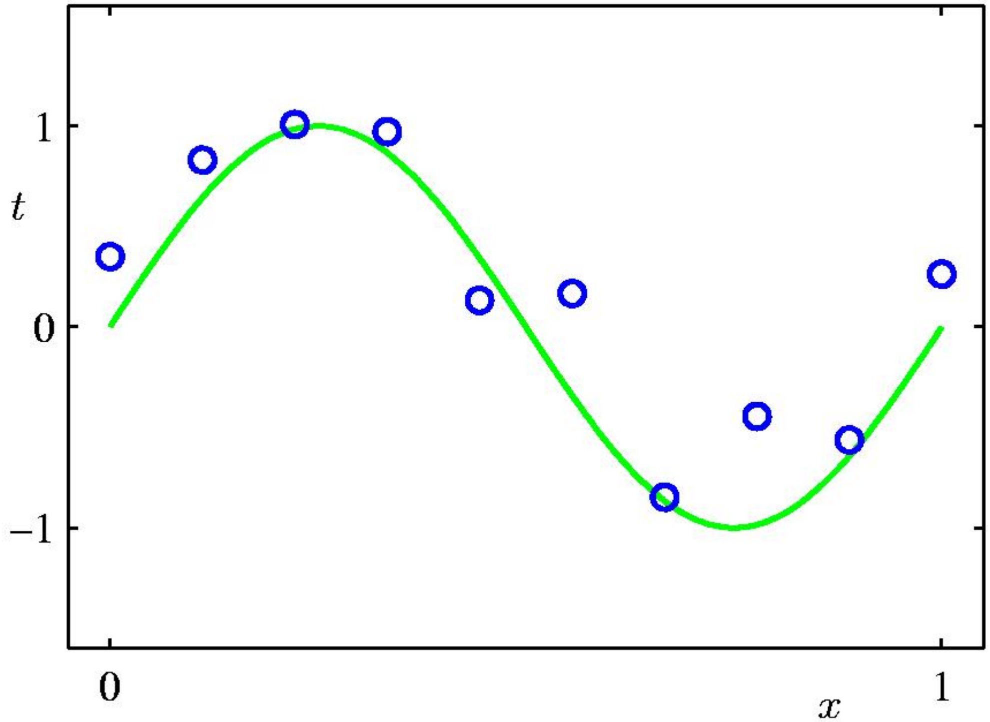
CPE/EE 695: Applied Machine Learning

Lecture 3 - 1: Linear Regression

Dr. Shucheng Yu, Associate Professor
Department of Electrical and Computer Engineering
Stevens Institute of Technology

