REGRESSION

Regression

Relationships among several quantities

build a model that predicts the value of one variable as a function of other variables

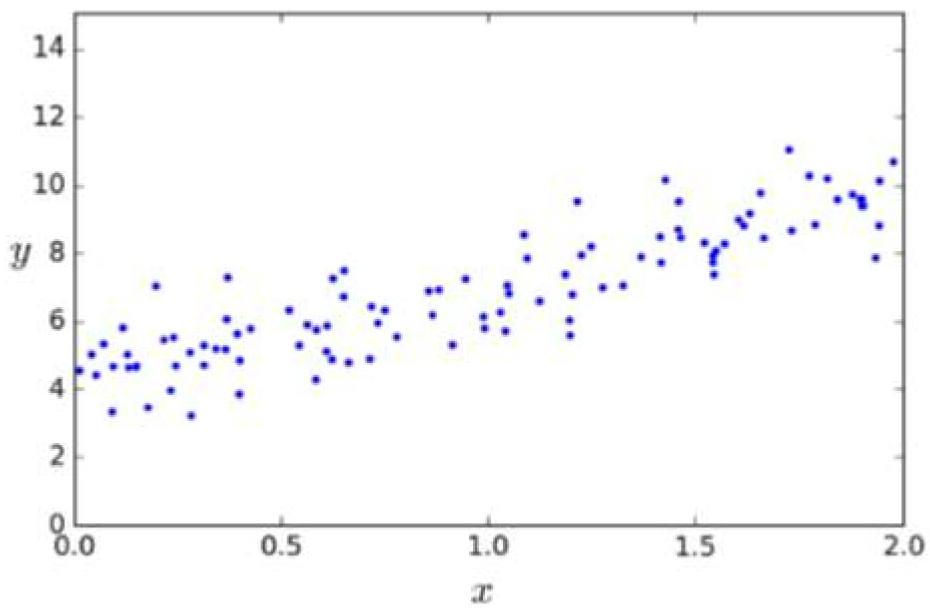
Regression

Relationships among several quantities

build a model that predicts the value of one variable as a function of other variables

simplest relation between two variables x and y is the linear equation

$$y = \beta_0 + \beta_1 x \qquad (x_1, y_1), \ldots, (x_n, y_n)$$



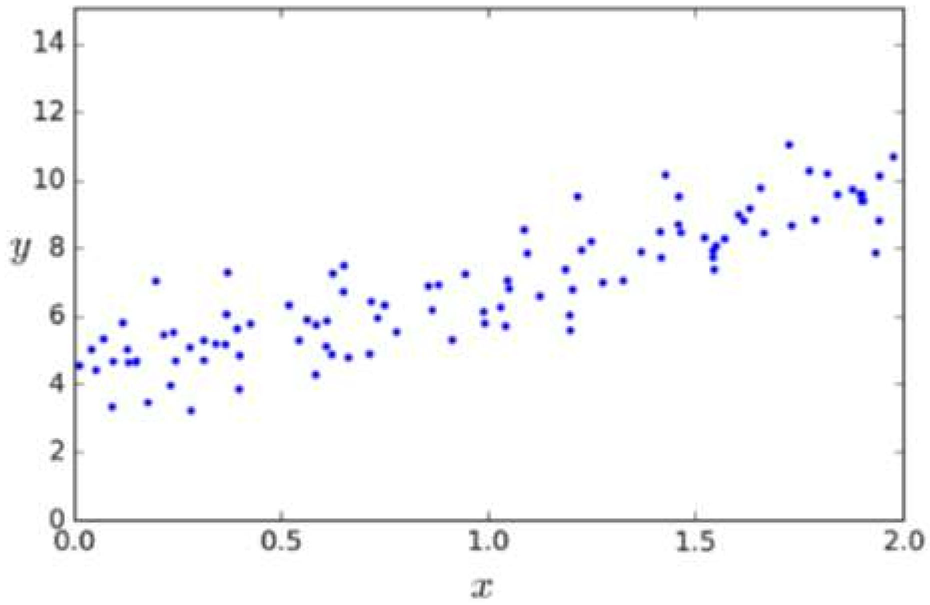
Regression of y on x

Regression

Relationships among several quantities
build a model that predicts the value of one variable as a function of other variables
simplest relation between two variables x and y is the linear equation

Regression coefficients

$$y = \beta_0 + \beta_1 x \qquad (x_1, y_1), \ldots, (x_n, y_n)$$



Regression of y on x

Regression

If the data points were on the line, the parameters would satisfy the equations

Predicted y-value		Observed y-value
$\beta_0 + \beta_1 x_1$	$=$	y_1
$\beta_0 + \beta_1 x_2$	$=$	y_2
$\vdots$		$\vdots$
$\beta_0 + \beta_1 x_n$	$=$	y_n

Regression

If the data points were on the line, the parameters would satisfy the equations

Predicted y-value		Observed y-value
$\beta_0 + \beta_1 x_1$	$=$	y_1
$\beta_0 + \beta_1 x_2$	$=$	y_2
$\vdots$		$\vdots$
$\beta_0 + \beta_1 x_n$	$=$	y_n

$$X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}, \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

$$\boxed{X\boldsymbol{\beta} = \mathbf{y}}$$

Regression

If the data points were on the line, the parameters would satisfy the equations

Predicted y-value		Observed y-value
$\beta_0 + \beta_1 x_1$	$=$	y_1
$\beta_0 + \beta_1 x_2$	$=$	y_2
$\vdots$		$\vdots$
$\beta_0 + \beta_1 x_n$	$=$	y_n

if the data points don't lie on a line

$$X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}, \quad \beta = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

