

Lecture 04: Regression

[SCS4049-02] Machine Learning and Data Science

Seongsik Park (s.park@dgu.edu)

AI Department, Dongguk University

Linear regression

Linear regression

Linear model

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n \quad (1)$$

In this equation,

- \hat{y} is the predicted value (for true y)
- n is the number of features
- x_i is the i -th feature value
- θ_i is the i -th model parameter (including the bias term θ_0 and the feature weights $\theta_1; \theta_2; \dots; \theta_n$)

Linear regression

This can be written much more concisely using a vectorized form,

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n \quad (2)$$

$$= \theta^T \mathbf{x} \quad (3)$$

In this equation,

- θ is the model's parameter vector, containing the bias term θ_0 and the feature weights θ_1 to θ_n
- \mathbf{x} is the instance's feature vector, containing θ_0 to always equal to 1
- $\theta^T \mathbf{x}$ is the dot product of the vectors θ and \mathbf{x} , which is of course equal to $\theta_0 \theta_0 + \theta_1 x_1 + \dots + \theta_n x_n$
- \hat{y} is the hypothesis function, using the model parameter

Linear regression

That's the linear regression model – but how do we train it?

Recall that training a model means setting its parameters so that the model best fits the training set.

We first need a measure of how well (or poorly) the model fits the training data.

The most common performance measure of a regression model is the Root Mean Square Error (RMSE).

$$RMSE(\hat{y}; y) = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (4)$$

Linear regression

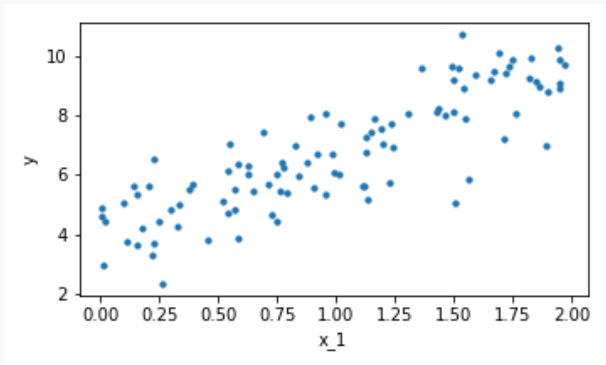


Figure 1: Linear regression: training dataset

Generating training dataset

$$y = \theta_0 + \theta_1 x_1 + \epsilon \quad (5)$$

$$= \theta_0 + \theta_1 x_1 + \epsilon \quad \text{where } \epsilon \sim N(0; \sigma^2) \quad (6)$$

Linear regression: design matrix

Design matrix (regressor matrix, model matrix, data matrix)

- (sample, example) m
- feature vector \ddagger n \dagger
- m sample row vector design matrix \dagger

$$\begin{matrix}
 & 2 & 3 \\
 & (1) & 7 \\
 \% \ddagger & \begin{pmatrix} 6 \\ 7 \\ 7 \\ 4 \end{pmatrix} & \begin{pmatrix} 7 \\ 7 \\ 7 \\ 5 \end{pmatrix} \\
 & (2) & \\
 & \vdots & \\
 & () &
 \end{matrix}
 =
 \begin{matrix}
 & 2 & 3 \\
 & 1 & 7 \\
 & 2 & 7 \\
 & \vdots & 7 \\
 & 4 & 5
 \end{matrix}
 \quad (7)$$

$$\begin{matrix}
 \dagger = \begin{pmatrix} 6 \\ 7 \\ 7 \\ 4 \end{pmatrix} = \begin{matrix} 2 & 3 \\ 1st\ sample & \sim^{(1)} \\ 2nd\ sample & \sim^{(2)} \\ \vdots & \vdots \\ m\text{-th\ sample} & \sim^{()} \end{matrix}
 \end{matrix}
 =
 \begin{matrix}
 \begin{pmatrix} 6 \\ 7 \\ 7 \\ 4 \end{pmatrix} = \begin{matrix} 2 & 3 \\ \sim^{(1)} & \sim^{(1)} \\ \sim^{(2)} & \sim^{(2)} \\ \vdots & \vdots \\ \sim^{()} & \sim^{()} \end{matrix}
 \end{matrix}
 \begin{matrix}
 \begin{pmatrix} 7 \\ 7 \\ 7 \\ 5 \end{pmatrix} = \begin{matrix} 2 & 3 \\ \sim^{(1)} & \sim^{(1)} \\ \sim^{(2)} & \sim^{(2)} \\ \vdots & \vdots \\ \sim^{()} & \sim^{()} \end{matrix}
 \end{matrix}
 \quad (8)$$