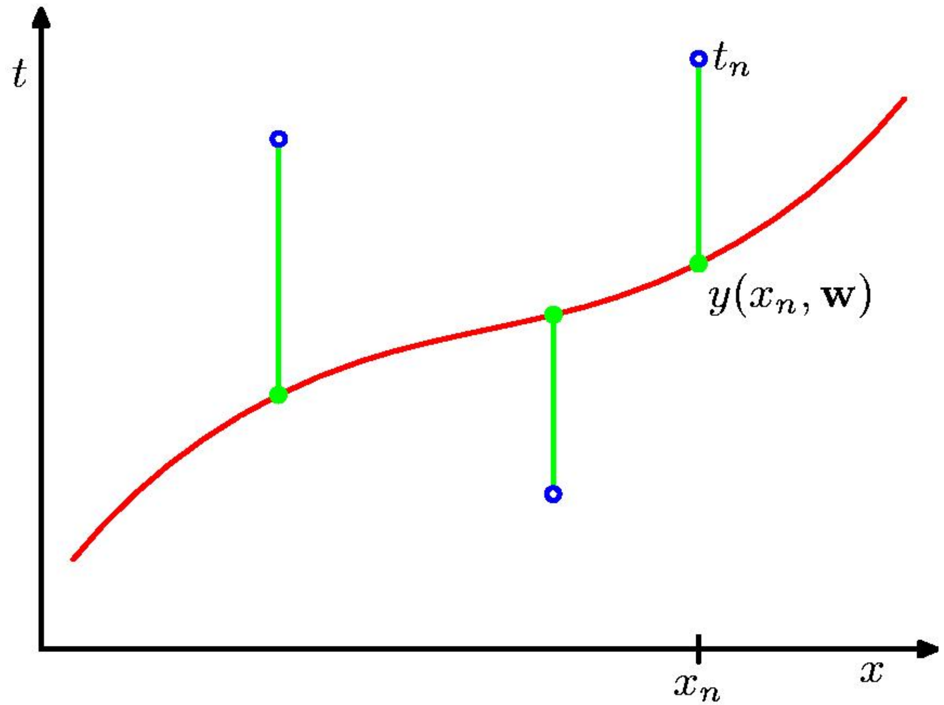
Polynomial Curve Fitting



$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

Polynomial Regression

Sum-of-Squares Error Function



$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2$$

Minimize E(w) for unknown w. (maximum likelihood)

Linear Regression

Linear model prediction:

$$\hat{y} = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n$$

Linear Regression

Linear model prediction:

$$\hat{y} = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n$$

Observed value of $\hat{y}$ would be:

$$y = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n + \varepsilon$$

Linear Regression

Linear model prediction:

$$\hat{y} = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n$$

Vectorized form:

$$\hat{y} = h_{\boldsymbol{w}}(\boldsymbol{x}) = \boldsymbol{w}^T \cdot \boldsymbol{x}$$

Linear Regression

Linear model prediction:

$$\hat{y} = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n$$

Vectorized form:

$$\hat{y} = h_{\boldsymbol{w}}(\boldsymbol{x}) = \boldsymbol{w}^T \cdot \boldsymbol{x}$$

Mean Square Error (MSE) cost function:

$$MSE(X, h_w) = \frac{1}{m} \sum_{i=1}^{m} (\boldsymbol{w}^T \cdot \boldsymbol{x}^{(\boldsymbol{i})} - \mathrm{y}^{(\mathrm{i})})^2$$

Linear Regression

Linear model prediction:

$$\hat{y} = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n$$

Vectorized form:

$$\hat{y} = h_{\boldsymbol{w}}(\boldsymbol{x}) = \boldsymbol{w}^T \cdot \boldsymbol{x}$$

Mean Square Error (MSE) cost function:

$$MSE(X, h_w) = \frac{1}{m} \sum_{i=1}^{m} (\boldsymbol{w}^T \cdot \boldsymbol{x}^{(\boldsymbol{i})} - \mathrm{y}^{(\mathrm{i})})^2$$

Normal Equation (closed-form solution):

$$\widehat{w} = (X^T \cdot X)^{-1} \cdot X^T \cdot \boldsymbol{y}$$

$\widehat{w}$: the value of that minimizes the cost function

X : the training data set

$\boldsymbol{y}$: the vector of target values containing $\mathrm{y}^{(1)}$ to $\mathrm{y}^{(\mathrm{m})}$

Linear Regression

Linear model prediction:

$$\hat{y} = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n$$

Vectorized form:

$$\hat{y} = h_{\boldsymbol{w}}(\boldsymbol{x}) = \boldsymbol{w}^T \cdot \boldsymbol{x}$$

Mean Square Error (MSE) cost function:

$$MSE(X, h_w) = \frac{1}{m} \sum_{i=1}^{m} (\boldsymbol{w}^T \cdot \boldsymbol{x}^{(\boldsymbol{i})} - \mathrm{y}^{(\mathrm{i})})^2$$

