第 6 章　线性回归

根据已知训练数据学习/训练，选择合适的模型h(X)，代价函数越小，h函数越逼近实际目标函数。

$$\min_{\beta} \quad \mathrm{J}(\beta) = \frac{1}{2\mathrm{m}} \sum_{i=1}^{m} \left(h_{\beta}(X_i) - Y_i\right)^2$$

找全局最优解

回归

- 指研究一组随机变量(Y1 ，Y2 ，…，Yi)和另一组(X1，X2，…，Xk)变量之间关系的统计分析方法。回归分析是一种数学模型。
- 是监督学习的一个重要问题，用于预测输入变量和输出变量之间的关系。
- 回归模型是表示输入变量到输出变量之间映射的函数。
- 回归问题等价于函数拟合：使用一条函数曲线使其很好地拟合已知函数，且很好地预测未知数据。

回归应用

- 预测空气指数PM2.5

  f（过去的PM2.5信息）＝明天的PM2.5值

- 股票预测

  f（  ）＝股票走势等

- 自动驾驶

  f（  ）＝方向盘角度、油门刹车幅度

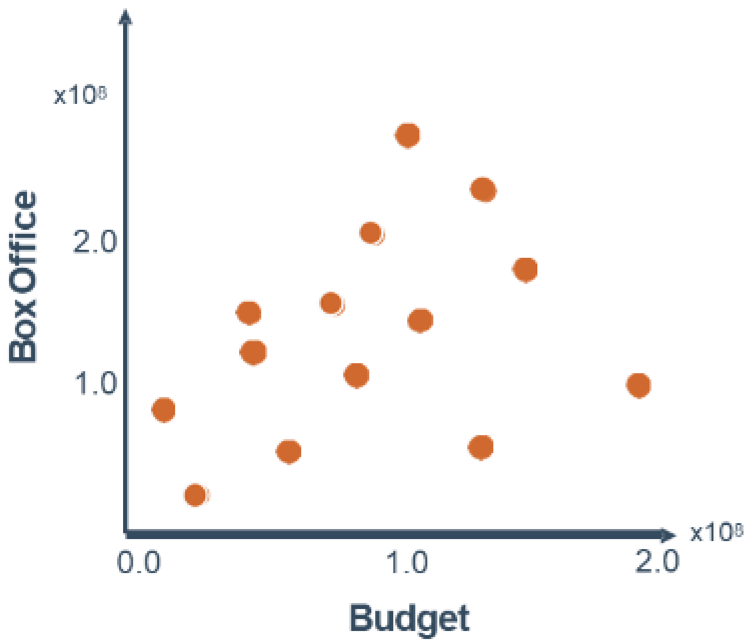
- 推荐系统

  f（用户A　　商品B）＝购买可能性值

线性回归

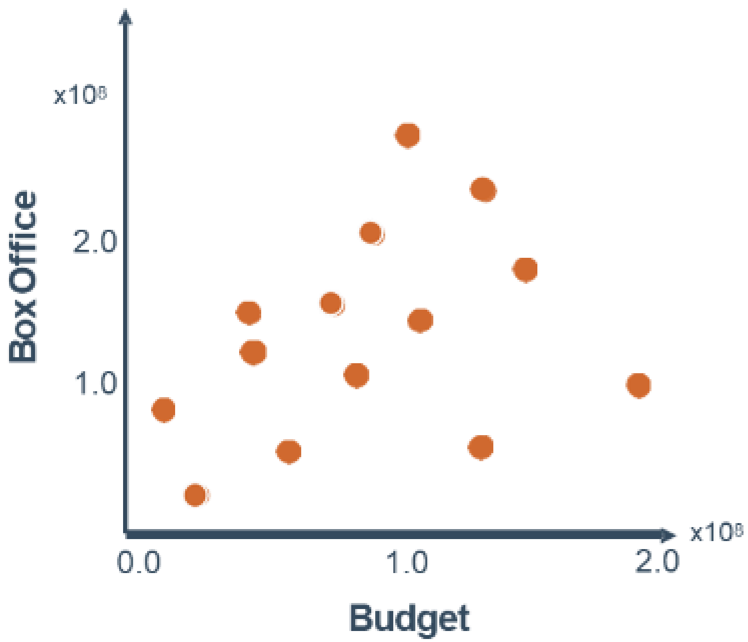
- 线性回归（linear regression）是利用数理统计中的回归分析，来确定两个或两个以上变量间相互依赖的定量关系的一种统计分析方法，运用十分广泛。
- 机器学习中，线性回归假设特征和目标变量之间满足线性关系，根据给定的训练数据建立一个线性模型，并用此模型进行预测。