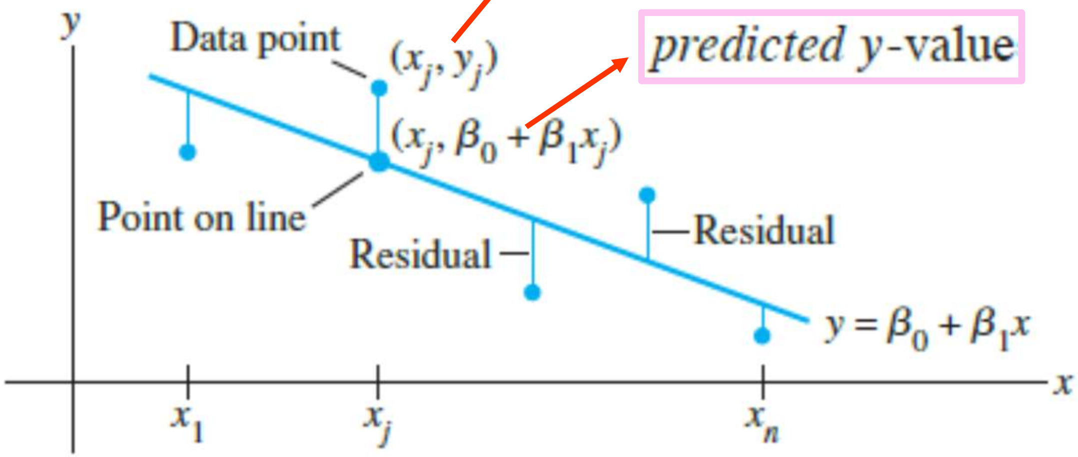
$$\boxed{X\beta = \mathbf{y}}$$

Regression

Suppose β_0 and β_1 are fixed.

y_j the *observed* value

predicted y-value



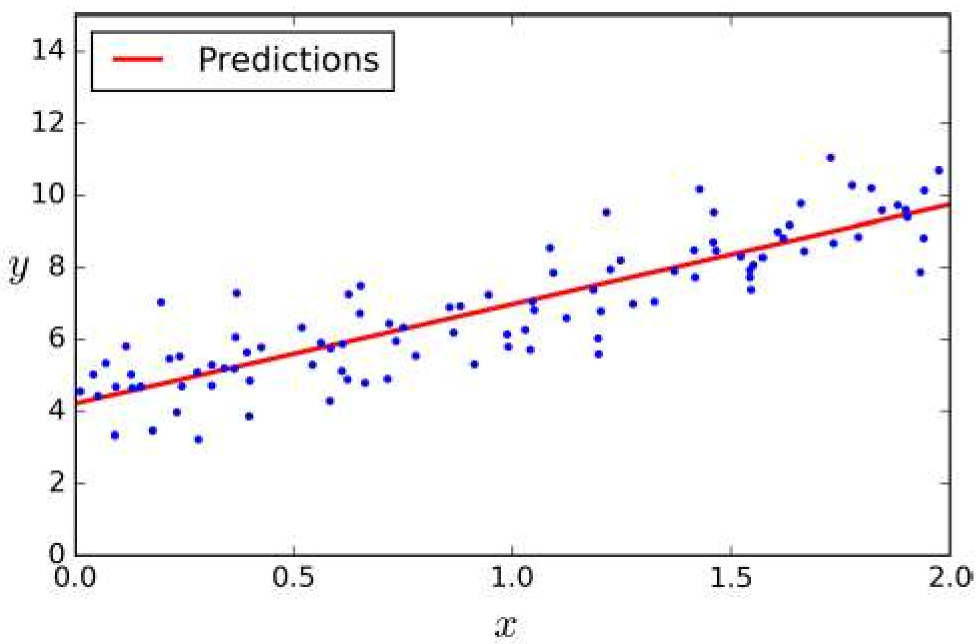
Regression

There are several ways to measure how “close” the line is to the data

The usual choice is to add the squares of the residuals

least-squares line is the that minimizes the sum of the squares of the residuals

$$residual = \epsilon = y - X\beta$$



Regression

There are several ways to measure how “close” the line is to the data

The usual choice is to add the squares of the residuals

least-squares line is the that minimizes the sum of the squares of the residuals

$$residual = \epsilon = y - X\beta$$

$$\min \|X\beta - y\|_2^2$$

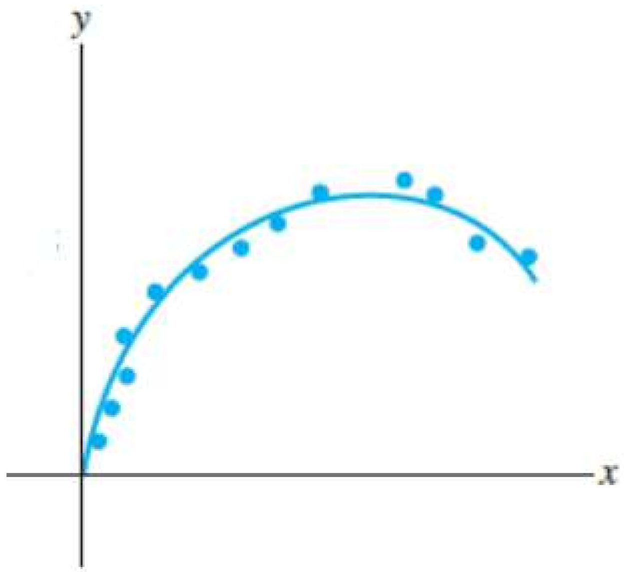
Cost function

Regression(Curve fitting)

data points $(x_1, y_1), \ldots, (x_n, y_n)$ on a scatter plot do not lie close to any line.

some other functional relationship between x and y

$$y = \beta_0 f_0(x) + \beta_1 f_1(x) + \cdots + \beta_k f_k(x)$$



Regression(Curve fitting)

Example

$$\boxed{y = \beta_0 + \beta_1 x + \beta_2 x^2}$$

$$(x_1, y_1), \ldots, (x_n, y_n)$$

Regression(Curve fitting)

Example

$$\boxed{y = \beta_0 + \beta_1 x + \beta_2 x^2}$$

$$(x_1, y_1), \ldots, (x_n, y_n)$$

$$y_1 = \beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \epsilon_1$$

$$\begin{aligned}
y_1 &= \beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \epsilon_1 \\
y_2 &= \beta_0 + \beta_1 x_2 + \beta_2 x_2^2 + \epsilon_2 \\
&\vdots \qquad\qquad\quad \vdots \\
y_n &= \beta_0 + \beta_1 x_n + \beta_2 x_n^2 + \epsilon_n
\end{aligned}$$

Regression(Curve fitting)