Normal Equation (closed-form solution):

$$\widehat{w} = (X^T \cdot X)^{-1} \cdot X^T \cdot \boldsymbol{y}$$

$\widehat{w}$: the value of that minimizes the cost function

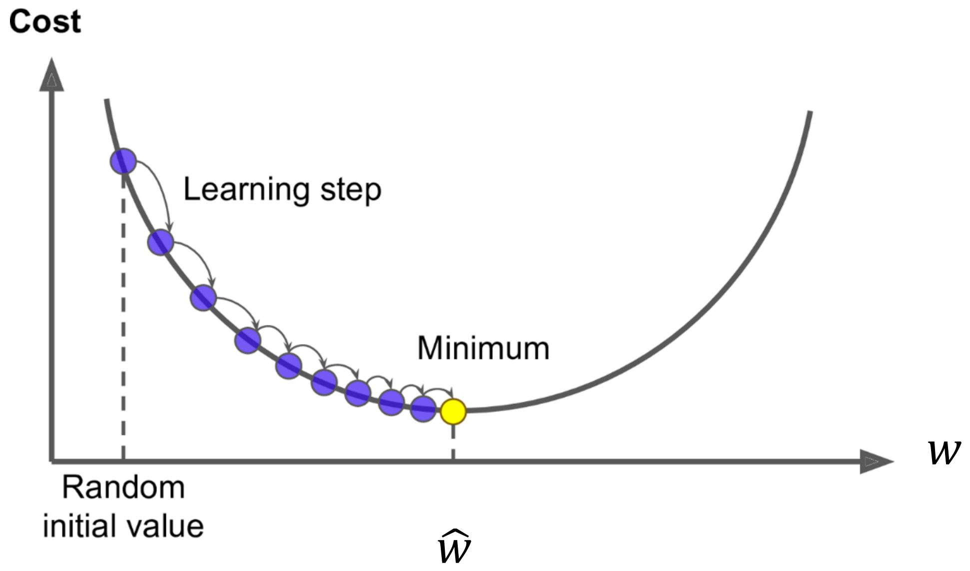
X : the training data set

$\boldsymbol{y}$: the vector of target values containing $\mathrm{y}^{(1)}$ to $\mathrm{y}^{(\mathrm{m})}$

Complexity: $O(n^3)$

Linear Regression

Gradient Descent:



$$w^{(next\ step)} = w - \alpha \cdot \nabla_w MSE(w)$$

What is Grade Descent?

A first-order iterative optimization algorithm to find the minimum of a multivariable function F(**x**).

Rational:

If F(**x**) is differentiable around a point A, F decreases **fastest** from A in the direction of negative gradient of F(x) at A (i.e., **-**$\boldsymbol{\nabla F(A)}$). In other words, let

$$\boldsymbol{A_{n+1} = A_n - \alpha * \nabla F(A)}$$

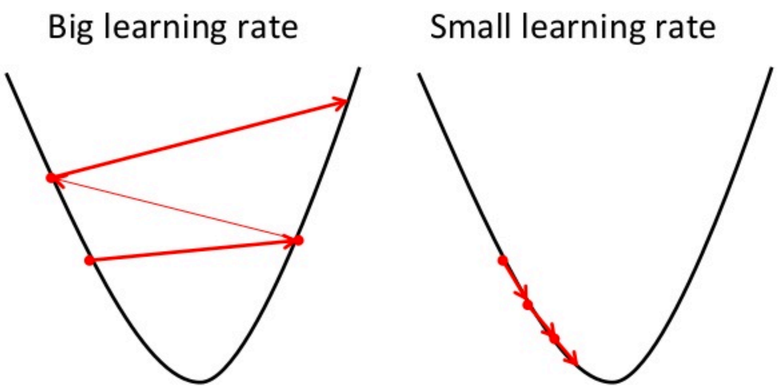
for small enough $\boldsymbol{\alpha}$, we have $\boldsymbol{F(A_{n+1}) \leq F(A_n)}$.

Through a set of such points $\boldsymbol{A_0, A_1}$, ..., it converges to a **local minimum**.

If function F(**x**) is **convex**, the local minimum is the **global minimum**.