线性回归



$$y_\beta(x) = \beta_0 + \beta_1 x$$

线性回归



$$y_\beta(x) = \beta_0 + \beta_1 x$$

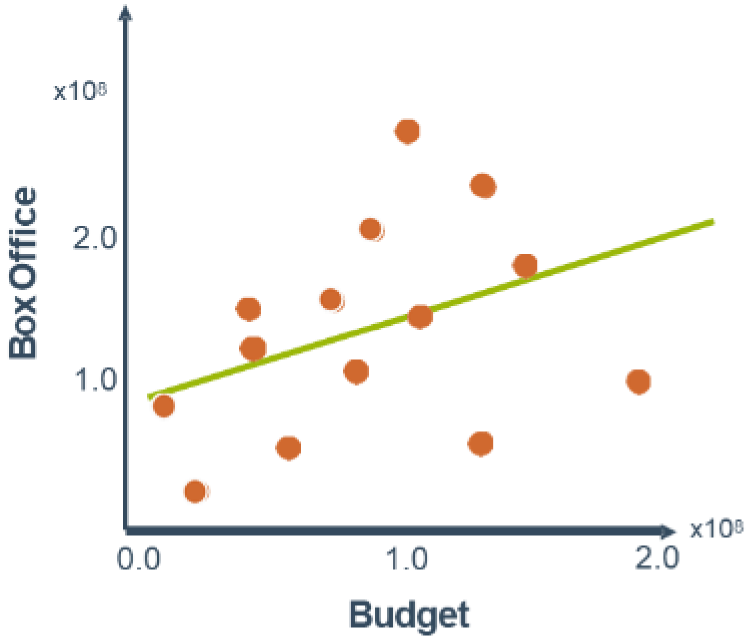
box office revenue

coefficient 0

coefficient 1

movie budget

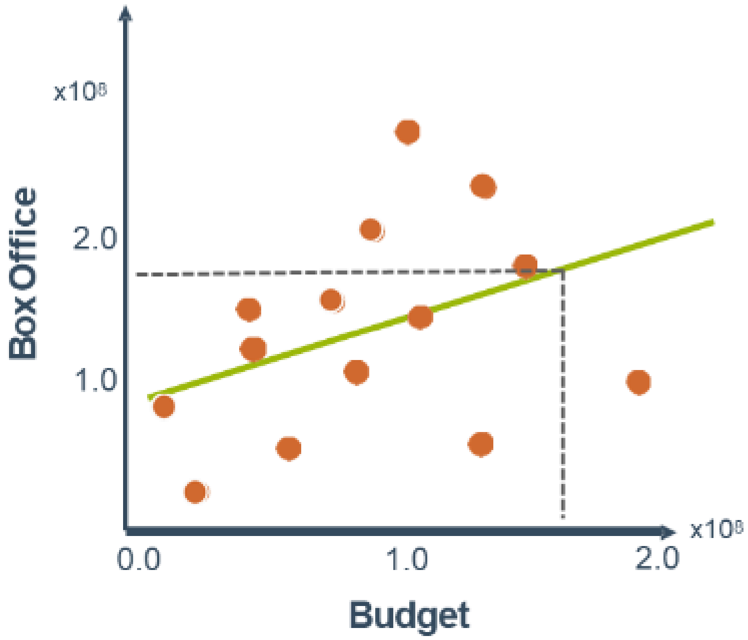
线性回归



$$y_\beta(x) = \beta_0 + \beta_1 x$$

β_0= 80 million, β_1= 0.6

使用线性回归预测

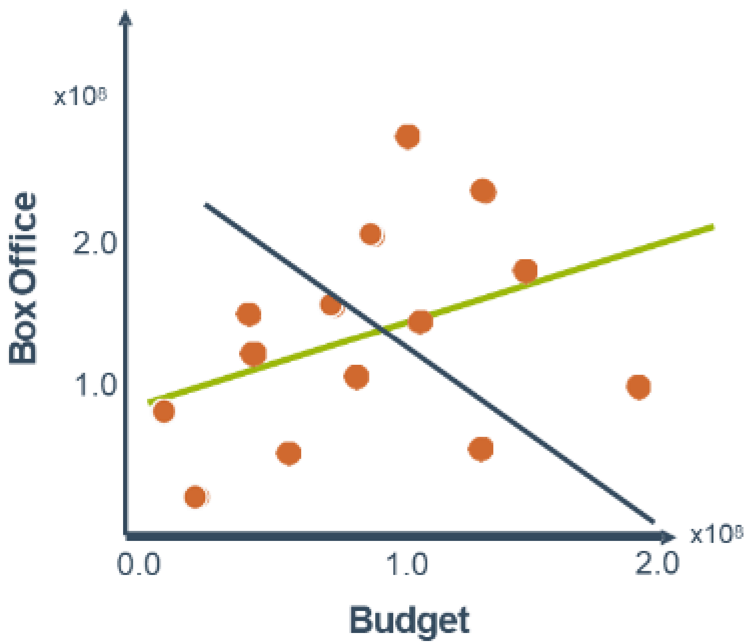


$$y_\beta(x) = \beta_0 + \beta_1 x$$

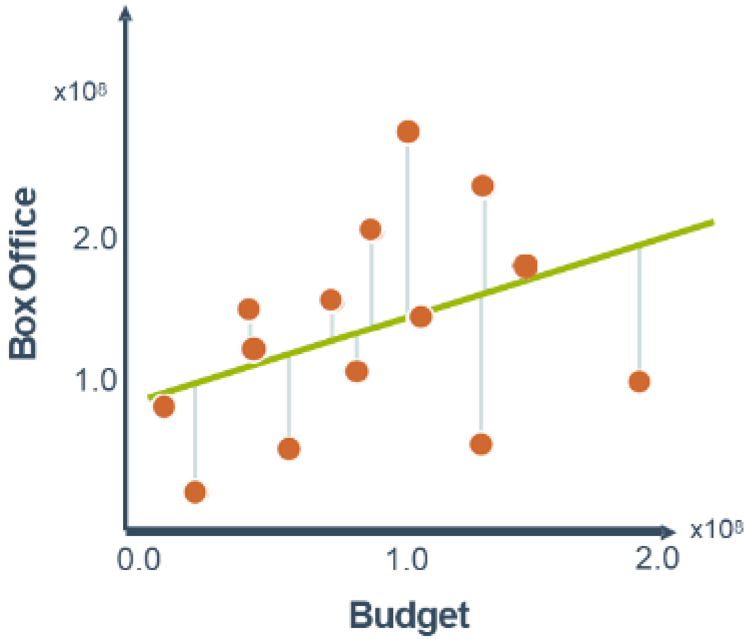
β_0= 80 million, β_1= 0.6

给定1.6亿的预算，预测票房收益为1.75亿

哪个模型拟合得更好？



计算残差

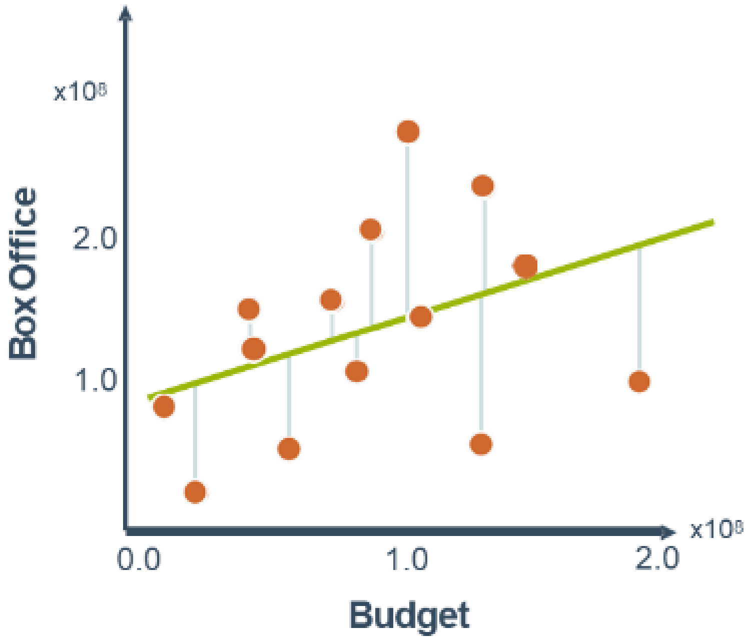


$$y_\beta\left(x_{obs}^{(i)}\right) - y_{obs}^{(i)}$$

↑ Predicted value

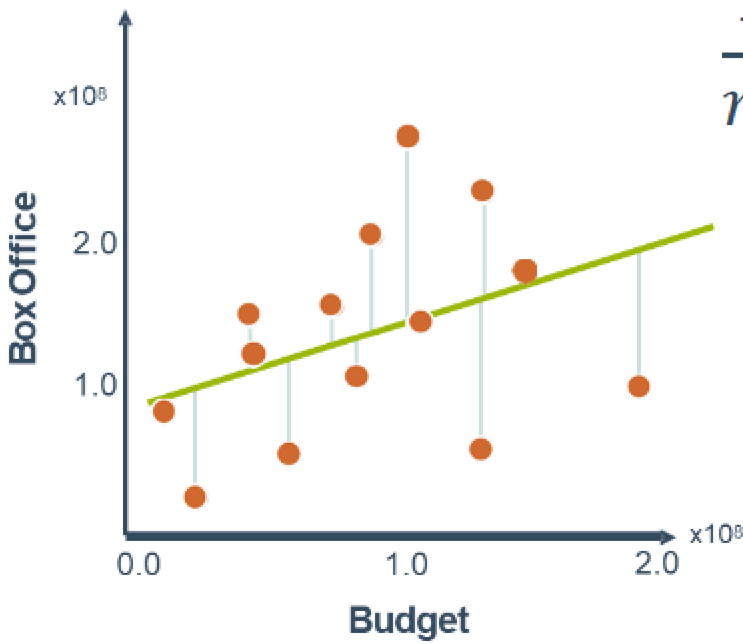
↑ Observed value

计算残差



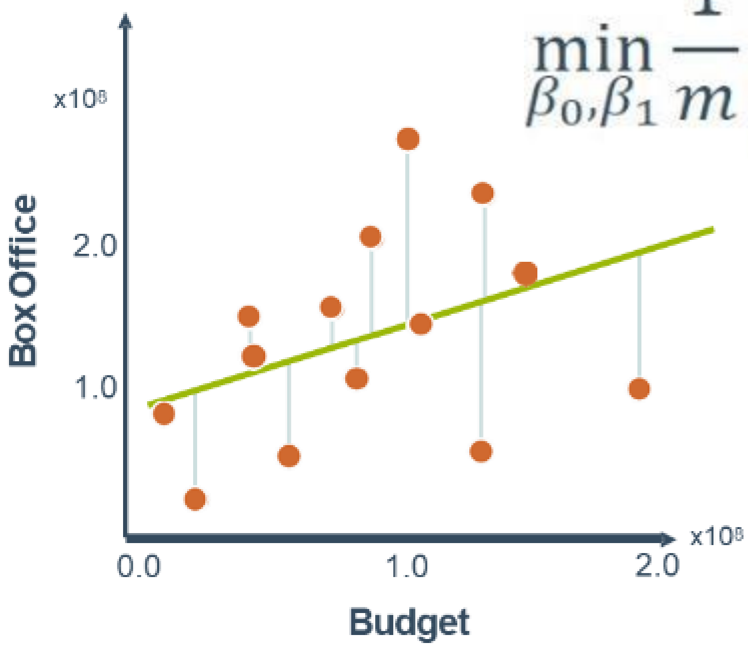
$$\left(\beta_0 + \beta_1 x_{obs}^{(i)}\right) - y_{obs}^{(i)}$$

均方误差（Mean Squared Error, MSE）



$$\frac{1}{m}\sum_{i=1}^{m}\left(\left(\beta_0+\beta_1 x_{obs}^{(i)}\right)-y_{obs}^{(i)}\right)^2$$

最小均方误差



$$\min_{\beta_0,\beta_1} \frac{1}{m} \sum_{i=1}^{m} \left(\left(\beta_0 + \beta_1 x_{obs}^{(i)} \right) - y_{obs}^{(i)} \right)^2$$

其他评价指标

平均绝对误差（Mean Absolute Error, MAE）：

$$\frac{1}{m}\sum_{i=1}^{m}\left|y_\beta\left(x^{(i)}\right)-y_{obs}^{(i)}\right|$$

均方根误差（Root Mean Squared Error, RMSE）：

$$\sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(y_\beta\left(x^{(i)}\right)-y_{obs}^{(i)}\right)^2}$$

MAE is the easiest to understand, because it's the average error.
MSE is more popular than MAE, because MSE "punishes" larger errors.
RMSE is even more popular than MSE, because RMSE is interpretable in the "y" units.

其他评价指标

残差平方和（SSE）：

$$\sum_{i=1}^{m}\left(y_\beta(x^{(i)}) - y_{obs}^{(i)}\right)^2$$

总离差平方和（TSS）：

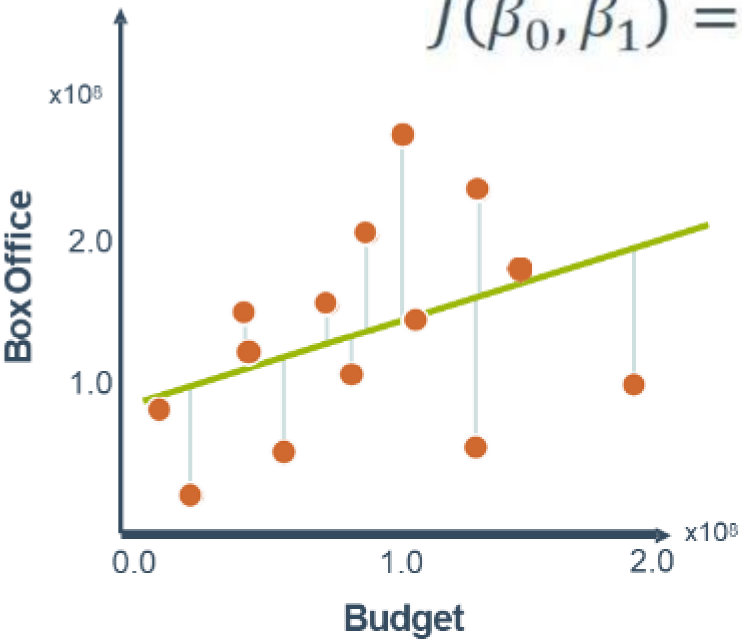
$$\sum_{i=1}^{m}\left(\overline{y_{obs}} - y_{obs}^{(i)}\right)^2$$

决定系数（R2）：

$$1 - \frac{SSE}{TSS}$$

代价函数

$$J(\beta_0, \beta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(\left(\beta_0 + \beta_1 x_{obs}^{(i)}\right) - y_{obs}^{(i)}\right)^2$$



最小二乘法

$$y_\beta(x) = \beta_0 + \beta_1 x$$

$$\widehat{\beta_1} = \frac{\sum_{\mathrm{i}=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sum_{\mathrm{i}=1}^{n}(x_i - \overline{x})^2}$$

$$\widehat{\beta_0} = \overline{y} - \widehat{\beta_1}\overline{x}$$

最小二乘法

	工资	额度
1	4000	20000
2	5000	30000
3	8000	50000
4	10000	70000
5	12000	60000
6	15000	?

（1）求贷款额度 y 关于月收入 x 的线性回归方程；

（2）利用（1）中的回归方程，预测张三（月工资15000）的贷款额度。

最小二乘法

某地区2007年至2013年农村居民家庭人均纯收入 y （单位：千元）的数据如下表：

年份	2007	2008	2009	2010	2011	2012	2013
年份代号 x	1	2	3	4	5	6	7
人均收入 y	2.9	3.3	3.6	4.4	4.8	5.2	5.9

（ 1 ）求 y 关于 x 的线性回归方程；

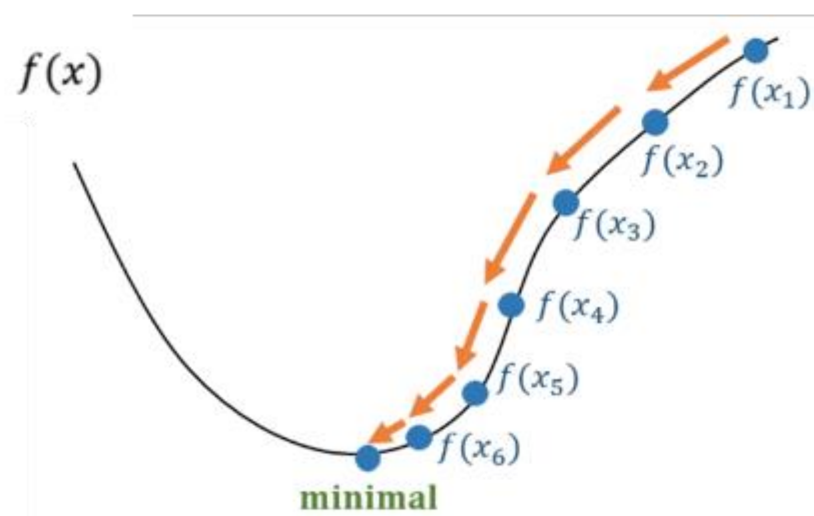
（ 2 ）利用（ 1 ）中的回归方程，预测该地区2015年农村居民家庭人均纯收入。

梯度下降法

$$y_\beta(x) = \beta_0 + \beta_1 x$$

$$J(\beta_0, \beta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(\left(\beta_0 + \beta_1 x_{obs}^{(i)}\right) - y_{obs}^{(i)}\right)^2$$

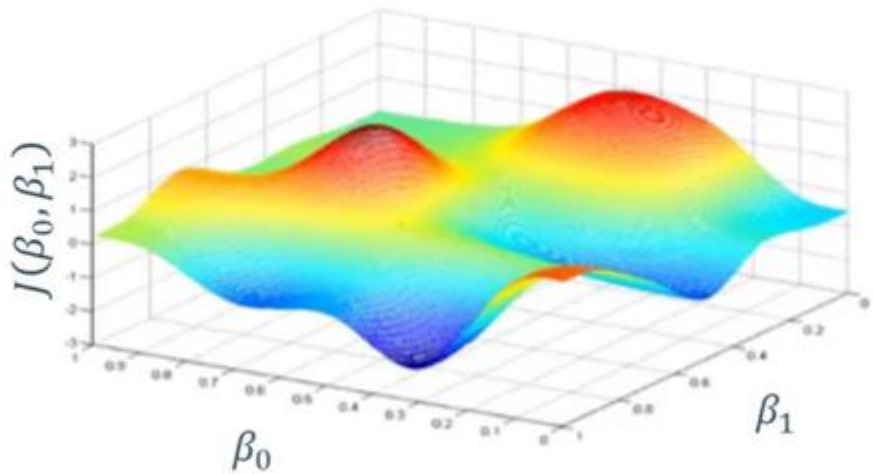
一种常用的一阶优化方法。该算法从任一点开始，沿该点**梯度**的反方向运动一段距离，再沿新位置的梯度反方向运行一段距离，如此迭代，解一直朝下坡最陡的方向运动，希望能运动到函数的全局最小点。

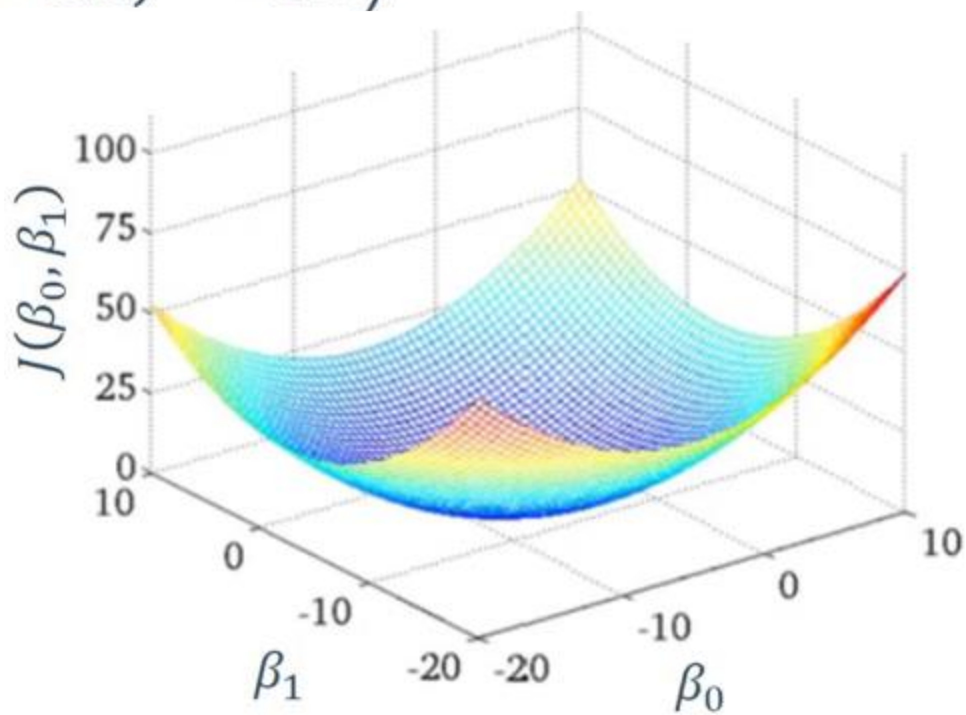


梯度下降法

$$y_\beta(x) = \beta_0 + \beta_1 x$$

$$J(\beta_0, \beta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(\left(\beta_0 + \beta_1 x_{obs}^{(i)}\right) - y_{obs}^{(i)}\right)^2$$