Example

$$\boxed{y = \beta_0 + \beta_1 x + \beta_2 x^2}$$

$$(x_1, y_1), \ldots, (x_n, y_n)$$

$$\begin{aligned}
y_1 &= \beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \epsilon_1 \\
y_1 &= \beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \epsilon_1 \\
y_2 &= \beta_0 + \beta_1 x_2 + \beta_2 x_2^2 + \epsilon_2 \\
\vdots & \qquad\qquad\quad \vdots \\
y_n &= \beta_0 + \beta_1 x_n + \beta_2 x_n^2 + \epsilon_n
\end{aligned}$$

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ \vdots & \vdots & \vdots \\ 1 & x_n & x_n^2 \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

Regression(Curve fitting)

Example

$$\boxed{y = \beta_0 + \beta_1 x + \beta_2 x^2}$$

$$(x_1, y_1), \ldots, (x_n, y_n)$$

$$
\begin{aligned}
y_1 &= \beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \epsilon_1 \\
y_1 &= \beta_0 + \beta_1 x_1 + \beta_2 x_1^2 + \epsilon_1 \\
y_2 &= \beta_0 + \beta_1 x_2 + \beta_2 x_2^2 + \epsilon_2 \\
\vdots & \qquad\qquad \vdots \\
y_n &= \beta_0 + \beta_1 x_n + \beta_2 x_n^2 + \epsilon_n
\end{aligned}
$$

$$
\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ \vdots & \vdots & \vdots \\ 1 & x_n & x_n^2 \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix}
$$

$$\mathbf{y} = X \boldsymbol{\beta} + \boldsymbol{\epsilon}$$

$$residual = \epsilon = y - X\beta$$

$$\min \|X\beta - y\|_2^2$$

Multiple Regression

Multiple Regression

We have n features and we want to predict y based on them

$$\boldsymbol{x_1, x_2, \dots, x_m} \Rightarrow \boldsymbol{y}$$

Multiple Regression

We have n features and we want to predict y based on them

$$\boldsymbol{x_1, x_2, \dots, x_m} \quad \Longrightarrow \quad \boldsymbol{y}$$

$$\boldsymbol{y = \beta_0 + \beta_1 x_1 + \beta_2\ x_2 + \cdots + \beta_m x_m}$$

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \qquad X = \begin{bmatrix} 1 & x_1^{(1)} & \dots & x_m^{(1)} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_1^{(n)} & \dots & x_m^{(n)} \end{bmatrix} \qquad \beta = \begin{bmatrix} \beta_0 \\ \vdots \\ \beta_m \end{bmatrix}$$

Multiple Regression

We have n features and we want to predict y based on them

$$\boldsymbol{x_1, x_2, \dots, x_m} \Rightarrow \boldsymbol{y}$$

$$\boldsymbol{y = \beta_0 + \beta_1 x_1 + \beta_2\, x_2 + \cdots + \beta_m x_m}$$

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \qquad X = \begin{bmatrix} 1 & x_1^{(1)} & \dots & x_m^{(1)} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_1^{(n)} & \dots & x_m^{(n)} \end{bmatrix} \qquad \beta = \begin{bmatrix} \beta_0 \\ \vdots \\ \beta_m \end{bmatrix}$$

$$\boldsymbol{residual = \epsilon = y - X\beta}$$

$$\mathbf{min}\ \|\boldsymbol{X\beta - y}\|_2^2$$

Multiple Regression

$$\boldsymbol{y} = \boldsymbol{\beta_0} + \boldsymbol{\beta_1 f_1(x_1)} + \boldsymbol{\beta_2\, f_2(x_2)} + \cdots + \boldsymbol{\beta_m f_m(x_m)}$$

$$X = \begin{bmatrix} 1 & f_1(x_1^{(1)}) & \dots & f_m(x_m^{(1)}) \\ \vdots & \vdots \quad \vdots & & \vdots \\ 1 & f_1(x_1^{(n)}) & \dots & f_m(x_m^{(n)}) \end{bmatrix} \quad y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \quad \beta = \begin{bmatrix} \beta_0 \\ \vdots \\ \beta_m \end{bmatrix}$$

Multiple Regression

$$\boldsymbol{y} = \boldsymbol{\beta}_0 + \boldsymbol{\beta}_1 \boldsymbol{f}_1(\boldsymbol{x}_1) + \boldsymbol{\beta}_2\, \boldsymbol{f}_2(\boldsymbol{x}_2) + \cdots + \boldsymbol{\beta}_m \boldsymbol{f}_m(\boldsymbol{x}_m)$$

$$X = \begin{bmatrix} 1 & f_1(x_1^{(1)}) & \dots & f_m(x_m^{(1)}) \\ \vdots & \vdots \quad \vdots & & \vdots \\ 1 & f_1(x_1^{(n)}) & \dots & f_m(x_m^{(n)}) \end{bmatrix} \qquad y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \qquad \beta = \begin{bmatrix} \beta_0 \\ \vdots \\ \beta_m \end{bmatrix}$$

Least Square Problem

$$\boldsymbol{residual} = \boldsymbol{\epsilon} = \boldsymbol{y} - \boldsymbol{X\beta}$$

$$\mathbf{min}\ \|\boldsymbol{X\beta} - \boldsymbol{y}\|_2^2$$

Solving Least Square Problem

least-squares solution is a solution of the normal equations

$$\boxed{\min \|X\beta - y\|_2^2}$$

$$X^T X \beta = X^T \mathbf{y}$$

High computational complexity

Solving Least Square Problem

least-squares solution is a solution of the normal equations

$$\boxed{\min \|X\beta - y\|_2^2}$$

$$X^T X \beta = X^T \mathbf{y}$$

High computational complexity

Optimization Methods: Gradient Descent

Optimization algorithms

$$x_0$$

⇓

generate a sequence of iterates $\{x_k\}_{k=0}^{\infty}$

⇓

terminate : no more progress or a solution point with sufficient accuracy

Optimization algorithms

x_0

generate a sequence of iterates $\{x_k\}_{k=0}^{\infty}$

terminate : no more progress or a solution point with sufficient accuracy

$$x_{k+1} = x_k + \alpha p_k$$

direction

Step length or Learning Rate

α?

p_k?