Linear regression: design matrix

$\hat{\beta} = \dagger \quad m \quad \text{sample}$

$$\begin{matrix} 2 \\ \vdots \\ 6 \\ \vdots \\ 4 \end{matrix} \begin{matrix} (1) \\ (2) \\ \ddots \\ \vdots \\ \vdots \end{matrix} \begin{matrix} 3 \\ 2 \\ 5 \end{matrix} = \begin{matrix} \dagger \\ \vdots \\ \vdots \\ \vdots \\ \vdots \end{matrix} \begin{matrix} 2 \\ \vdots \\ 4 \end{matrix} \begin{matrix} \ddagger \\ \ddagger \\ \ddagger \\ \ddagger \\ \ddagger \end{matrix} \quad \begin{matrix} \sim^{(1)} \\ \sim^{(2)} \\ \ddots \\ \vdots \\ \vdots \end{matrix} \begin{matrix} \sim^{(1)} \\ \sim^{(2)} \\ \ddots \\ \vdots \\ \vdots \end{matrix} \quad \begin{matrix} \ddot{z} \\ \ddot{z} \\ \ddot{z} \\ \ddot{z} \\ \ddot{z} \end{matrix} \quad \begin{matrix} \sim^{(1)} \\ \sim^{(2)} \\ \ddots \\ \vdots \\ \vdots \end{matrix} \begin{matrix} 3 & 2 \\ 2 & 6 \\ 2 & 6 \\ 5 & 4 \end{matrix} \begin{matrix} 1 \\ 2 \\ \ddots \\ \vdots \\ \vdots \end{matrix} \begin{matrix} 3 \\ 2 \\ 2 \\ 2 \\ 5 \end{matrix} \quad (9)$$

$$\begin{matrix} 2 \\ \vdots \\ 6 \\ \vdots \\ 4 \end{matrix} \begin{matrix} (1) \\ (2) \\ \ddots \\ \vdots \\ \vdots \end{matrix} \begin{matrix} 3 \\ 2 \\ 5 \end{matrix} = \begin{matrix} \dagger \\ \vdots \\ \vdots \\ \vdots \\ \vdots \end{matrix} \begin{matrix} 2 \\ \vdots \\ 4 \end{matrix} \begin{matrix} \ddagger \\ \ddagger \\ \ddagger \\ \ddagger \\ \ddagger \end{matrix} \quad \begin{matrix} \sim^{(1)} \\ \sim^{(2)} \\ \ddots \\ \vdots \\ \vdots \end{matrix} \begin{matrix} \sim^{(1)} \\ \sim^{(2)} \\ \ddots \\ \vdots \\ \vdots \end{matrix} \quad \begin{matrix} \ddot{z} \\ \ddot{z} \\ \ddot{z} \\ \ddot{z} \\ \ddot{z} \end{matrix} \quad \begin{matrix} \sim^{(1)} \\ \sim^{(2)} \\ \ddots \\ \vdots \\ \vdots \end{matrix} \begin{matrix} 3 & 2 \\ 2 & 6 \\ 2 & 6 \\ 5 & 4 \end{matrix} \begin{matrix} 1 \\ 2 \\ \ddots \\ \vdots \\ \vdots \end{matrix} \begin{matrix} 3 \\ 2 \\ 2 \\ 2 \\ 5 \end{matrix} \quad (10)$$