Linear Regression

Gradient Descent:



Learning rate λ is very important

Linear Regression

Gradient Descent:

1) Batch Gradient Descent

   Using ALL data sets to calculate the gradient and update w:

   $$\frac{\partial}{\partial w_j} = \frac{2}{m}\sum_{i=1}^{m}\left(w^T \cdot x^{(i)} - y^{(i)}\right) {x_j}^{(i)}$$

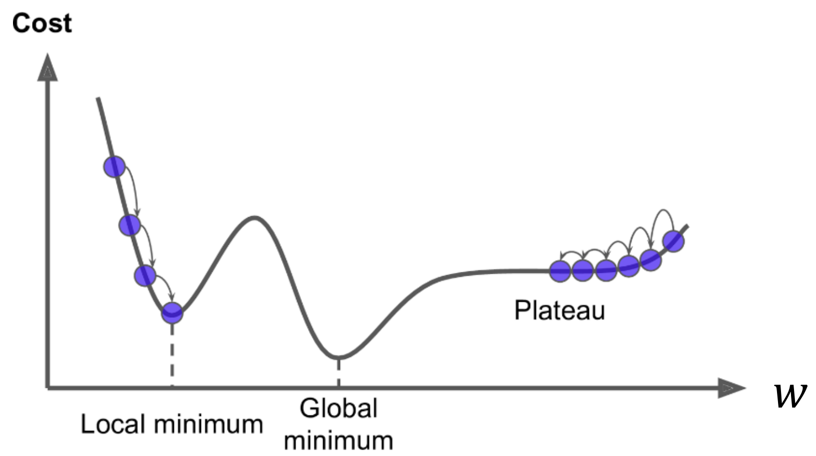
2) Stochastic Gradient Descent

   Using one RANDOM sample to calculate the gradient and update w.

3) Mini-batch Gradient Descent (using a small random set of samples)

Linear Regression

Gradient Descent:



Local minimum issue

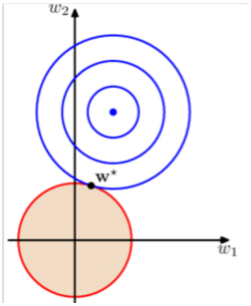
Batch Gradient Descent: converges fast, more likely to have local minimum
Stochastic Gradient Descent: converges slower, less likely to have local minimum
Mini-Batch Gradient Descent: in the middle of the two.

Regularized Linear Models

Ridge Regression

Cost function: $E(w) = MSE(w) + \frac{\alpha}{2}\sum_{i=1}^{m} {w_i}^2$

It has a closed-form solution $\mathbf{w} = \left(\lambda \mathbf{I} + \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1} \boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t}.$

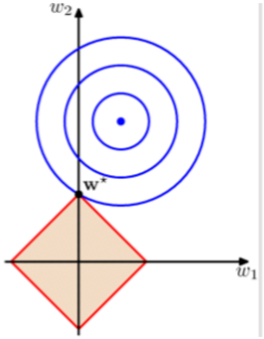


Lasso Regression

Cost function: $E(w) = MSE(w) + \alpha \sum_{i=1}^{m} |w_i|$

No closed-from solution for w;

But it tends to eliminate the weights of least important features.



Elastic Net

Cost function: $E(w) = MSE(w) + \lambda_1 \sum_{i=1}^{m} {w_i}^2 + \lambda_2 \sum_{i=1}^{m} |w_i|$

In the middle of Ridge and Lasso.

Usually is preferred over Lasso or Ridge.

Cost Functions

Mean Square Error (MSE):

$$MSE(X, h) = \frac{1}{m}\sum_{i=1}^{m}(h(\boldsymbol{x}^{(\boldsymbol{i})}) - \mathrm{y}^{(\mathrm{i})})^2$$

Root Mean Square Error (RMSE):

$$RMSE(X, h) = \sqrt{MSE}$$ (Euclidean norm)

Mean Absolute Error (MAE):

$$MAE(X, h) = \frac{1}{m}\sum_{i=1}^{m}\left|h\left(x^{(i)}\right) - y^{(i)}\right|$$ (Manhattan norm)

$\boldsymbol{l_k}$ ***norm*** of a vector v with n elements: $||v||_k = (|v_0|^k + \cdots + |v_n|^k)^{\frac{1}{k}}$

The higher the norm index, the more it focuses on large values and neglect small ones. Therefore, RMSE is more sensitive to outliers than MAE.

STEVENS
INSTITUTE of TECHNOLOGY
THE INNOVATION UNIVERSITY®

stevens.edu