Optimization algorithms

direction

Descent methods

✓any descent direction is guaranteed to produce a decrease in f , provided that the step length is sufficiently small

Need definition of Gradient

Gradient

$$f : \mathbf{R}^n \to \mathbf{R}$$

f is real-valued

$$\nabla f(x) \quad \Rightarrow \quad \nabla f(x)_i = \frac{\partial f(x)}{\partial x_i}, \quad i = 1, \ldots, n.$$

Gradient: example

Example:

quadratic function $$f : \mathbf{R}^n \to \mathbf{R}$$

$$f(x) = (1/2)x^T P x + q^T x + r$$

$P \in \mathbf{S}^n$, $q \in \mathbf{R}^n$, and $r \in \mathbf{R}$

Gradient: example

Example:

quadratic function $f : \mathbf{R}^n \to \mathbf{R}$

$$f(x) = (1/2)x^T P x + q^T x + r$$

$P \in \mathbf{S}^n$, $q \in \mathbf{R}^n$, and $r \in \mathbf{R}$

$$\nabla f(x) = Px + q$$

Directional Derivative

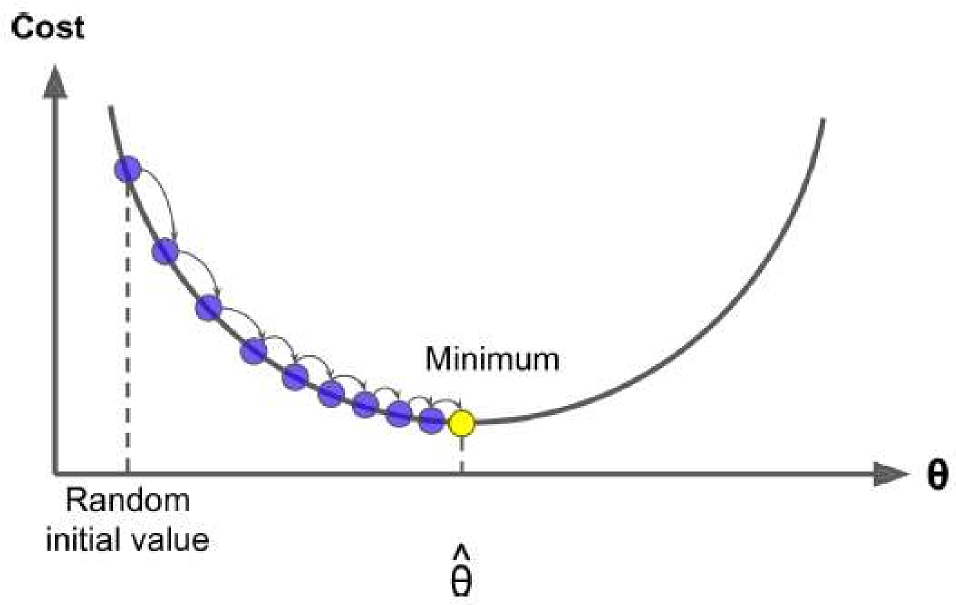
$$\nabla_p f(x) = < \nabla f(x), p >$$

$\nabla_p f(x) < 0$ ⇒ P is a descent direction

Gradient Descent

steepest descent direction

✓steepest descent direction $-\nabla f_k$ is the most obvious choice for search direction for a line search method.

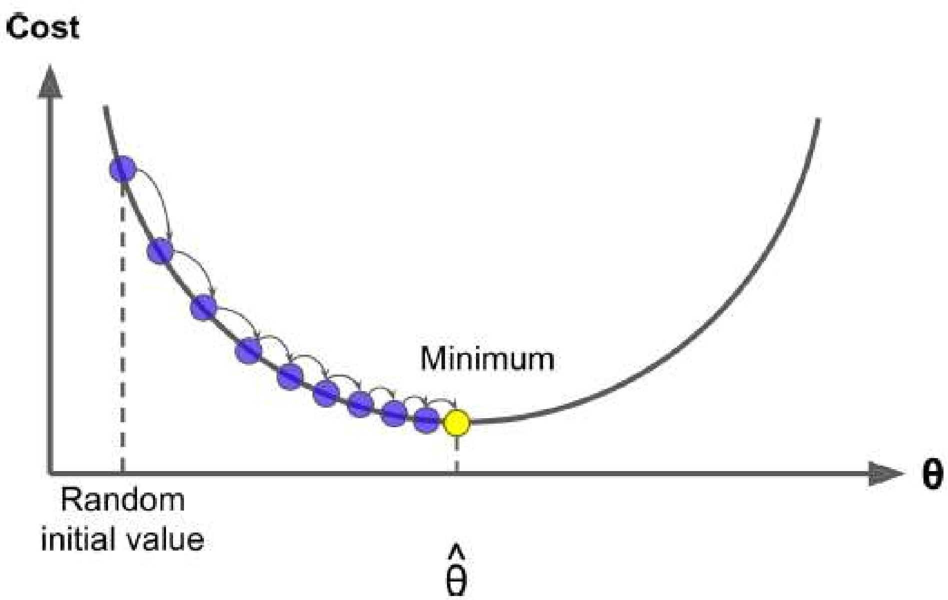


Gradient Descent

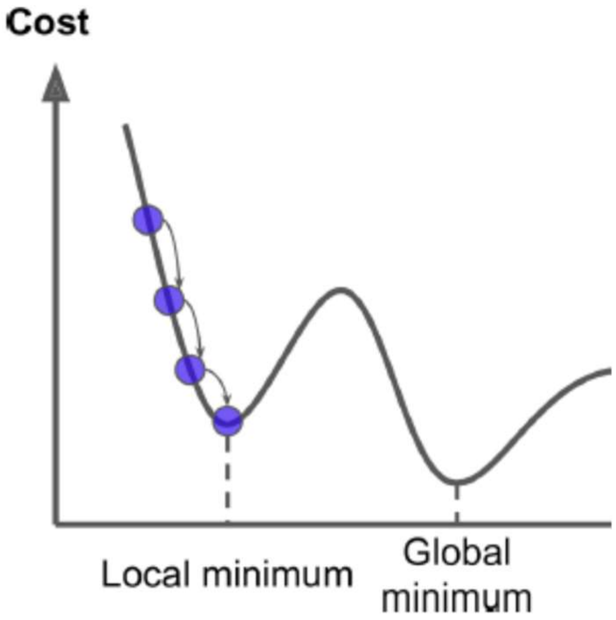
steepest descent direction

✓steepest descent direction $-\nabla f_k$ is the most obvious choice for search direction for a line search method.