$$\begin{matrix} 2 \\ \vdots \\ 6 \\ \vdots \\ 4 \end{matrix} \begin{matrix} (1) \\ (2) \\ \ddots \\ \vdots \\ \vdots \end{matrix} \begin{matrix} 3 \\ 2 \\ 5 \end{matrix} = \begin{matrix} \dagger \\ \vdots \\ \vdots \\ \vdots \\ \vdots \end{matrix} \begin{matrix} 2 \\ \vdots \\ 4 \end{matrix} \begin{matrix} \ddagger \\ \ddagger \\ \ddagger \\ \ddagger \\ \ddagger \end{matrix} \quad \begin{matrix} \sim^{(1)} \\ \sim^{(2)} \\ \ddots \\ \vdots \\ \vdots \end{matrix} \begin{matrix} \sim^{(1)} \\ \sim^{(2)} \\ \ddots \\ \vdots \\ \vdots \end{matrix} \quad \begin{matrix} \ddot{z} \\ \ddot{z} \\ \ddot{z} \\ \ddot{z} \\ \ddot{z} \end{matrix} \quad \begin{matrix} \sim^{(1)} \\ \sim^{(2)} \\ \ddots \\ \vdots \\ \vdots \end{matrix} \begin{matrix} 3 & 2 \\ 2 & 6 \\ 2 & 6 \\ 5 & 4 \end{matrix} \begin{matrix} 1 \\ 2 \\ \ddots \\ \vdots \\ \vdots \end{matrix} \begin{matrix} 3 \\ 2 \\ 2 \\ 2 \\ 5 \end{matrix} \quad (11)$$

Normal equation: geometric approach

Design matrix \mathbf{X}

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \quad (12)$$

$$\mathbf{y} = \beta_1 \mathbf{x}_1 + \beta_2 \mathbf{x}_2 + \dots + \beta_k \mathbf{x}_k + \boldsymbol{\epsilon} \quad (13)$$

$\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k$ hyperplane

Residual error $\boldsymbol{\epsilon}$ (orthogonal) \perp hyperplane

Normal equation: geometric interpretation

Residual error vector \mathbf{r} is orthogonal to the column space of \mathbf{X} . The normal vector to the hyperplane is $\mathbf{X}^T \mathbf{r}$.

$$\mathbf{X}^T \mathbf{r} = 0 \quad (14)$$

m sample

$$\mathbf{X}^T (\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}) = 0 \quad \Leftrightarrow \quad \mathbf{X}^T \mathbf{X} \hat{\boldsymbol{\beta}} = \mathbf{X}^T \mathbf{y} \quad (15)$$

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} \quad (16)$$

Normal equation: geometric interpretation

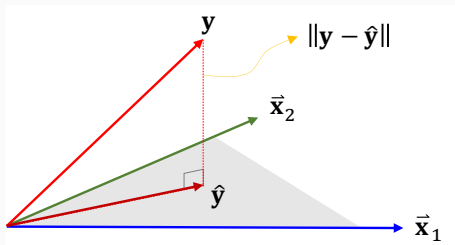


Figure 2: Normal equation: geometric interpretation

Normal equation

$$\hat{\beta} = (X^T X)^{-1} X^T y \quad (17)$$

Projection matrix

$$P = X (X^T X)^{-1} X^T \quad (18)$$

Normal equation: analytic approach

Sum of squared error (SSE)

$$rrB = \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (19)$$

$$= k\%o\%o\%o^2 = (\%o\%o\%o)(\%o\%o\%o) \quad (20)$$

Using $\%o\%o\%o$ †

$$rrB(\%o\%o\%o) = \%o\%o\%o\%o\%o\%o + (\%o\%o\%o) (\%o\%o\%o) = \%o\%o\%o\%o\%o\%o + \%o\%o\%o\%o\%o\%o \quad (21)$$

$$\frac{\partial rrB(\%o\%o\%o)}{\partial \%o\%o\%o} = 2(\%o\%o\%o\%o\%o\%o + \%o\%o\%o\%o\%o\%o) = \%o\%o\%o\%o\%o\%o \quad (22)$$

Hence, we have

$$\hat{\%o\%o\%o} = (\%o\%o\%o)^{-1} \%o\%o\%o\%o\%o\%o \quad (23)$$

$$\%o\%o\%o\%o\%o\%o = (\%o\%o\%o)^{-1} \%o\%o\%o\%o\%o\%o \quad (24)$$

Linear regression: cost function

We need to find the value of β that minimizes the RMSE. In practice, it is simpler to minimize the sum of squared error (SSE) than the MSE or the RMSE.

