Supervised Learning: **Regression**

Vinayaka Gude, Ph.D.
Elon University

Why use ML for regression?

Computational Complexity
Overfitting
Collinearity

Linear Regression

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

Predicted value

Model parameters

Features/inputs (n)

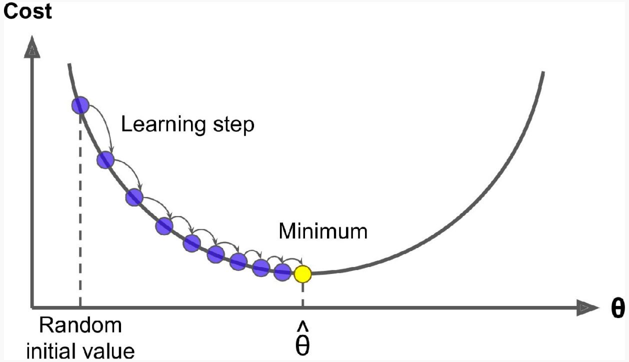
$$\hat{y} = h_{\boldsymbol{\theta}}(\mathbf{x}) = \boldsymbol{\theta} \cdot \mathbf{x}$$

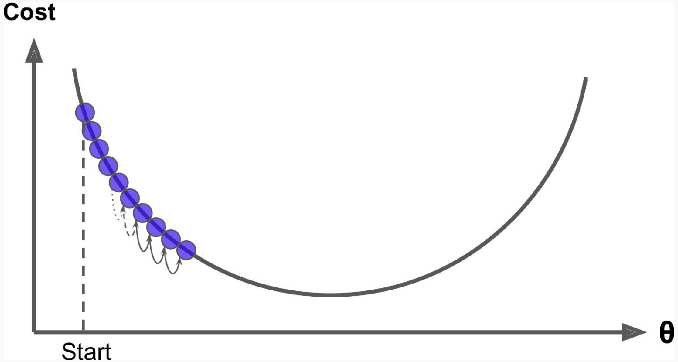
Dot product of model parameter & feature vectors

How do we develop a regression model? → Least Squares Method

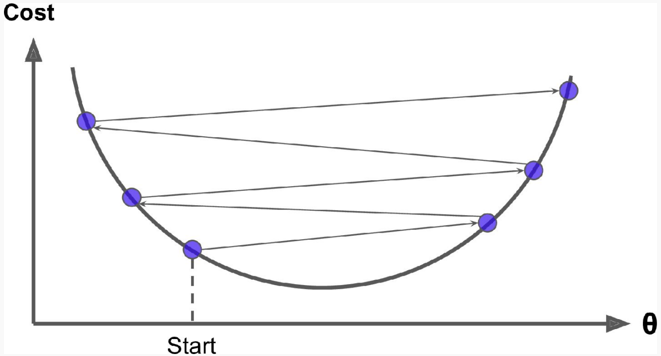
Normal Equation, $\widehat{\boldsymbol{\theta}} = \left(\mathbf{X}^\intercal \mathbf{X}\right)^{-1} \mathbf{X}^\intercal \mathbf{y}$

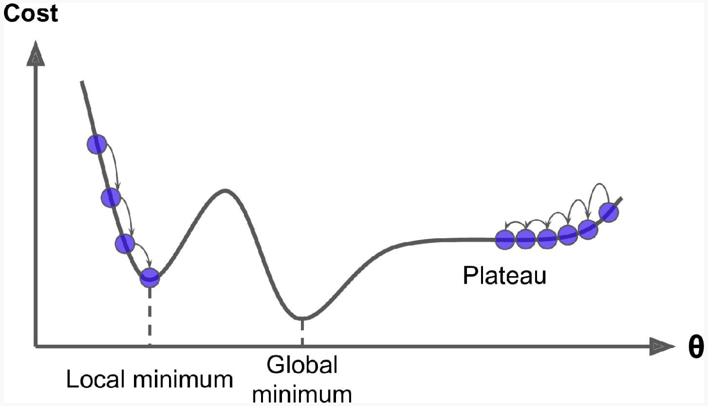
Gradient Descent





$$\frac{\partial}{\partial \theta_j} \mathrm{MSE}\left(\boldsymbol{\theta}\right) = \frac{2}{m} \sum_{i=1}^{m} \left(\boldsymbol{\theta}^\intercal \mathbf{x}^{(i)} - y^{(i)}\right) x_j^{(i)}$$