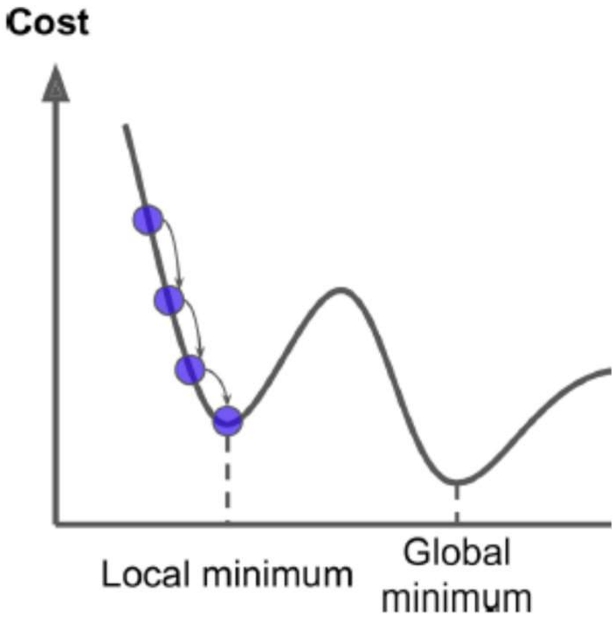
✓choose the step length α in a variety of ways $x_{k+1} = x_k + \alpha_k(-\nabla f(x_k))$



Gradient Descent

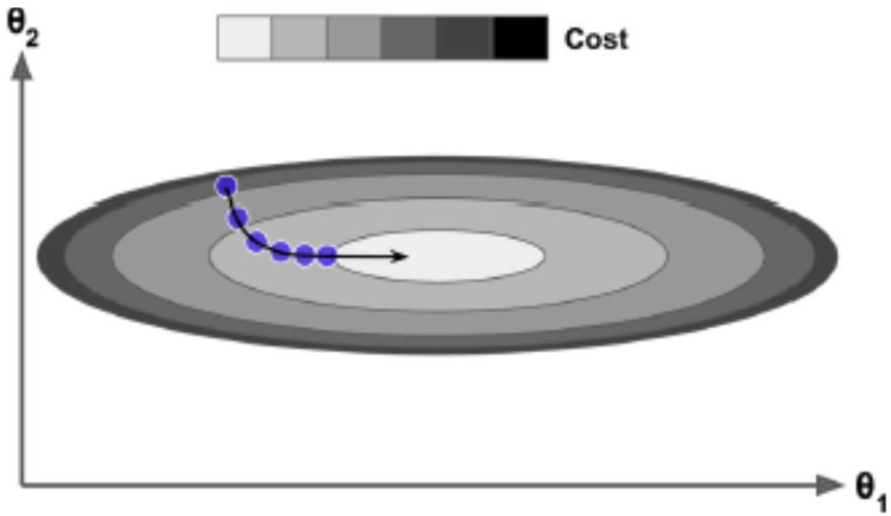


Gradient Descent

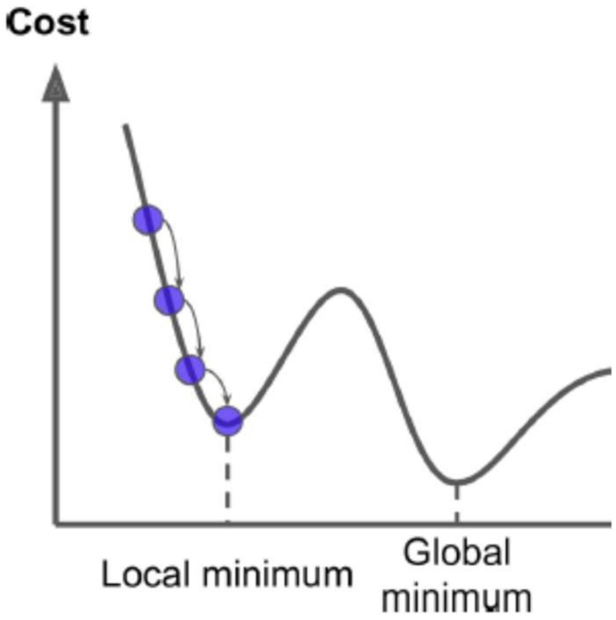


$$f(\boldsymbol{\beta}) = \|\boldsymbol{X\beta} - \boldsymbol{y}\|_2^2$$

Convex function



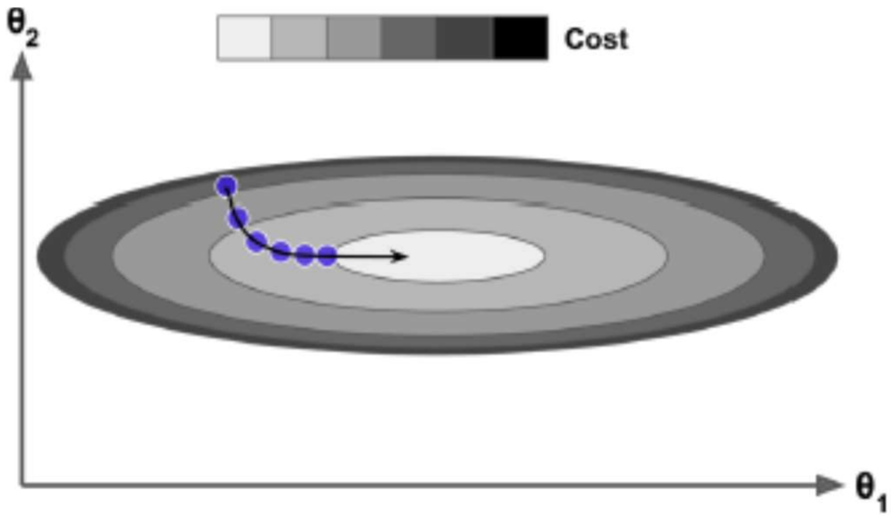
Gradient Descent



$$f(\boldsymbol{\beta}) = \|\boldsymbol{X\beta} - \boldsymbol{y}\|_2^2$$

Convex function

$$\nabla f(\boldsymbol{\beta}) = \boldsymbol{X}^T\boldsymbol{X\beta} - \boldsymbol{X}^T\boldsymbol{y}$$