$$J(\beta) = \frac{1}{2} \sum_{i=1}^n (y_i - \beta^T x_i)^2 \quad (25)$$

$$J(\beta) = \frac{1}{2} \sum_{i=1}^n (y_i - \beta_0 - \beta_1 x_i)^2 \quad (26)$$

$$J(\beta) = \sum_{i=1}^n (y_i - \beta_0 - \beta_1 x_i)^2 \quad (27)$$

Linear regression: closed from solution

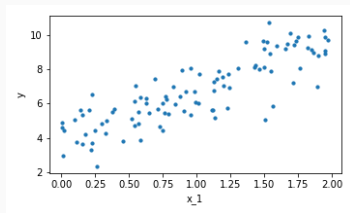


Figure 3: Linear regression: training dataset

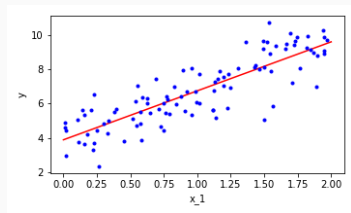


Figure 4: Linear regression: closed form solution

Linear regression: computational complexity

Normal equation

- Normal equation

$$\theta_0 = (X^T X)^{-1} X^T y \quad (28)$$

- $X \in \mathbb{R}^{n \times 2}$
- $X^T X \in \mathbb{R}^{2 \times 2}$
- $(X^T X)^{-1} = \mathcal{O}(2^3) = \mathcal{O}(8)$
- Feature

Gradient descent

Gradient descent

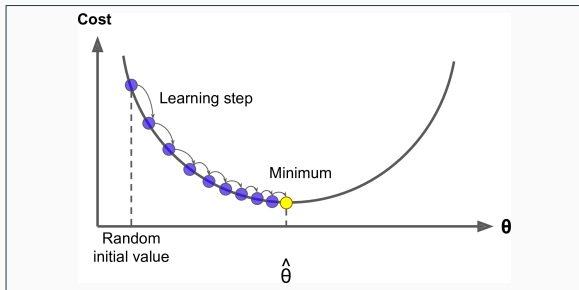


Figure 4-3. Gradient Descent

Gradient descent

Derivative

- $\theta(\cdot)$
- $\theta(\cdot)$
- : scalar / scalar

$$\theta(\cdot) = \frac{(\cdot)}{\cdot} \quad (29)$$

Gradient

- Derivative
- \mathbf{x}
- : vector / scalar

$$r(\ddagger) = \frac{\partial(\ddagger)}{\partial \ddagger} = \begin{pmatrix} 2 \\ 3 \\ 4 \\ \vdots \\ 5 \end{pmatrix} \begin{matrix} @ \\ @_1 \\ @_2 \\ @ \\ @ \end{matrix} \quad (30)$$

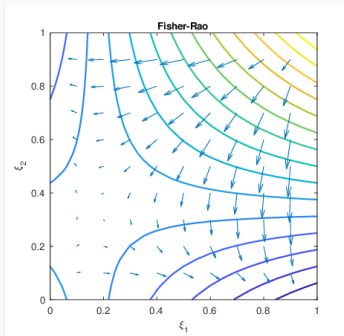


Figure 5: Gradient field

Gradient descent

Find the value of x where $f(x)$ is minimum (local or global)

- Find the value x at which $f'(x) = 0$

$$f'(x) = 0 \quad (31)$$

- Sure it's minimum?

$$\frac{f''(x)}{2} > 0 \quad (32)$$

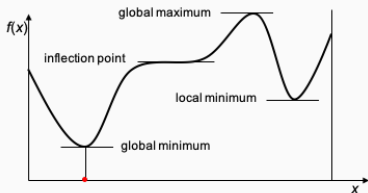


Figure 6: Local and global minimum

Gradient descent: optimization method

- (cost function, loss function)
 - () (ground truth)
 - utility,
 - e.g., sum of squared error (SSE), cross-entropy
- (learning algorithm)
 - (training data)
- - normal equation close-form solution
 -