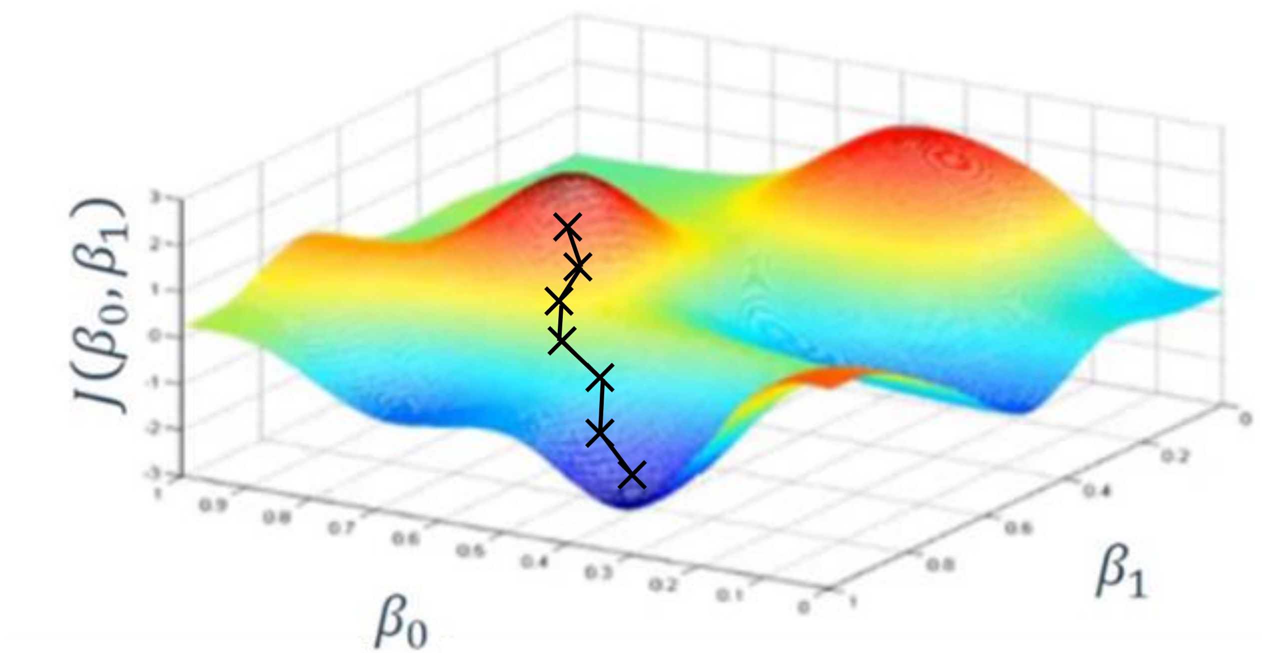


梯度下降法

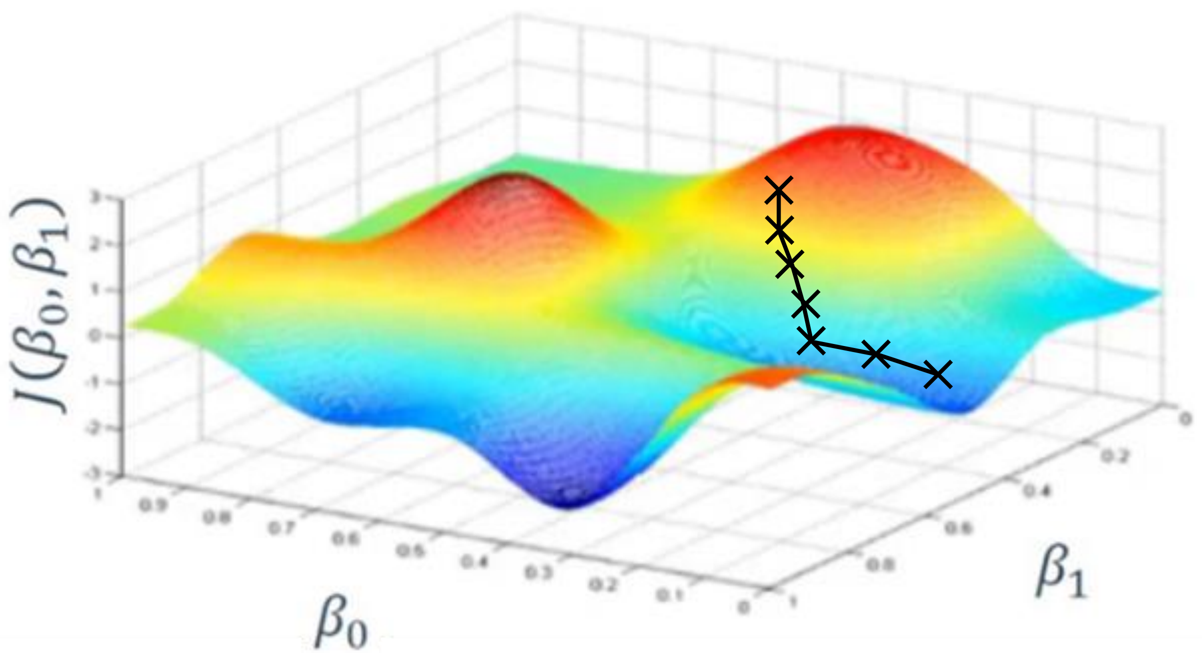
$J(\beta_0,\beta_1) \Rightarrow \min\limits_{\beta_0,\beta_1} J(\beta_0,\beta_1)$

初始：（β_0,β_1）$\xrightarrow{\text{不断改变 } \beta_0,\beta_1}$ $J(\beta_0,\beta_1)\downarrow$

梯度下降法



梯度下降法



梯度下降法

数学定义：

$$\beta_i := \beta_i - \alpha \frac{\partial}{\partial \beta_i} \mathrm{J}(\beta_0, \beta_1), \quad i=0,1 \Rightarrow$$ 不断迭代直到收敛

正确过程：同时更新

$$\text{temp0} := \beta_0 - \alpha \frac{\partial}{\partial \beta_0} \mathrm{J}(\beta_0, \beta_1)$$

$$\text{temp1} := \beta_1 - \alpha \frac{\partial}{\partial \beta_1} \mathrm{J}(\beta_0, \beta_1)$$

$$\beta_0 := \text{temp0}$$

$$\beta_1 := \text{temp1}$$

梯度下降法

偏导数项意义：

$$\beta_i := \beta_i - \alpha \frac{\partial}{\partial \beta_i} \mathrm{J}(\beta_0, \beta_1), \quad i=0,1$$

$$\Downarrow$$

$$\min_{\beta_1} \mathrm{J}(\beta_1)$$ 为例

梯度下降法

学习速率α：

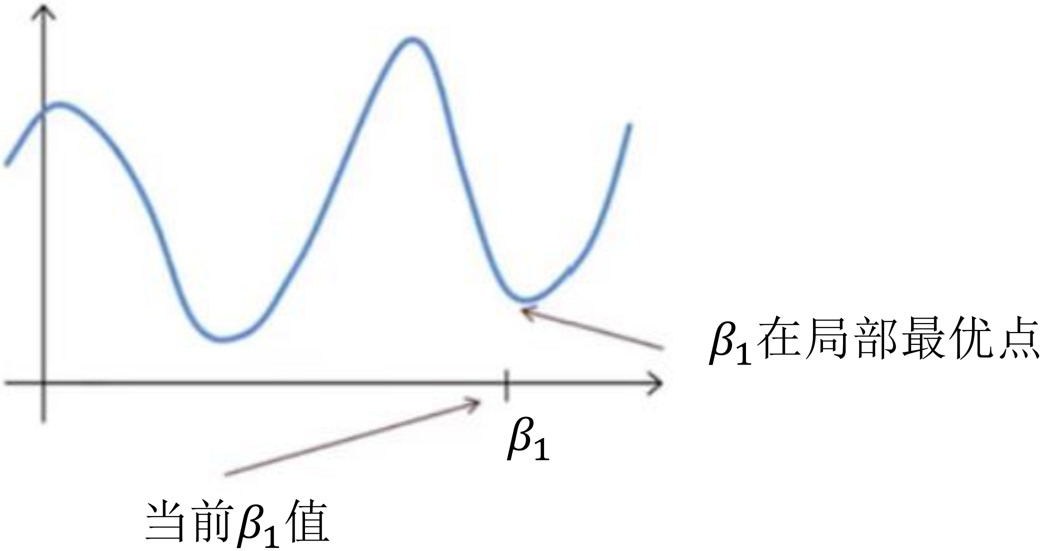
$$\beta_i := \beta_i - \alpha \frac{\partial}{\partial \beta_i} \mathrm{J}(\beta_0, \beta_1), \quad i=0,1$$

取值大小影响收敛速度

梯度下降法

局部最低点：

$$\beta_1 := \beta_1 - \alpha \frac{\partial}{\partial \beta_1} J(\beta_1)$$



接近局部最优点时，梯度下降法会自动减小步长，所有没必要另外减小α。

梯度下降法

线性回归模型：

$$h_\beta(x) = \beta_0 + \beta_1 \mathrm{x}$$

$$\mathrm{J}(\beta_0, \beta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\beta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

$$\beta_i := \beta_i - \alpha \frac{\partial}{\partial \beta_i} \mathrm{J}(\beta_0, \beta_1), \quad i=0,1$$

$$\min_{\beta_0, \beta_1} \mathrm{J}(\beta_0, \beta_1)$$

梯度下降法

线性回归模型：

$$\frac{\partial}{\partial \beta_i}\mathrm{J}(\beta_0,\beta_1)=?$$

梯度下降法

梯度下降算法:

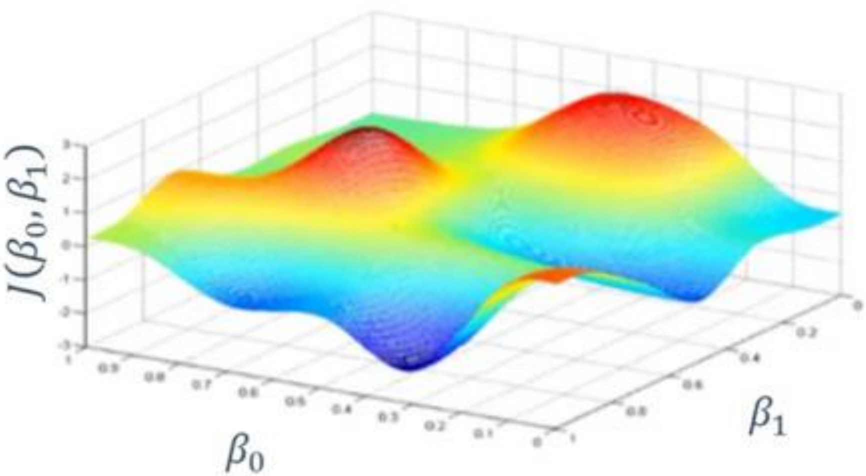
Repeat until convergence {

$$\beta_0 := \beta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\beta\left(x^{(i)}\right) - y^{(i)} \right)$$