Gradient Descent

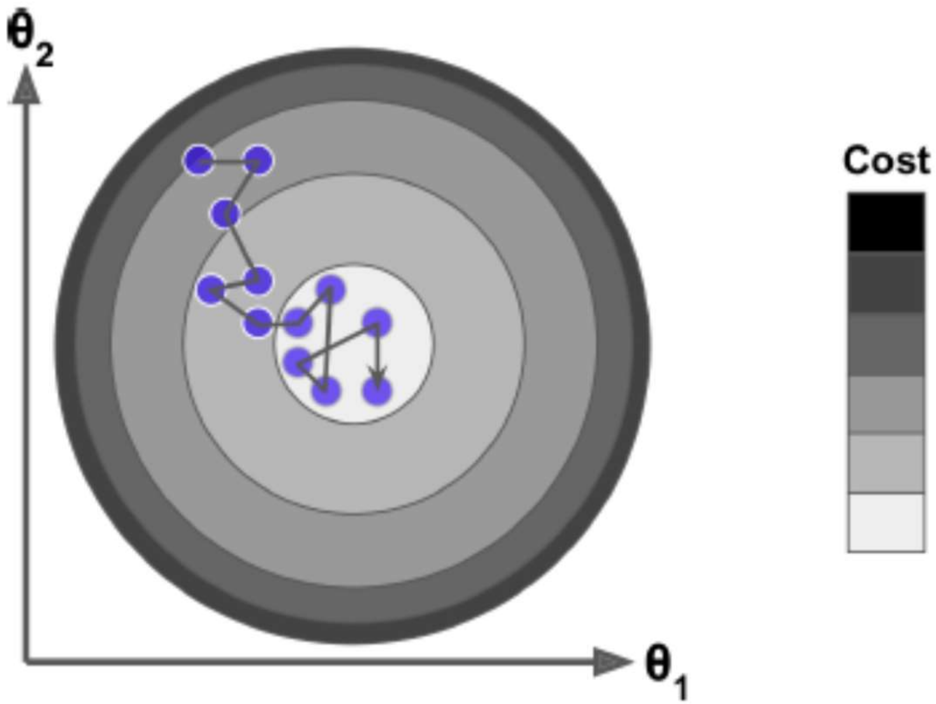
- ✓ training a model means searching for a combination of model parameters that minimizes a cost function
- ✓ search in the model's parameter space
- ✓ more parameters a model has, more dimensions this space has, and the harder search
- ✓ Batch Gradient Descent uses the whole training set to compute the gradients at every step, which makes it very slow when the training set is large.

Gradient Descent

- ✓ training a model means searching for a combination of model parameters that minimizes a cost function
- ✓ search in the model's parameter space
- ✓ more parameters a model has, more dimensions this space has, and the harder search
- ✓ Batch Gradient Descent uses the whole training set to compute the gradients at every step, which makes it very slow when the training set is large.

- ✓ Stochastic Gradient Descent : picks a random instance in the training set at every step and computes the gradients based only on that single instance.
- ✓ is much less regular than Batch Gradient Descent
- ✓ instead of gently decreasing until it reaches the minimum, the cost function will bounce up and down, decreasing only on average.