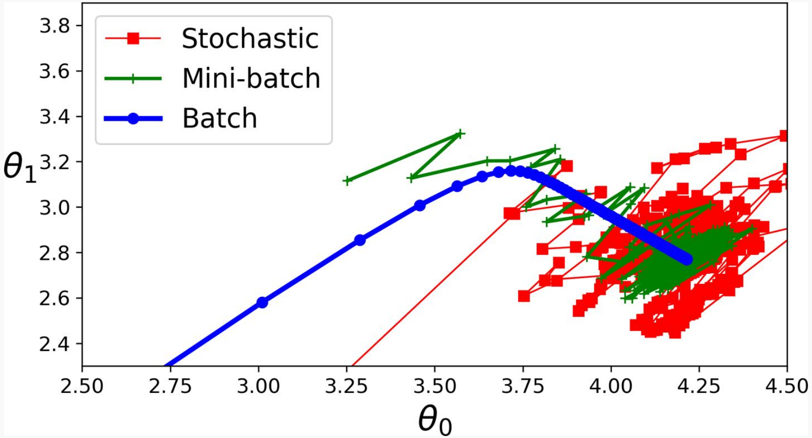


Gradient Descent approaches

Batch Uses all the training data in each iteration → **terribly slow**

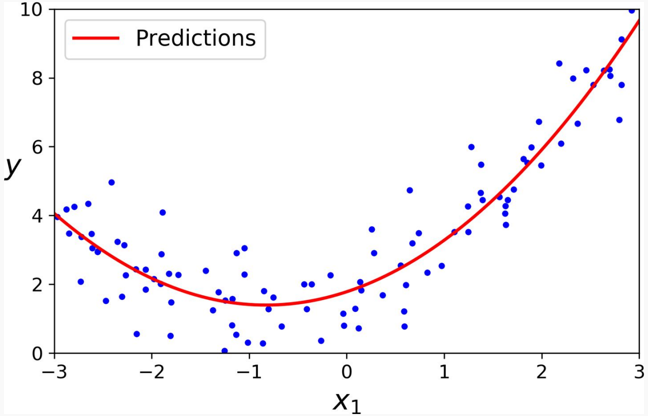
Stochastic Selects a random instance at every step and computes the gradient → **fastest but less regular**

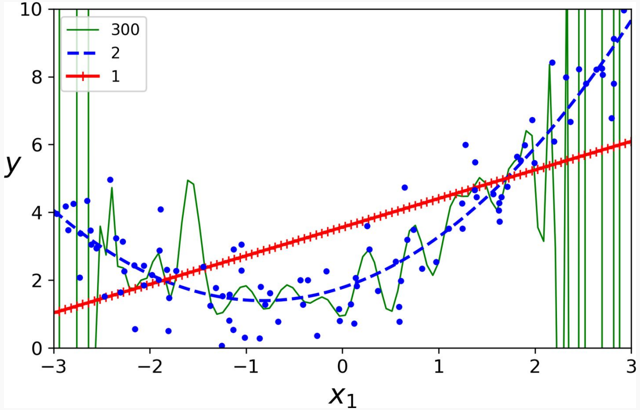
Mini-batch Uses mini batches → **balance b/w 2 approaches**



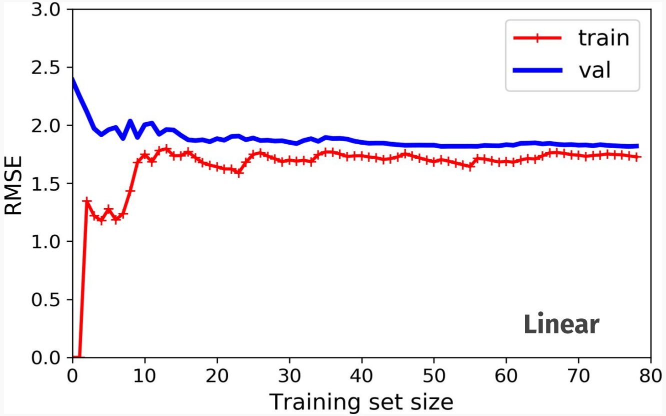
Polynomial Regression

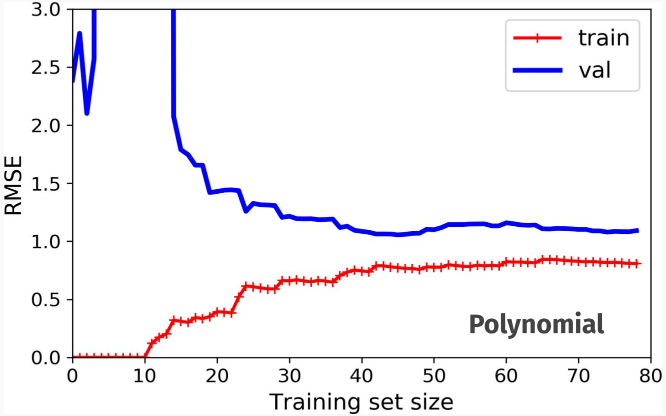
Useful when the data is too complex for a linear equation