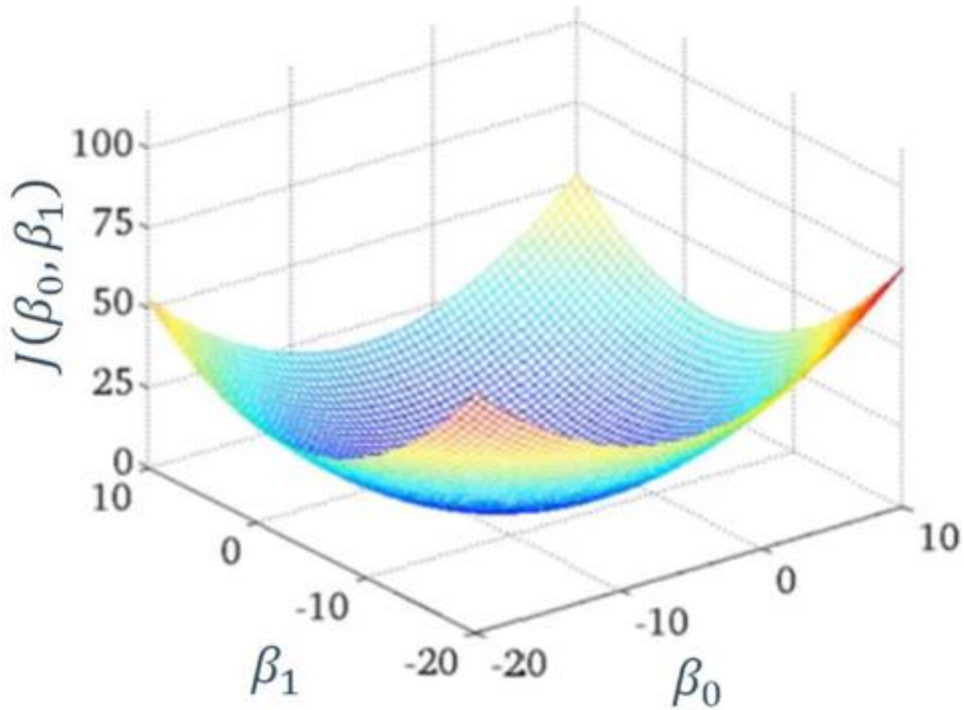
$$\beta_1 := \beta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\beta\left(x^{(i)}\right) - y^{(i)} \right) \cdot x^{(i)}$$

}

梯度下降法



$J(\beta_0, \beta_1)$
β_0
β_1

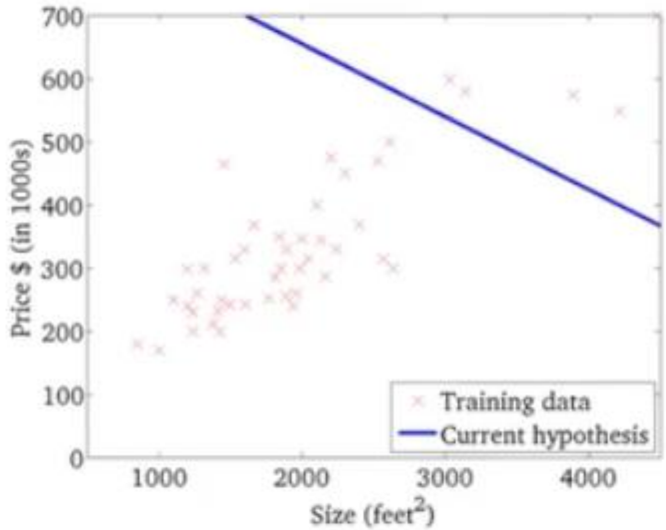


100
75
50
25
0
$J(\beta_0, \beta_1)$
10
0
-10
-20
β_1
-20
-10
0
10
β_0

梯度下降法

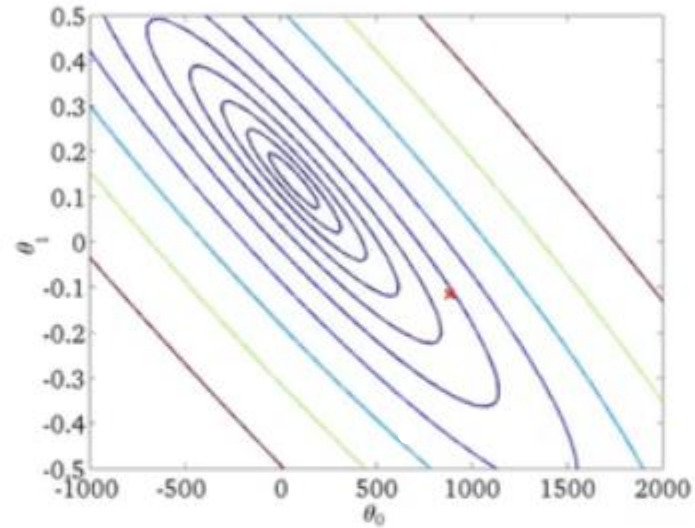
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

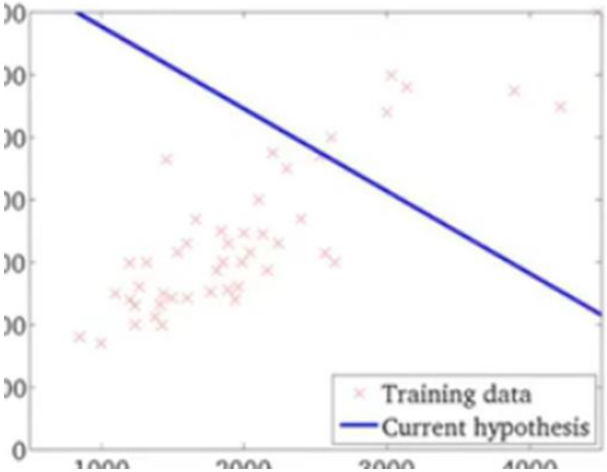
(function of the parameters θ_0, θ_1)



梯度下降法

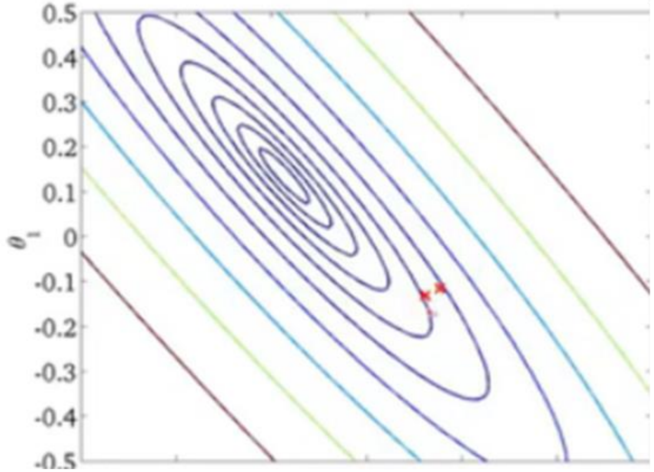
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

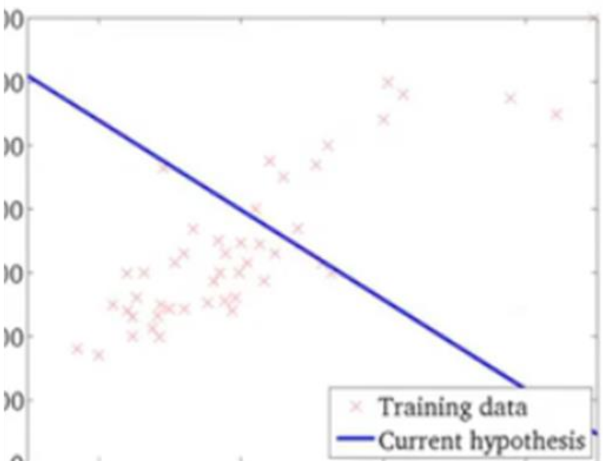
(function of the parameters θ_0, θ_1)



梯度下降法

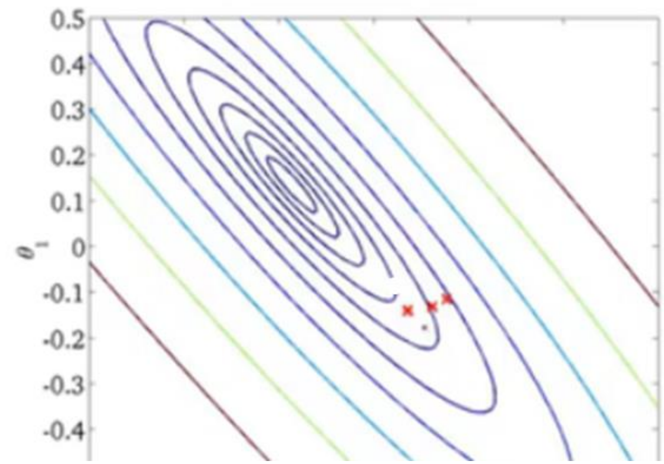
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

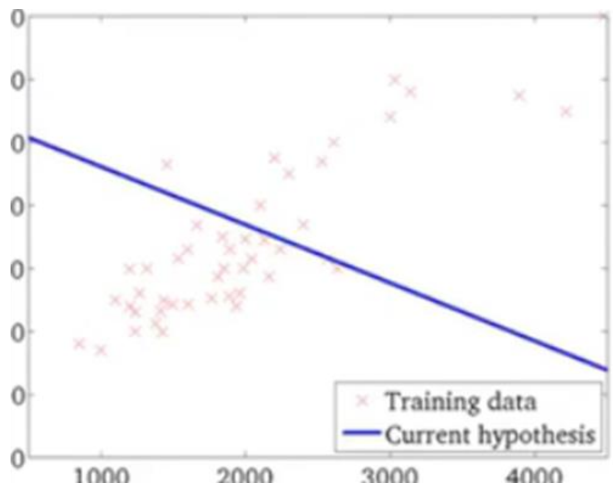
(function of the parameters θ_0, θ_1)



梯度下降法

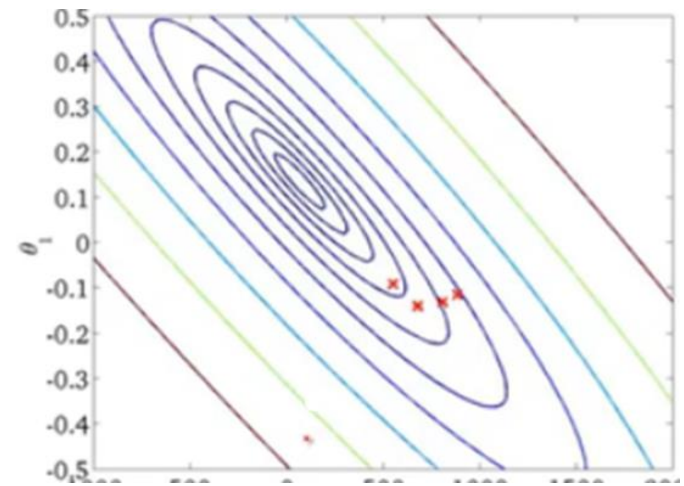
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

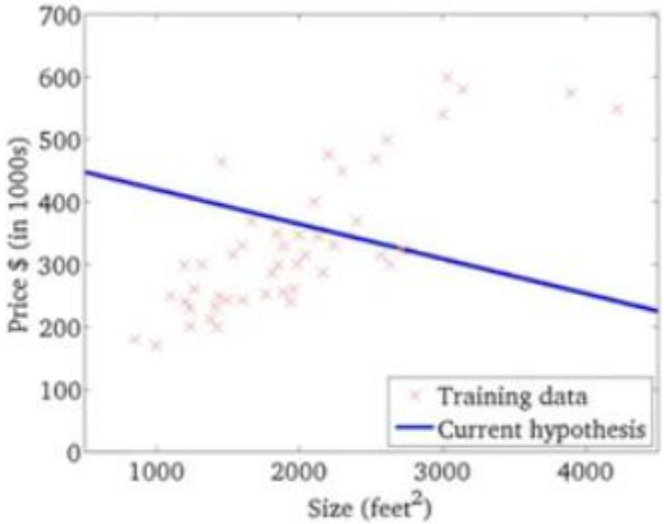
(function of the parameters θ_0, θ_1)



梯度下降法

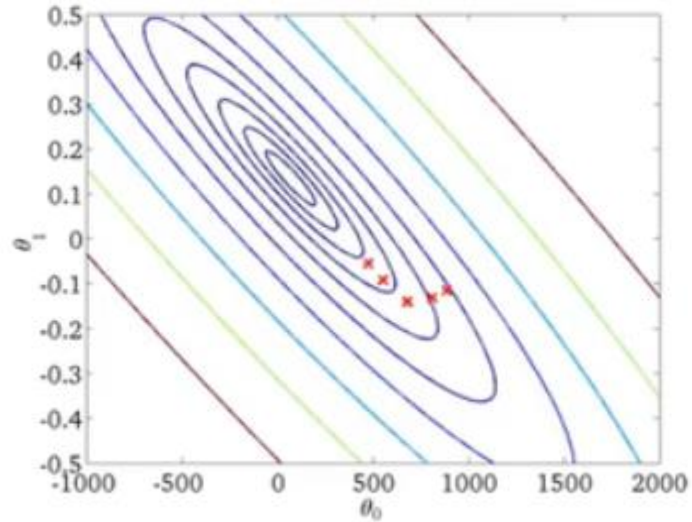
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