Gradient descent: method

$\hat{w}_j =$

SSE cost

$$J(\theta) = \sum_{i=1}^n (y^{(i)} - \hat{y}^{(i)})^2 \quad (33)$$

$$J'(\theta) = 2 \sum_{i=1}^n (y^{(i)} - \hat{y}^{(i)}) \cdot (-x^{(i)}) \quad (34)$$

$$J'(\theta) = -2 \sum_{i=1}^n (y^{(i)} - \hat{y}^{(i)}) x^{(i)}$$

$$\theta_j^1 = \theta_j^0 - \alpha J'(\theta_j^0) \quad (35)$$

$$\theta_j^2 = \theta_j^1 - \alpha J'(\theta_j^1) \quad (36)$$

$$\tilde{z} = \tilde{z} \quad (37)$$

$$J(\theta) = J(\theta^{(j-1)}) - \alpha \sum_{i=1}^n (y^{(i)} - \hat{y}^{(i)}) x^{(i)}$$

(learning rate) update

hyperparameter

()

Gradient descent: learning rate

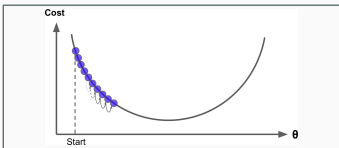


Figure 4-4. Learning rate too small

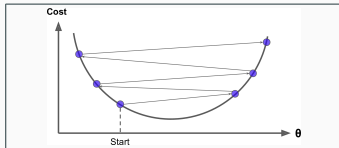


Figure 4-5. Learning rate too large

- (iteration)
- (iteration)
- Gradient vector (norm) (tolerance)
- convex batch gradient descent
- $\zeta(1/)$
- tolerance 1/10
- 10

Batch gradient descent

Linear regression model θ

$$J(\theta) = \frac{1}{2} \sum_{i=1}^n (y_i - \theta^T x_i)^2 \quad (38)$$

$$\frac{\partial J(\theta)}{\partial \theta_j} = \sum_{i=1}^n (y_i - \theta^T x_i) x_{ij} \quad (39)$$

$$\frac{\partial J(\theta)}{\partial \theta} = \begin{bmatrix} \frac{\partial J(\theta)}{\partial \theta_1} \\ \frac{\partial J(\theta)}{\partial \theta_2} \\ \vdots \\ \frac{\partial J(\theta)}{\partial \theta_n} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^n (y_i - \theta^T x_i) x_{i1} \\ \sum_{i=1}^n (y_i - \theta^T x_i) x_{i2} \\ \vdots \\ \sum_{i=1}^n (y_i - \theta^T x_i) x_{in} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^n (y_i - \theta^T x_i) x_{i1} \\ \sum_{i=1}^n (y_i - \theta^T x_i) x_{i2} \\ \vdots \\ \sum_{i=1}^n (y_i - \theta^T x_i) x_{in} \end{bmatrix} \quad (40)$$

Gradient descent step

$$\theta^{(k+1)} = \theta^{(k)} - \alpha \frac{\partial J(\theta)}{\partial \theta} \quad (41)$$

where k is iteration number and $\theta^{(0)}$ is arbitrary initial value

Batch gradient descent

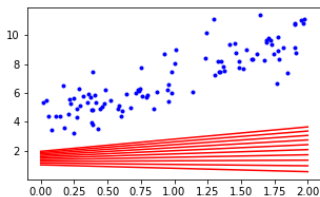
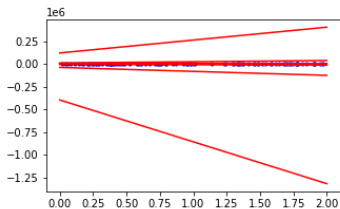
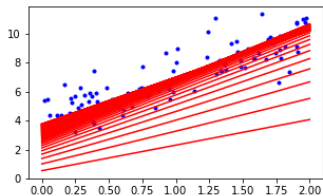


Figure 7: Learning rate = 0.001, 0.01, 0.0001

Batch gradient descent: computational complexity

Batch gradient descent algorithm

- $\mathcal{O}(n^2)$ batch
- $\mathcal{O}(n)$
- Normal equation: feature
- Gradient descent: feature

Learning rate

- Hyperparameter α (learning rate)
- α

Learning schedule

- Constant learning rate

- 0.1, 0.01

- Time-based decay

$$\eta = \frac{\eta_0}{(1 + \beta t)} \quad (42)$$

η_0 : β : hyperparameter, t : iteration

- Step decay

- 0.1 epoch

- 0.05, 5 epoch

- Epoch:

β 20 epoch 1/10

= 0.05 epoch

- Exponential decay

$$\eta = \eta_0 e^{-\beta t} \quad (43)$$

η_0 : β : hyperparameter, t : iteration

Stochastic gradient descent

For our linear regression model $\hat{y} = \mathbf{x}^T \boldsymbol{\beta}$

$$J(\boldsymbol{\beta}) = \frac{1}{2} \sum_{i=1}^n (y_i - \mathbf{x}_i^T \boldsymbol{\beta})^2 = \sum_{i=1}^n J_i(\boldsymbol{\beta}) \quad (44)$$

$$\boldsymbol{\beta}^{(t+1)} = \boldsymbol{\beta}^{(t)} - \eta \nabla J_i(\boldsymbol{\beta}^{(t)}) \quad (45)$$

- sample gradient
- parameter update
- sequential learning or online learning
-
-
- BGD local optimum
-
- BGD global optimum

Mini-batch gradient descent

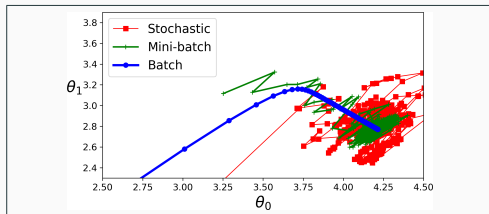


Figure 4-11. Gradient Descent paths in parameter space

- gradient
- 100,000 = (mini-batch size 100) (1,000 mini-batches)
- Batch gradient descent stochastic gradient descent(SGD)
- SGD
- SGD local minimum
- GPU

Linear regression comparison

Table 4-1. Comparison of algorithms for Linear Regression

Algorithm	Large m	Out-of-core support	Large n	Hyperparams	Scaling required	Scikit-Learn
Normal Equation	Fast	No	Slow	0	No	n/a
SVD	Fast	No	Slow	0	No	LinearRegression
Batch GD	Slow	No	Fast	2	Yes	SGDRegressor
Stochastic GD	Fast	Yes	Fast	≥ 2	Yes	SGDRegressor
Mini-batch GD	Fast	Yes	Fast	≥ 2	Yes	SGDRegressor



There is almost no difference after training: all these algorithms end up with very similar models and make predictions in exactly the same way.