Learning Curves





Generalization Error = **Bias** + **Variance** + **Irreducible Error**

Bias: wrong assumptions

Variance: Complexity of the model

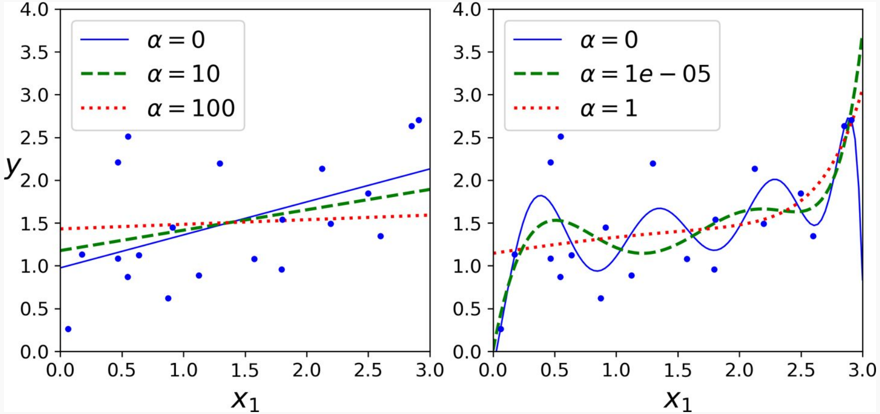
Irreducible Error: noise in the data

Regularized Regression - Ridge

Cost function, $J(\boldsymbol{\theta}) = \mathrm{MSE}(\boldsymbol{\theta}) + \boxed{\alpha \frac{1}{2} \sum_{i=1}^{n} {\theta_i}^2}$ **(L_2 regularization)**

The regularization term added to the cost function forces the learning algorithm to not only minimize the error but also **minimize the model parameters**

Overfitting can be reduced by constraining the model



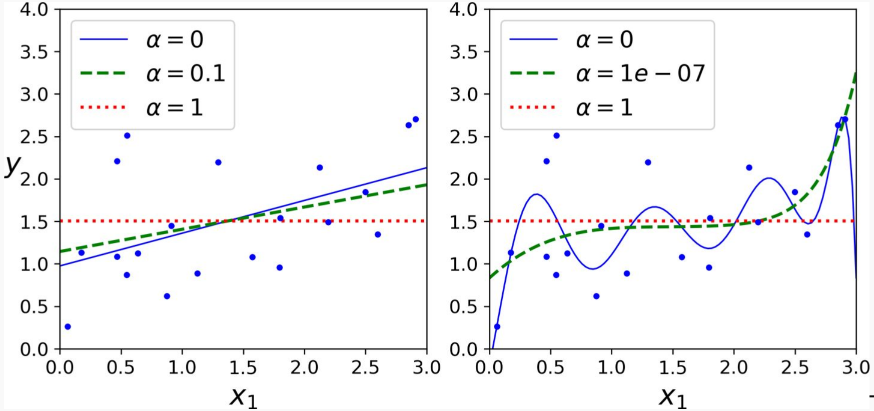
All input data has to be <u>standardized</u> before using Ridge Regression

Regularized Regression - Lasso

Least Absolute Shrinkage and Selection Operator Regression

Cost function, $$J(\boldsymbol{\theta}) = \mathrm{MSE}(\boldsymbol{\theta}) \boxed{+ \alpha \sum_{i=1}^{n} |\theta_i|}$$ **(L_1 regularization)**

Tends to eliminate the weights of the least important features → Lasso Regression automatically performs feature selection and outputs a sparse model



Regularized Regression - ElasticNet

Cost function, $J(\boldsymbol{\theta}) = \mathrm{MSE}(\boldsymbol{\theta}) + \boxed{r\alpha \sum_{i=1}^{n} |\theta_i| + \frac{1-r}{2}\alpha \sum_{i=1}^{n} {\theta_i}^2}$ **(L_1 + L_2 regularization)**

Balance between Ridge and Lasso

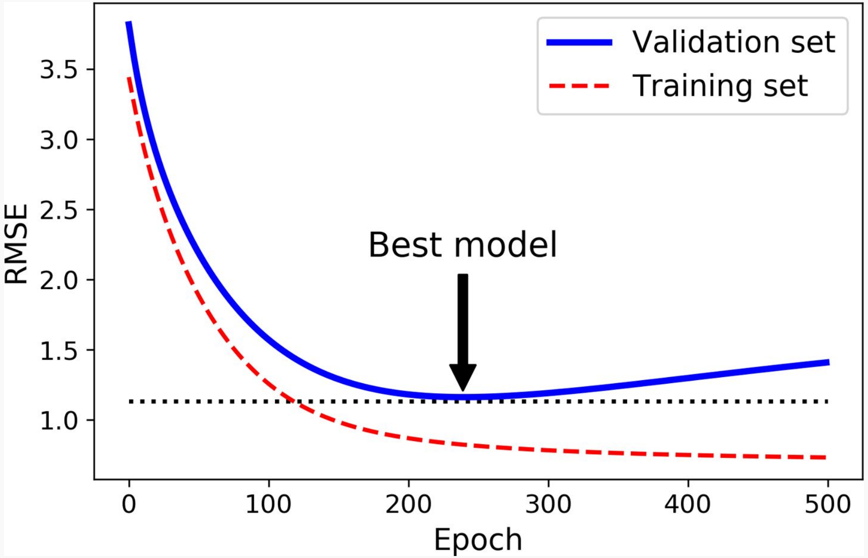
On most problems, some kind of regularization is always needed → **Avoid simple linear**

Ridge regression is a good default

Use Lasso or Elastic Net in presence of unimportant features

Early Stopping

Stop training once validation error reaches the minimum



Thank you!

Any questions?

gude.vinayaka@outlook.com