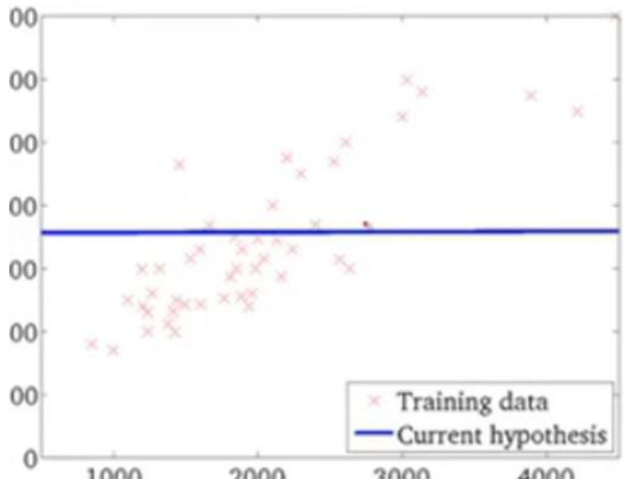
(function of the parameters θ_0, θ_1)



梯度下降法

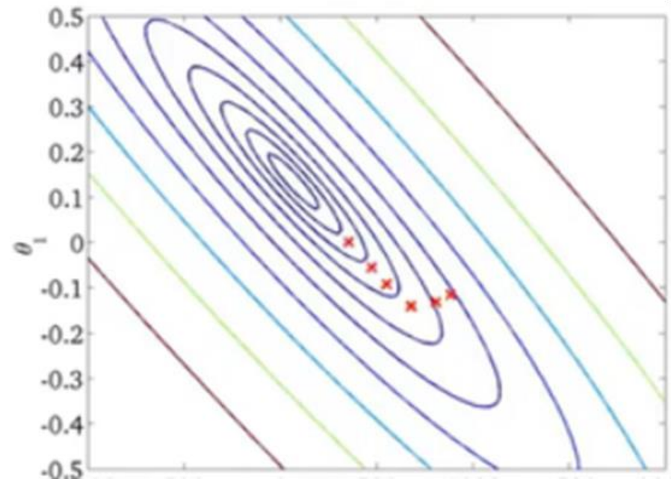
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

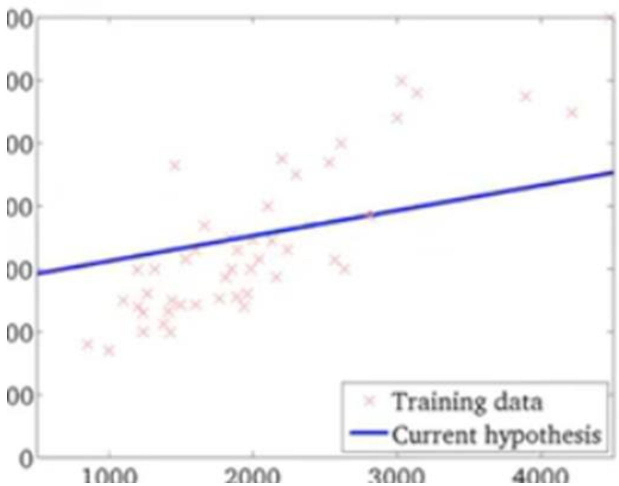
(function of the parameters θ_0, θ_1)



梯度下降法

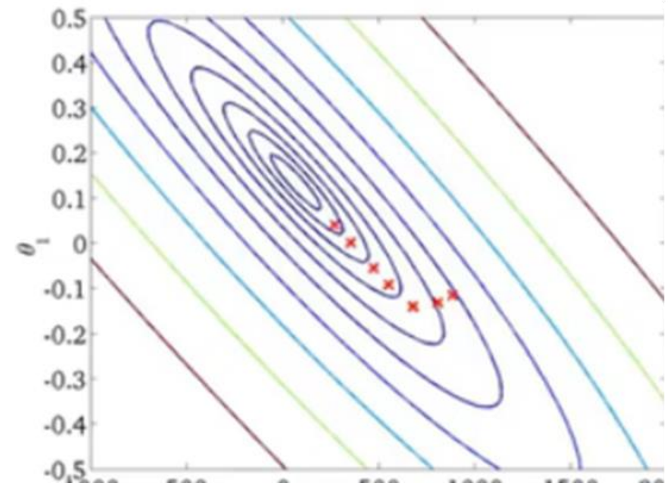
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

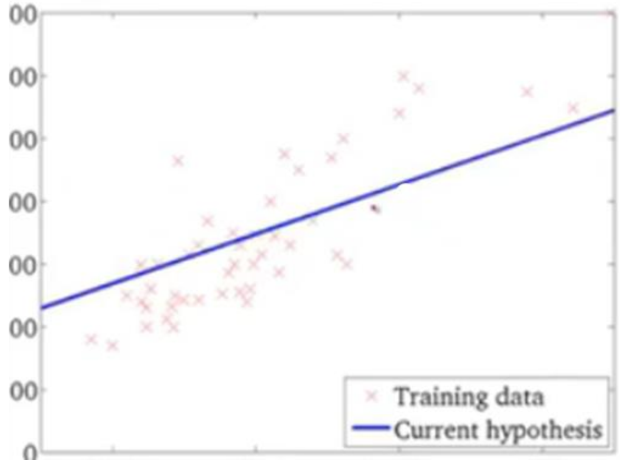
(function of the parameters θ_0, θ_1)



梯度下降法

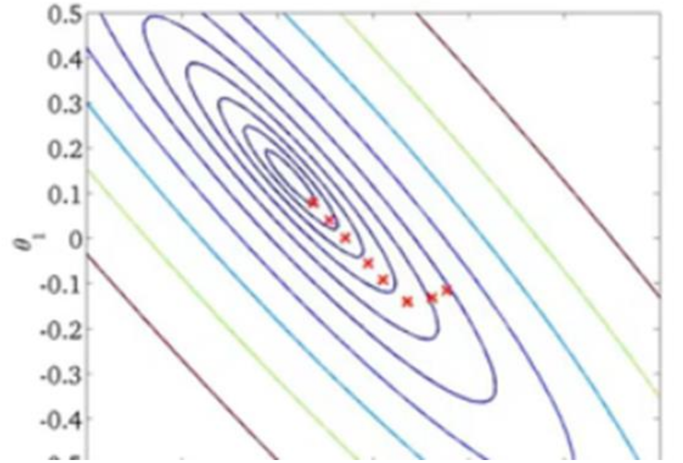
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

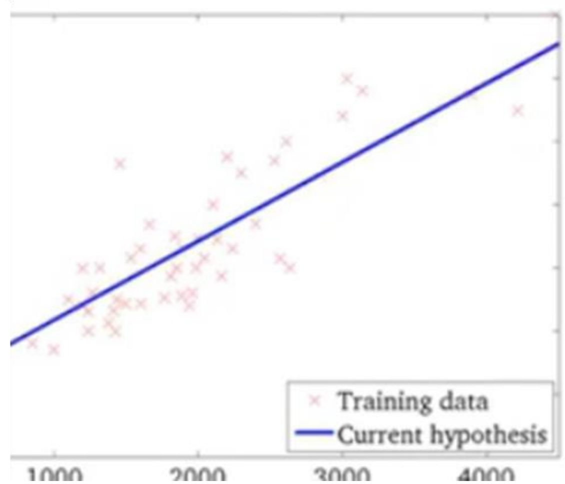
(function of the parameters θ_0, θ_1)



梯度下降法

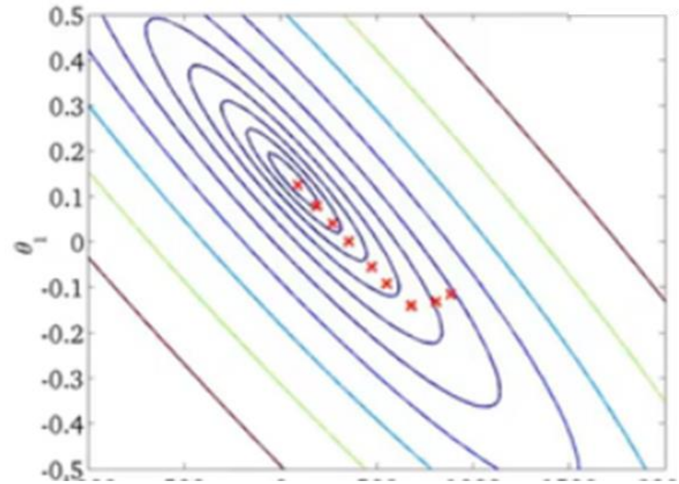
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)



增加多项式特征

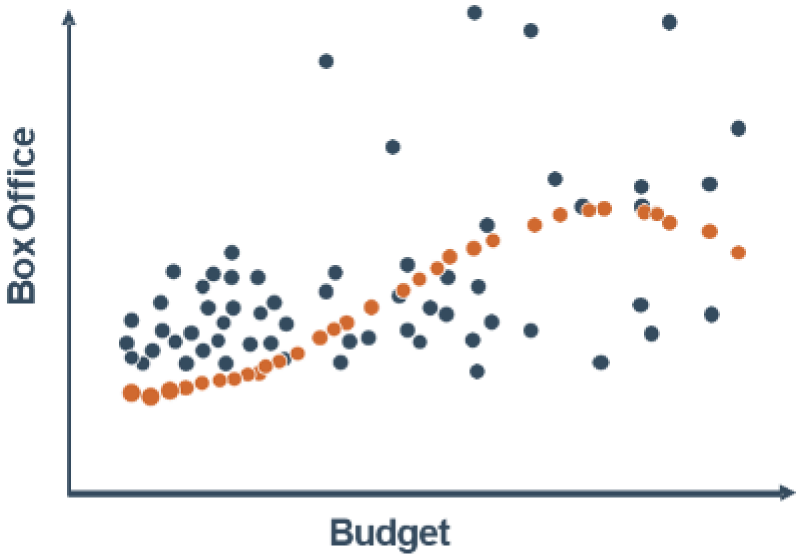
- 通过增加多项式特征来捕捉更高阶的数据特征

$$y_\beta(x) = \beta_0 + \beta_1 x + \beta_2 x^2$$

增加多项式特征

- 通过增加多项式特征来捕捉更高阶的数据特征
- “线性回归”意味着特征间的线性组合

$$y_\beta(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3$$



增加多项式特征

- 通过增加多项式特征来捕捉更高阶的数据特征
- “线性回归”意味着特征间的线性组合

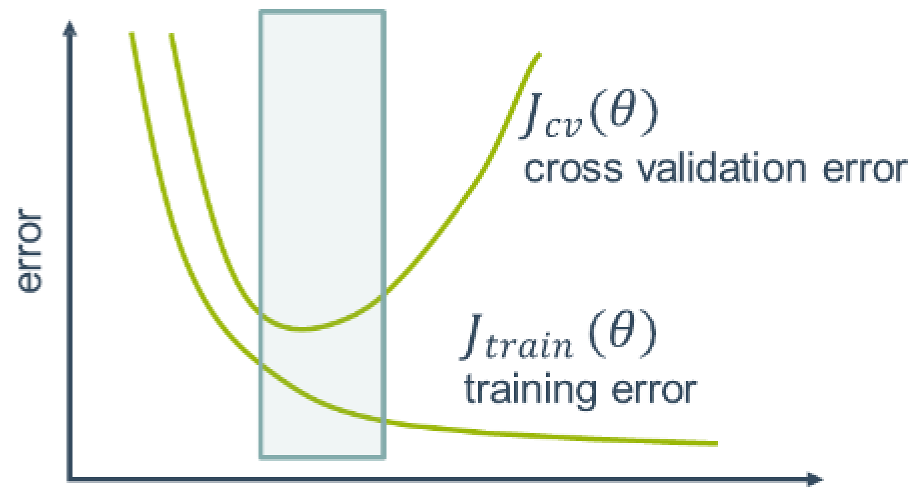
$$y_\beta(x) = \beta_0 + \beta_1 \log(x)$$