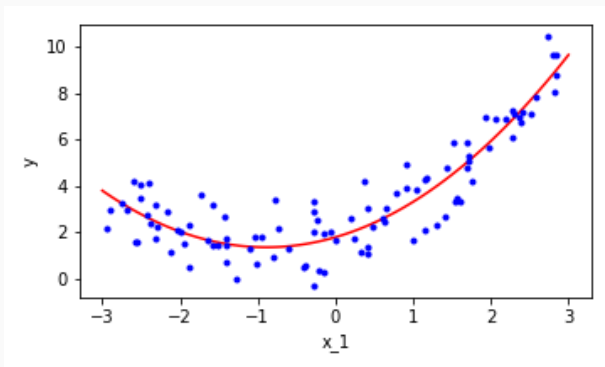
Polynomial regression

Polynomial regression

Polynomial regression

-
- polynomial feature
- x_1 polynomial

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_1^2 \quad (46)$$



Polynomial regression: closed form solution

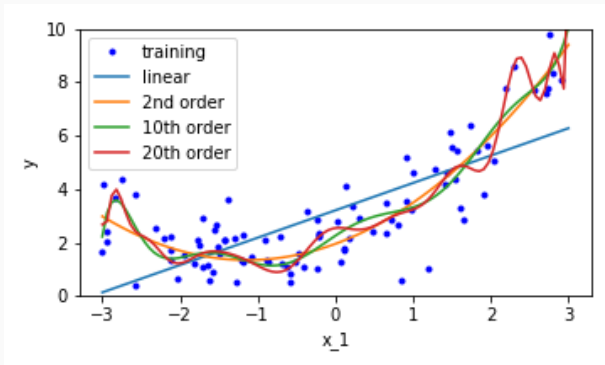


Figure 9: Polynomial regression: closed form solution

Learning curves: model selection

Learning curves

- error
- validation error
- error

training

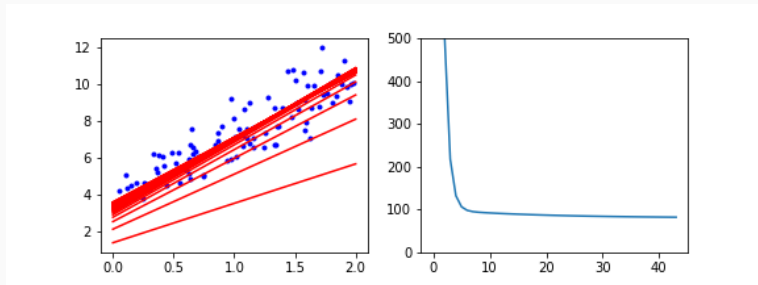


Figure 10: Linear regression with learning curve (SSE)

Learning curves: model selection

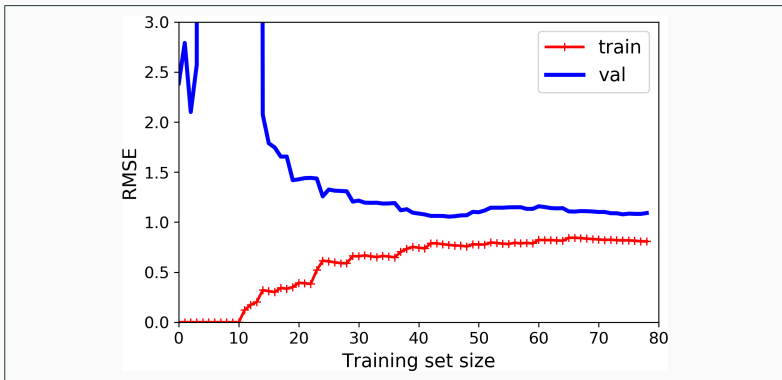


Figure 4-16. Learning curves for the polynomial model

Regularized linear model

Ridge regression

SSE

model parameter

penalty

•

model parameter

$$j(\beta) = \text{SSE}(\beta) + \frac{\alpha}{2} \sum_{k=1}^p \beta_k^2 \quad (47)$$

• Hyperparameter

cost

• Closed form solution $\hat{\beta} = (\mathbf{X}^T \mathbf{X} + \alpha \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y}$

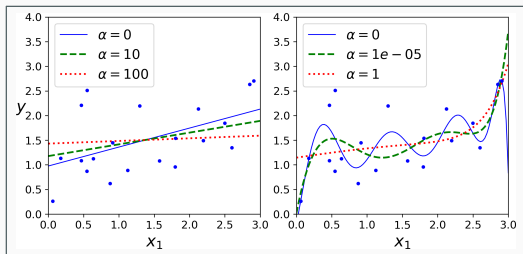


Figure 4-17. Ridge Regression

Lasso regression

SSE

model parameter

penalty

- Least Absolute Shrinkage and Selection Operator Regression (LASSO)

- feature parameter 0

$$j(\beta) = \text{SSE}(\beta) + \sum_{j=1}^p \alpha |\beta_j| \quad (48)$$

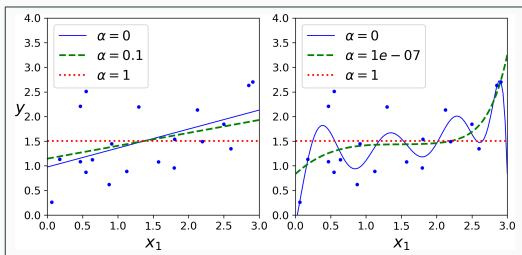


Figure 4-18. Lasso Regression

Appendix

- “Chap 4” of A. Geron, Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow

HW #1

Due: 9 20 † 23 59

- pdf ž
- (Adobe scan, Office lens) pdf ž
- (latex, word, ppt,) ž pdf ž
- ž

28 Figure 9 ž

1. 2 polynomial 200 ž (10)
2. Closed-form 1, 2, 10, 20 ž (10)
3. batch gradient descent ž (10)
4. BGD iteration SSE ž
(10)