Stochastic Gradient Descent



final parameter values are good, but not optimal

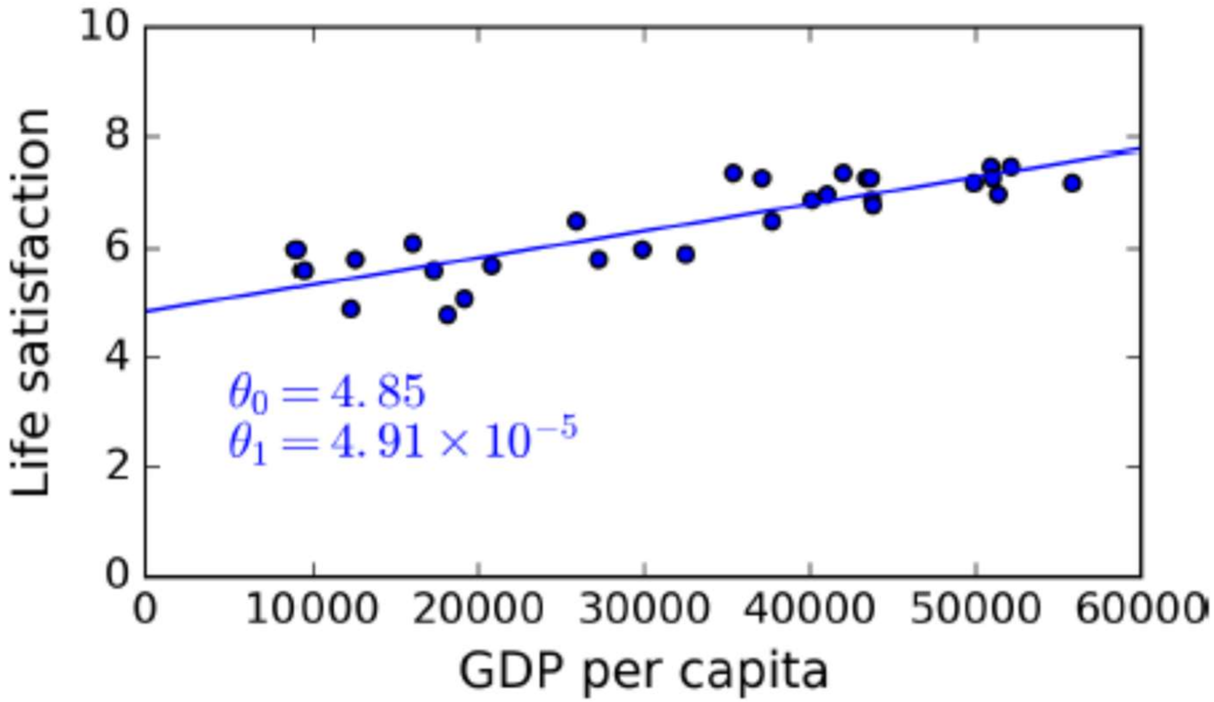
Overfitting and Underfitting

Overfitting

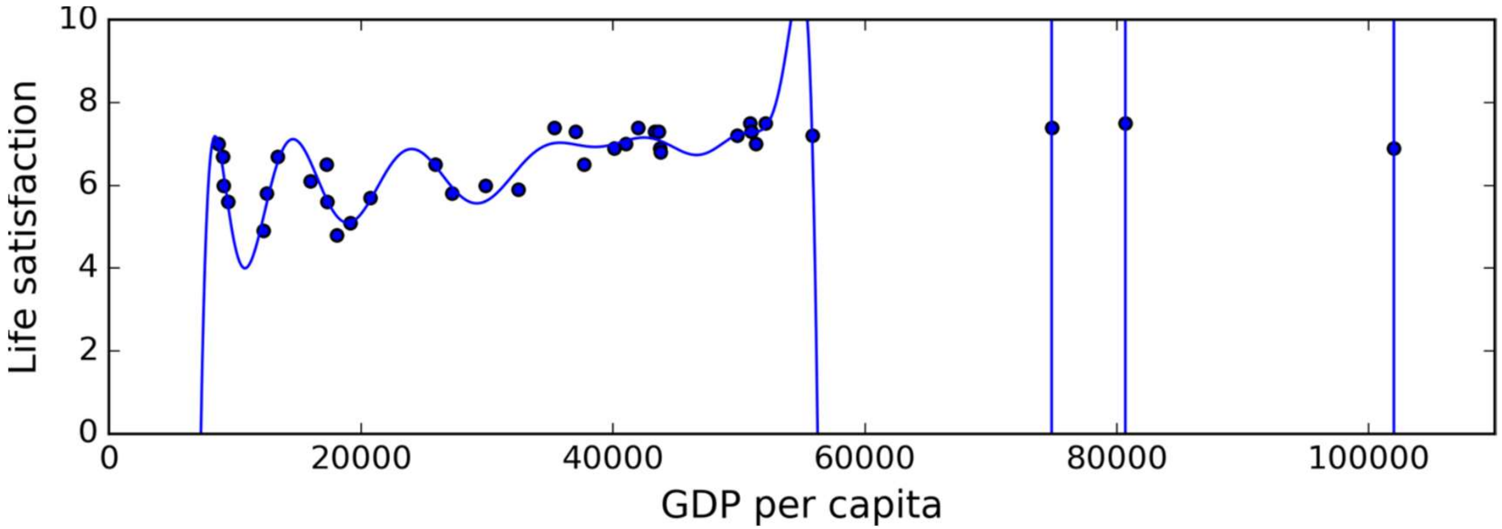
- ❖ Overgeneralizing is something that we humans do all too often
- ❖ machines can fall into the same trap
- ❖ In Machine Learning this is called overfitting
- ❖ model performs well on the training data, but it does not generalize well

Overfitting

- ❖ Overgeneralizing is something that we humans do all too often
- ❖ machines can fall into the same trap
- ❖ In Machine Learning this is called overfitting
- ❖ model performs well on the training data, but it does not generalize well



Overfitting



Even though it performs much better on the training data than the simple linear model, would you really trust its predictions?

Overfitting

Overfitting happens when the model is too complex relative to the amount and noisiness of the training data

- simplify the model
- gather more training data
- reduce the noise in the training data

Overfitting

Overfitting happens when the model is too complex relative to the amount and noisiness of the training data

- ❖ simplify the model
- ❖ gather more training data
- ❖ reduce the noise in the training data

Underfitting

- ❖ underfitting is the opposite of overfitting
- ❖ it occurs when your model is too simple to learn the underlying structure of the data
- ❖ more powerful model

Bias/Variance Tradeoff

Bias

- due to wrong assumptions
- Assuming that the data is linear when it is actually quadratic.
- A high-bias model is most likely to underfit the training data.

Variance

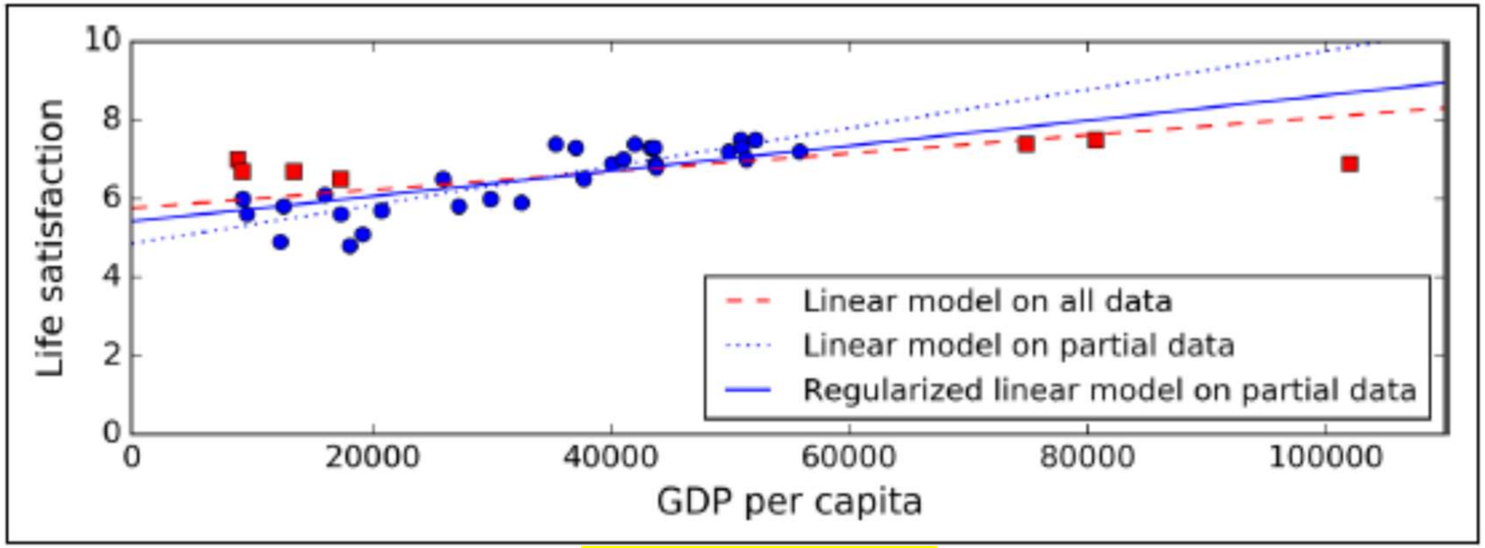
- model's excessive sensitivity to small variations in the training data.
- A model with many degrees of freedom is likely to have high variance
- overfit the training data.