增加多项式特征

- 可以选择变量间的交互项：　$y_\beta(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2$

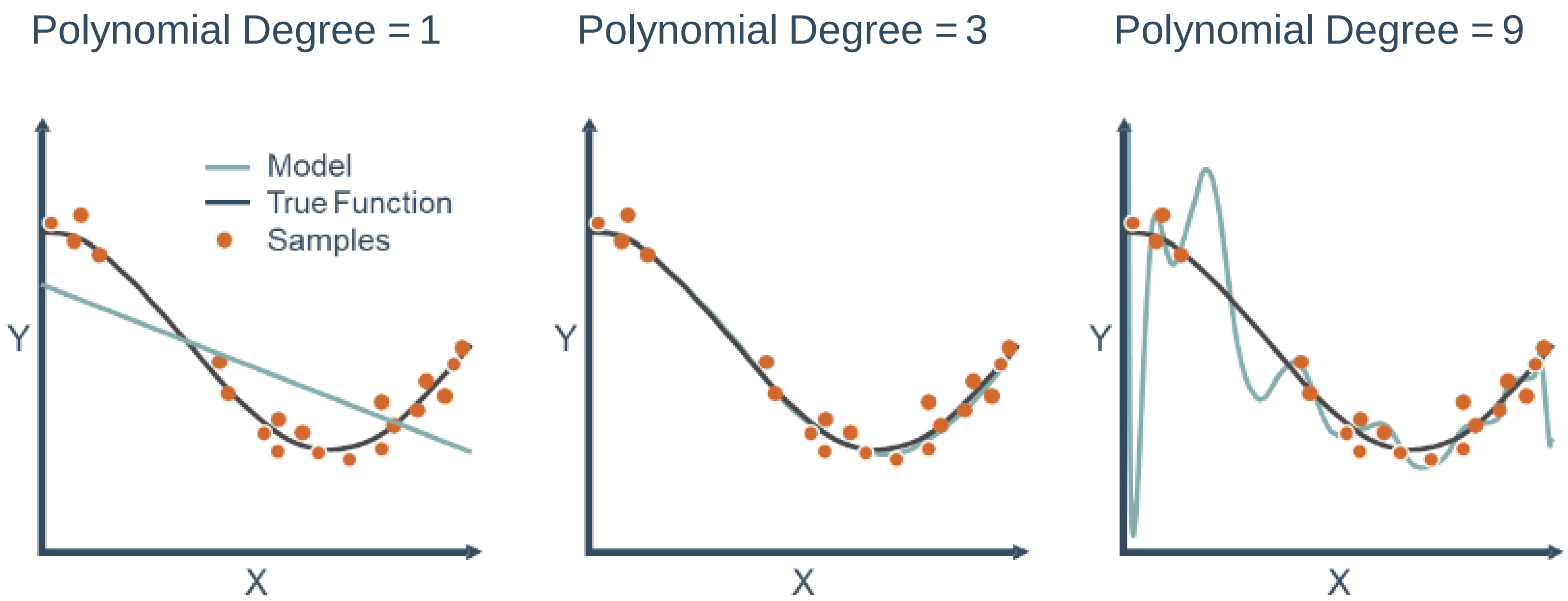
- 如何选择正确的函数形式：➡检查每个变量与结果之间的关系

模型复杂度与误差



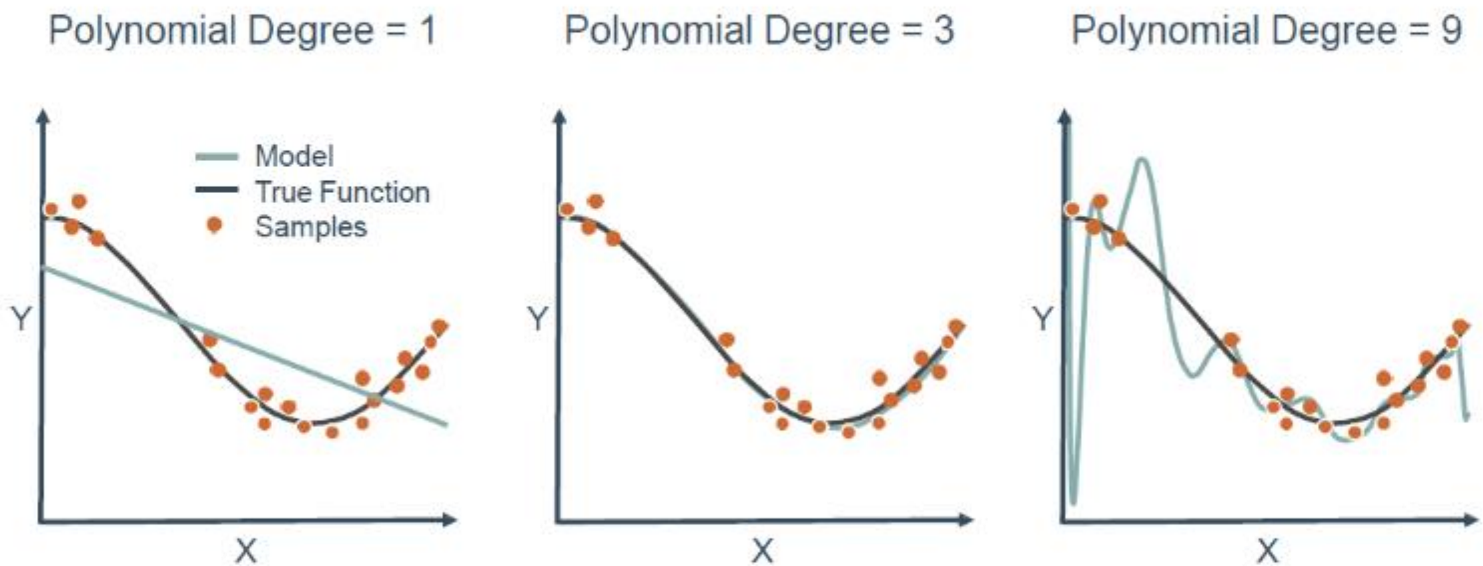
error
$J_{cv}(\theta)$
cross validation error
$J_{train}(\theta)$
training error

防止欠拟合与过拟合



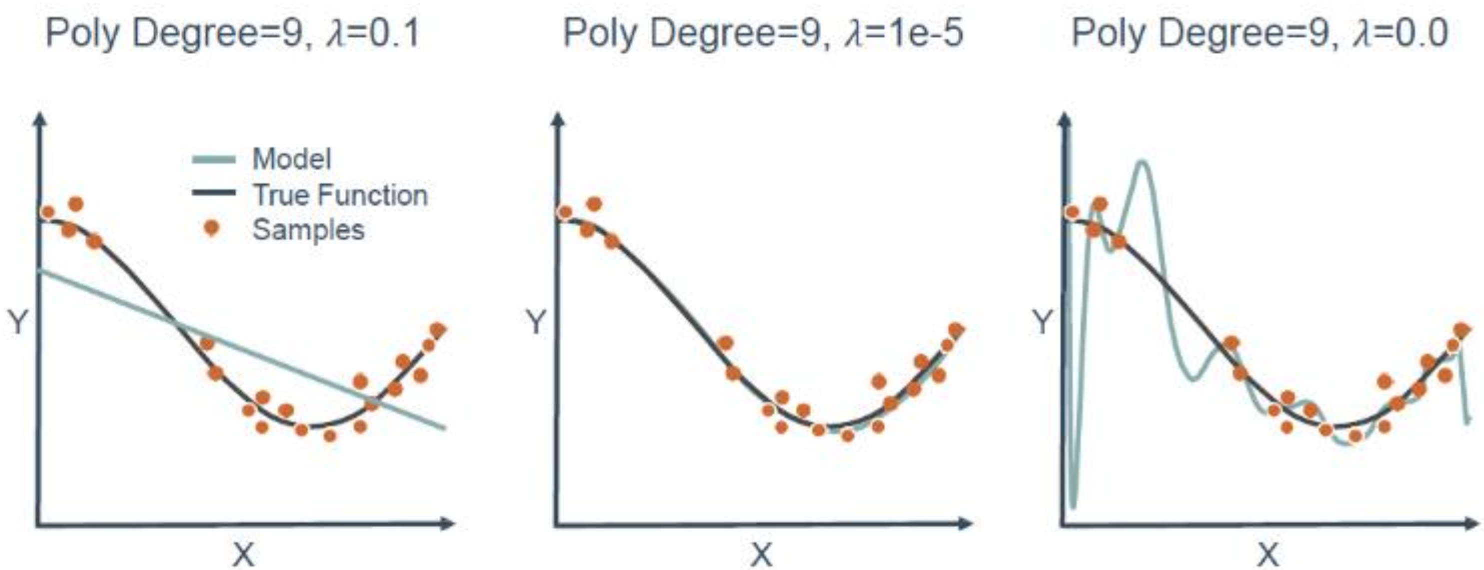
如何用一个9次多项式拟合数据，并防止过拟合？

防止欠拟合与过拟合



$$J(\beta) = \frac{1}{2m}\sum_{i=1}^{m}(y_\beta(x^{(i)}) - y^{(i)})^2$$

正则化（regularization）



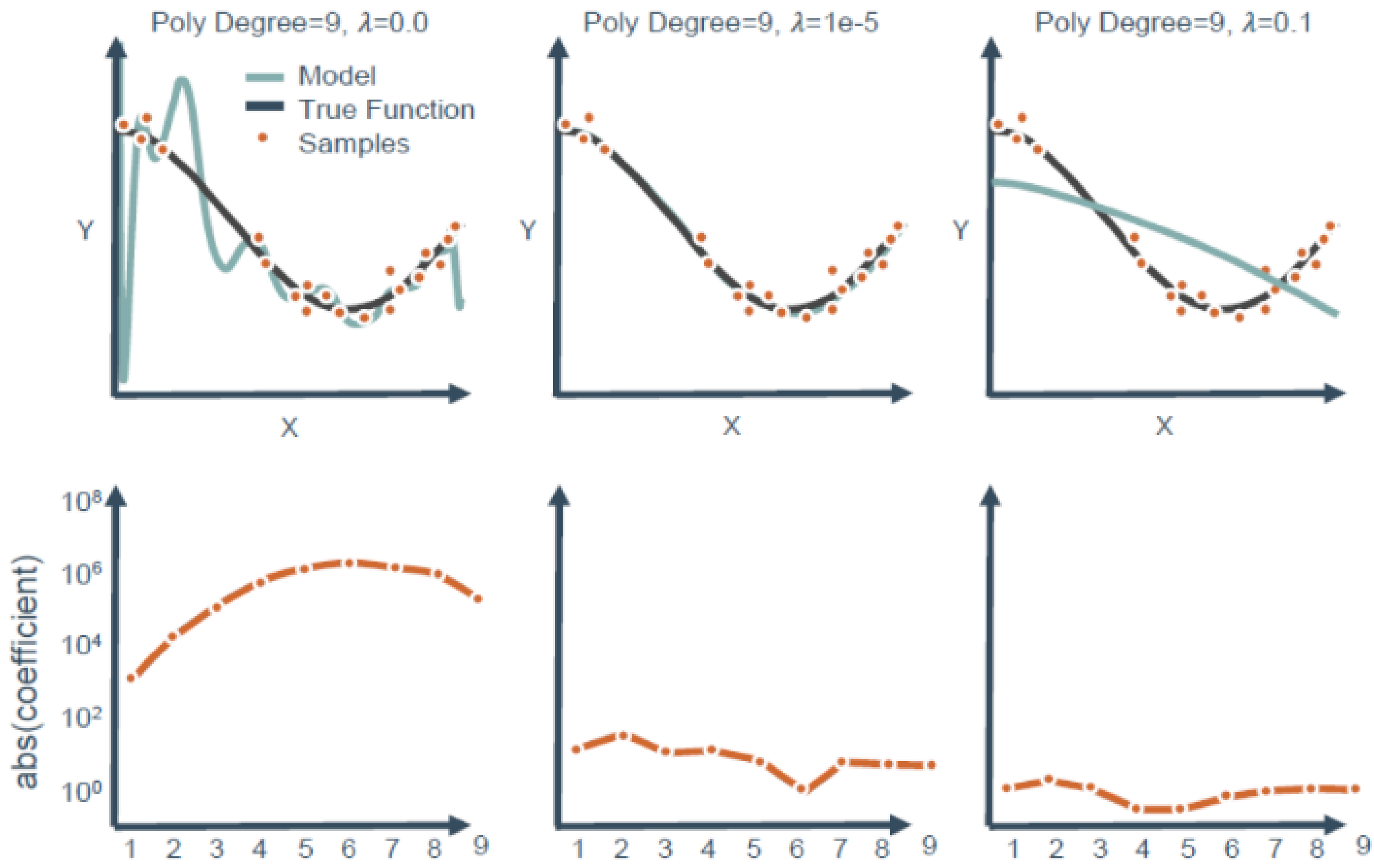
$$J(\beta) = \frac{1}{2m}\sum_{i=1}^{m}\left(y_\beta\left(x^{(i)}\right) - y^{(i)}\right)^2 + \lambda\sum_{j=1}^{n}\beta_j^2$$

岭回归（Ridge Regression）(L2)

$$J(\beta)=\frac{1}{2m}\sum_{i=1}^{m}(y_{\beta}\left(x^{(i)}\right)-y^{(i)})^2+\lambda\sum_{j=1}^{n}\beta_j^2$$

- 惩罚项收缩了所有系数的大小
- 越大的系数被惩罚得越多，因为惩罚的是平方

岭回归对模型参数的效果



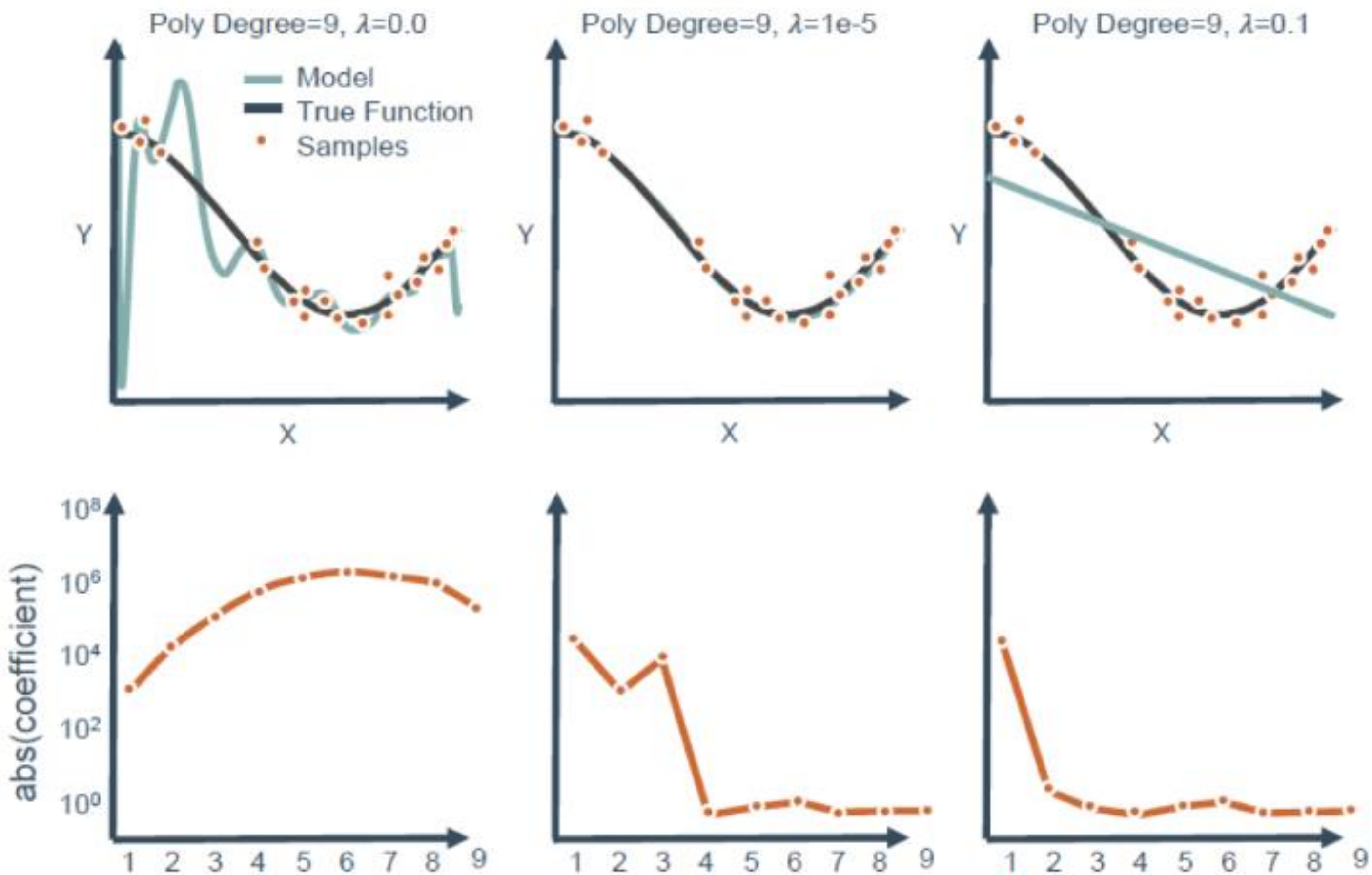
$$J(\beta) = \frac{1}{2m}\sum_{i=1}^{m}\left(y_\beta\left(x^{(i)}\right) - y^{(i)}\right)^2 + \lambda\sum_{j=1}^{n}\beta_j^2$$

套索回归（Lasso Regression）(L1)

$$J(\beta) = \frac{1}{2m}\sum_{i=1}^{m}(y_\beta(x^{(i)}) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}|\beta_j|$$

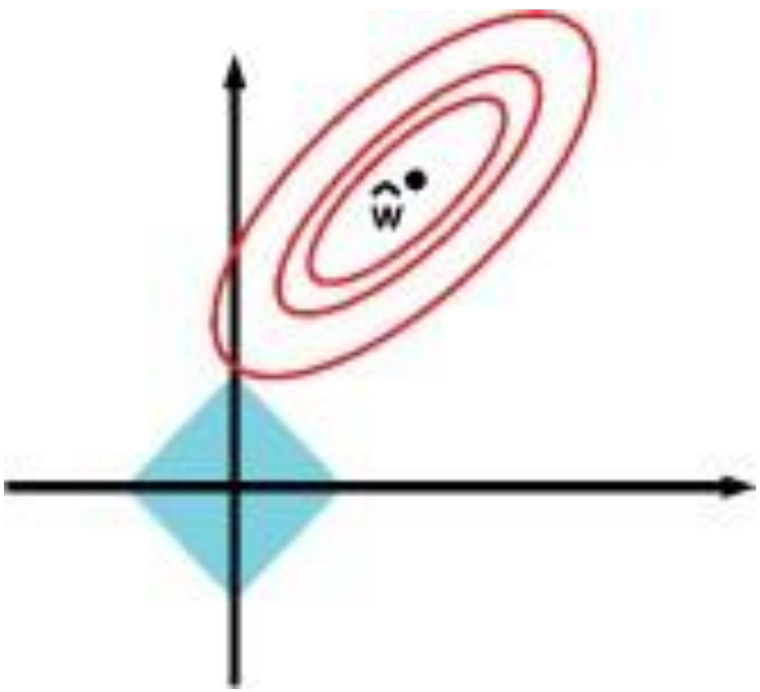
- 惩罚项有选择地收缩了某些系数
- 可以被用来做特征选择
- 比岭回归收敛速度慢

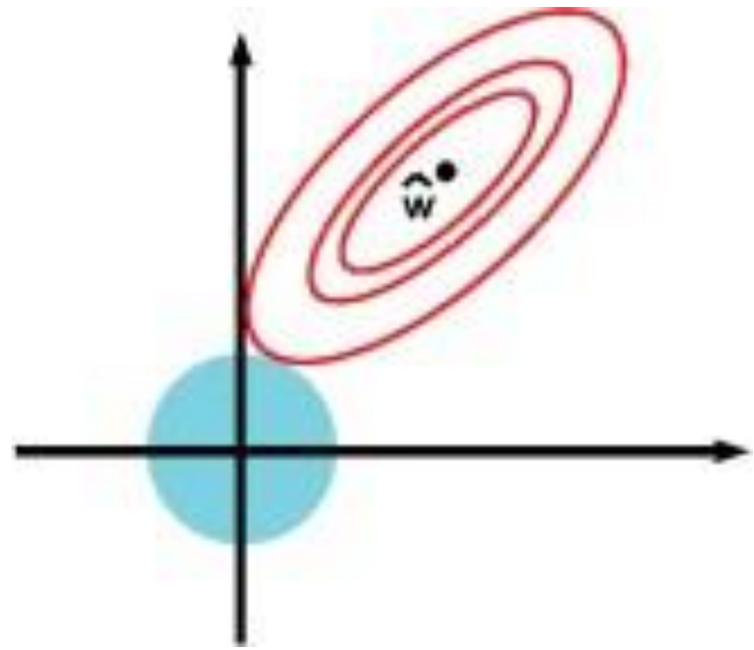
套索回归对模型参数的效果



$$J(\beta)=\frac{1}{2m}\sum_{i=1}^{m}\left(y_{\beta}\left(x^{(i)}\right)-y^{(i)}\right)^2+\lambda\sum_{j=1}^{n}|\beta_j|$$

L1与L2正则化





ElasticNet正则化

$$J(\beta) = \frac{1}{2m}\sum_{i=1}^{m}(y_\beta(x^{(i)}) - y^{(i)})^2 + \lambda_1 \sum_{j=1}^{n} |\beta_j| + \lambda_2 \sum_{j=1}^{n} \beta_j^2$$

- 岭回归和套索回归的综合，用以平衡稀疏和平滑两个问题
- 需要调节额外的参数，来分配L1和L2正则化惩罚项的比例

超参数及其优化

- 正则化系数（λ_1 和λ_2）是根据经验决定的

使用测试数据调节 λ?

超参数及其优化

- 正则化系数（λ_1 和λ_2）是根据经验决定的

测试数据集来调节λ_1 和λ_2

使用测试数据集调节 λ?

超参数及其优化

- 正则化系数（λ_1 和λ_2）是根据经验决定的
- 想让模型泛化---不要使用测试数据集来调节λ_1 和λ_2
- 划分出另一个数据集来调节超参数---验证集（validation set）

用交叉验证来调节λ

线性回归实例

——李宏毅2020机器学习深度学习

线性回归实例

Step 1: Model

$y = b + w \cdot x_{cp}$

A set of function

Model

$f_1, f_2 \cdots$

w and b are parameters
(can be any value)

f_1: $y = 10.0 + 9.0 \cdot x_{cp}$

f_2: $y = 9.8 + 9.2 \cdot x_{cp}$

f_3: $y = -0.8 - 1.2 \cdot x_{cp}$

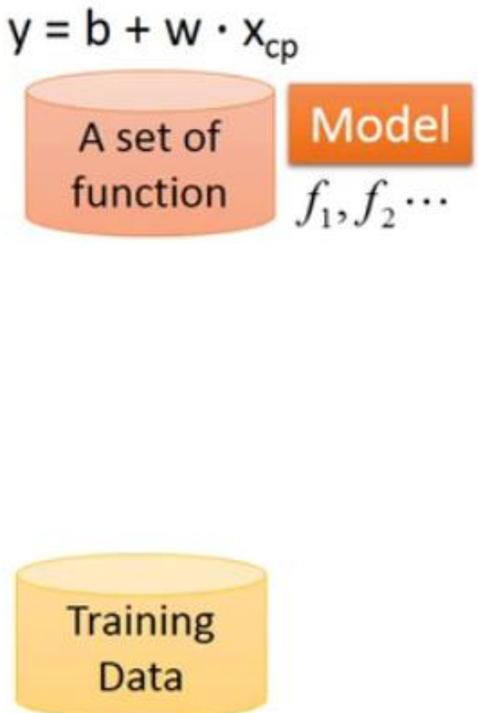
……

$f($ 

$x) =$ CP after evolution y

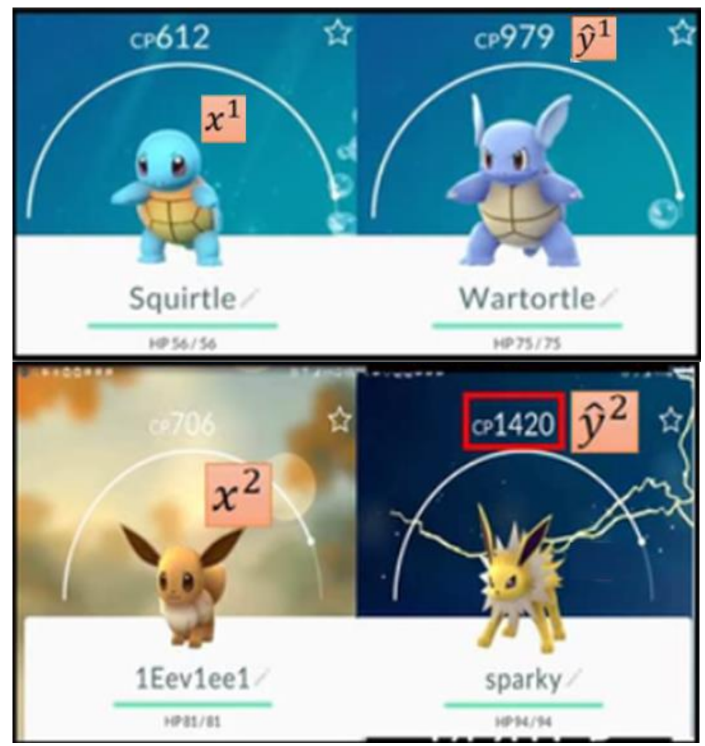
Linear model: $y = b + \sum w_i x_i$

线性回归实例



$y = b + w \cdot x_{cp}$
A set of function
Model
$f_1, f_2 \cdots$
Training Data

输入 输出



CP612
x^1
Squirtle
HP56/56
CP979
$\hat{y}^1$
Wartortle
HP75/75
CP706
x^2
1Eev1ee1
HP81/81
CP1420
$\hat{y}^2$
sparky
HP94/94