Regularization

- Constraining a model to make it simpler and reduce the risk of overfitting is called regularization
- degrees of freedom
- force it to keep it small
- find the right balance between fitting the data perfectly and keeping the model simple enough to ensure that it will generalize well

Regularization

- ❖ Constraining a model to make it simpler and reduce the risk of overfitting is called regularization
- ❖ degrees of freedom
- ❖ force it to keep it small
- ❖ find the right balance between fitting the data perfectly and keeping the model simple enough to ensure that it will generalize well



hyperparameter

Regularized Regression

- ✓ the fewer degrees of freedom model has, the harder it will be for it to overfit the data
- ✓ For a linear model, regularization is typically achieved by constraining the weights of the model.

Ridge Regression : *Tikhonov regularization*

Regularized Regression

- ✓ the fewer degrees of freedom model has, the harder it will be for it to overfit the data
- ✓ For a linear model, regularization is typically achieved by constraining the weights of the model.

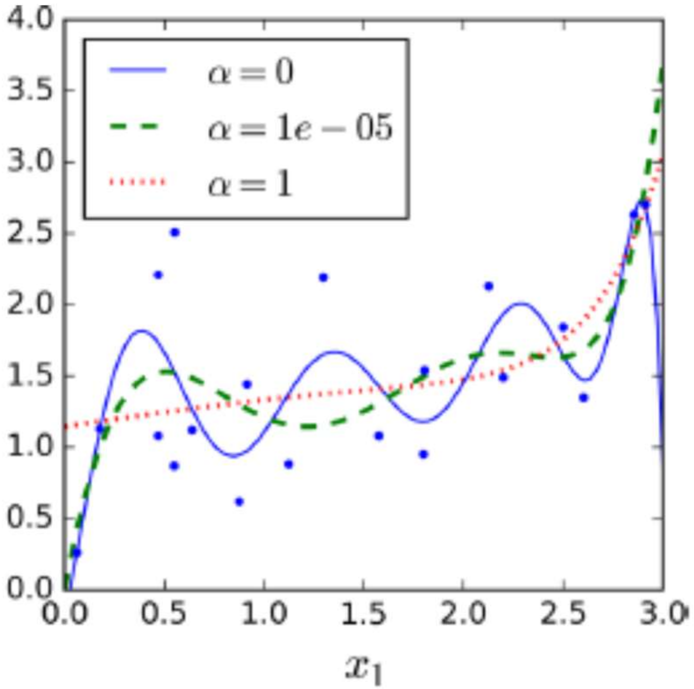
Ridge Regression : *Tikhonov regularization*

Hyperparameter

regularization term $\|\beta\|^2$

Cost Function $\boldsymbol{f}(\boldsymbol{\beta}) = \|\boldsymbol{X\beta} - \boldsymbol{y}\|_2^2 + \boldsymbol{\alpha}\|\boldsymbol{\beta}\|_2^2$

Tikhonov regularization



Lasso Regression

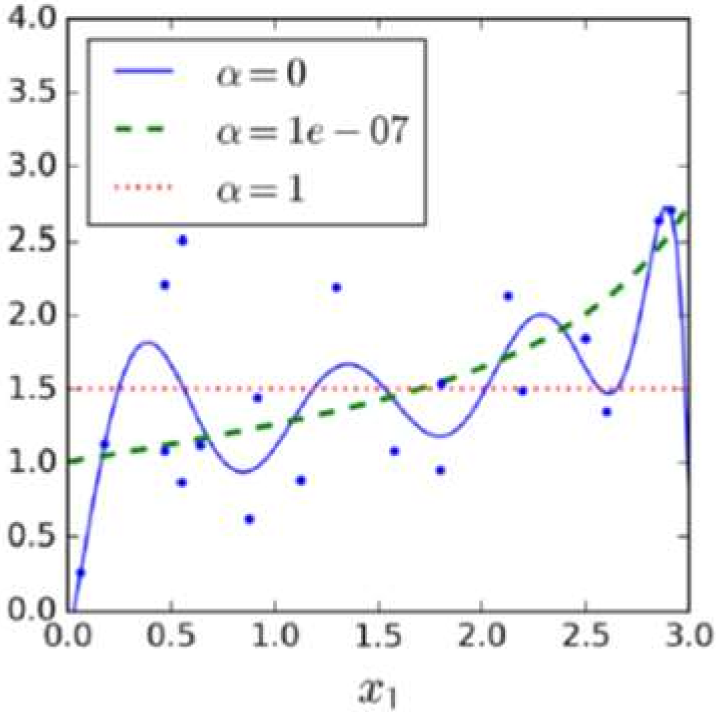
Least Absolute Shrinkage
Selection Operator Regression

$$\boldsymbol{f}(\boldsymbol{\beta}) = \|\boldsymbol{X\beta} - \boldsymbol{y}\|_2^2 + \boldsymbol{\alpha}\|\boldsymbol{\beta}\|_1$$

Lasso Regression

Least Absolute Shrinkage
Selection Operator Regression

$$f(\boldsymbol{\beta}) = \|\boldsymbol{X\beta} - \boldsymbol{y}\|_2^2 + \boldsymbol{\alpha}\|\boldsymbol{\beta}\|_1$$



Lasso Regression

Lasso Regression automatically performs feature selection and outputs a *sparse model*

Lasso cost function is not differentiable

Gradient Descent

Subgradient

Lasso Regression

Lasso Regression automatically performs feature selection and outputs a *sparse model*

Lasso cost function is not differentiable

Gradient Descent

Subgradient

$$\boldsymbol{\alpha}\|\boldsymbol{\beta}\|_1 \Rightarrow \alpha \begin{bmatrix} sign(\beta_1) \\ sign(\beta_2) \\ \vdots \\ sign(\beta_m) \end{bmatrix}$$