线性回归实例

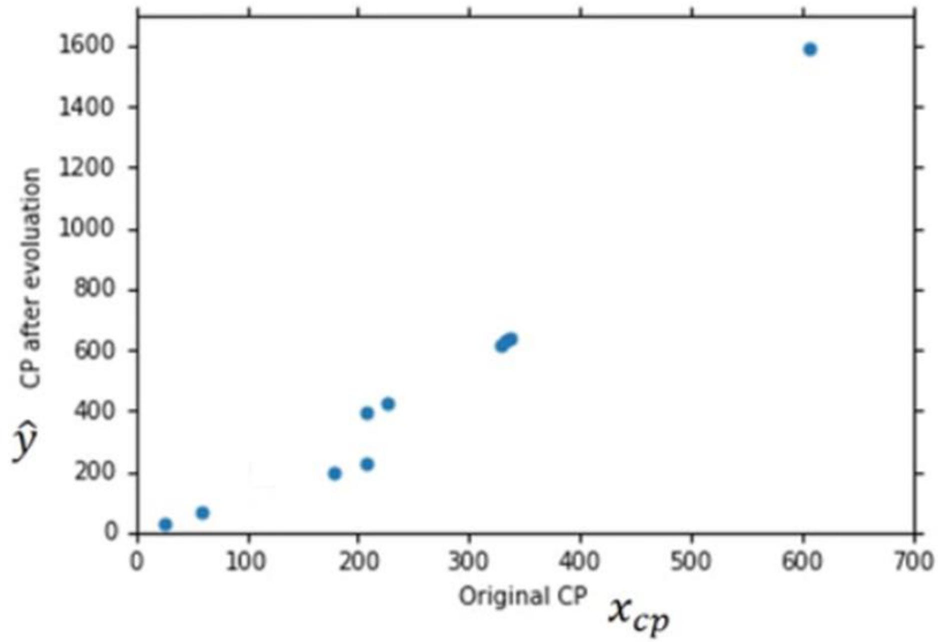
Training Data:
10 pokemons

$(x^1, \hat{y}^1)$

$(x^2, \hat{y}^2)$

⋮

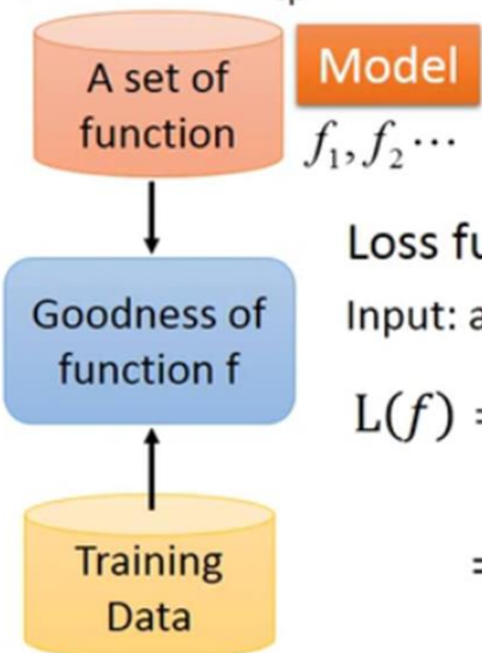
$(x^{10}, \hat{y}^{10})$



线性回归实例

Step 2: Goodness of Function

$y = b + w \cdot x_{cp}$



Loss function L:

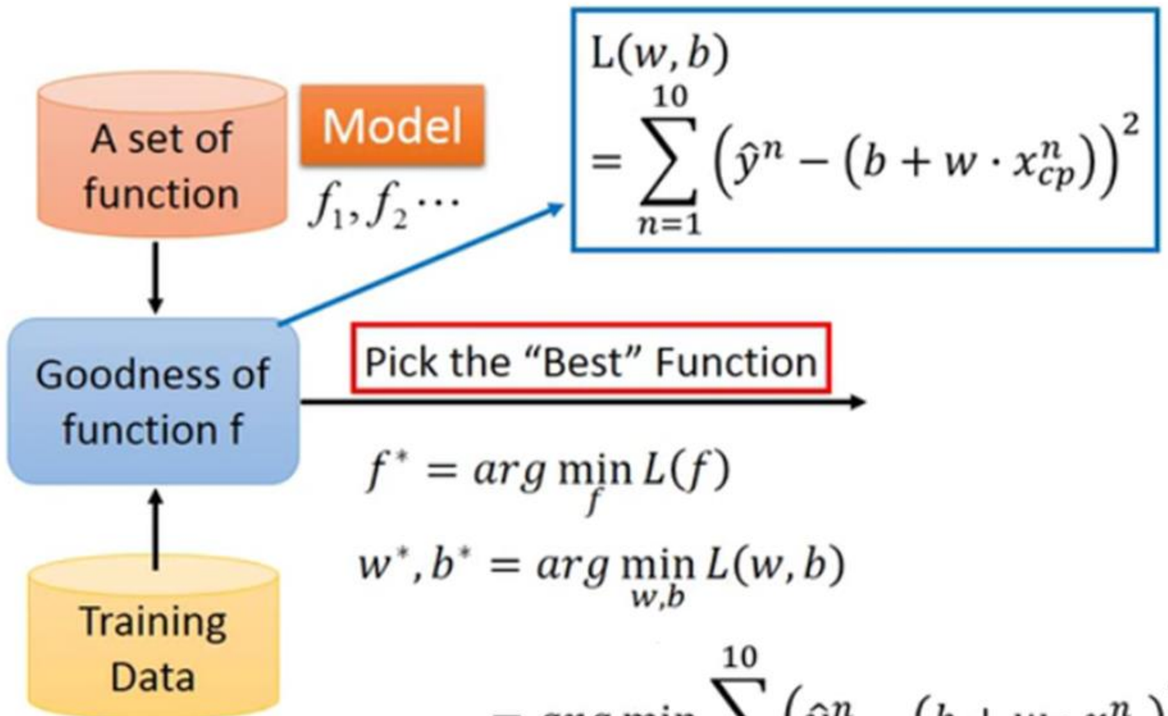
Input: a function, output: how bad it is

$$\mathrm{L}(f) = \mathrm{L}(w, b)$$

$$= \sum_{n=1}^{10} \left(\hat{y}^n - (b + w \cdot x_{cp}^n)\right)^2$$

线性回归实例

Step 3: Best Function



$$L(w,b) = \sum_{n=1}^{10}\left(\hat{y}^n - (b + w \cdot x_{cp}^n)\right)^2$$

$$f^* = arg \min_f L(f)$$

$$w^*, b^* = arg \min_{w,b} L(w,b)$$

$$= arg \min_{w,b} \sum_{n=1}^{10}\left(\hat{y}^n - (b + w \cdot x_{cp}^n)\right)^2$$

线性回归实例

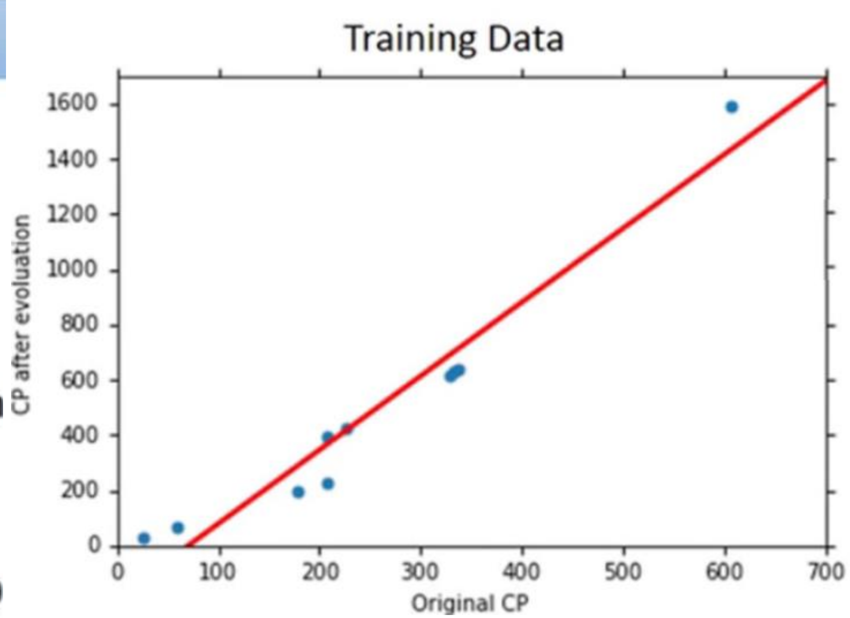
$y = b + w \cdot x_{cp}$

b = -188.4

w = 2.7

Average Error on Training Data

$$= \sum_{n=1}^{10} e^n = 31.9$$



线性回归实例

How's the results?
- Generalization

What we really care about is the error on new data (testing data)

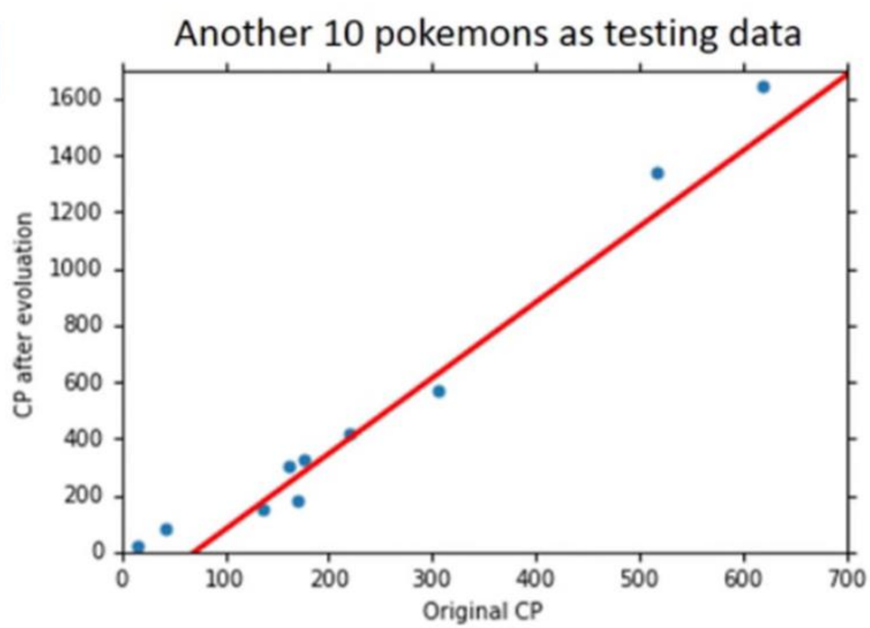
$y = b + w \cdot x_{cp}$

b = -188.4

w = 2.7

Average Error on Testing Data

$$= \sum_{n=1}^{10} e^n = 35.0$$



线性回归实例

Selecting another Model

$$y = b + w_1 \cdot x_{cp} + w_2 \cdot (x_{cp})^2$$

Best Function

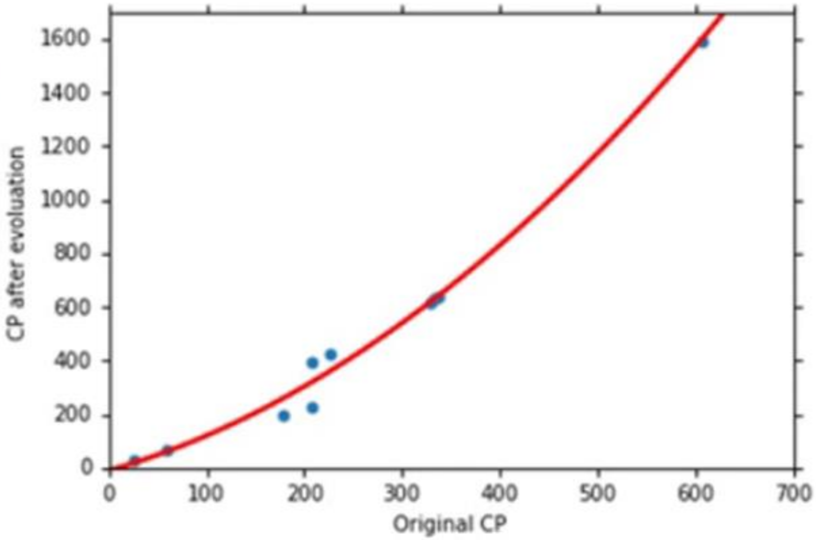
$b = -10.3$

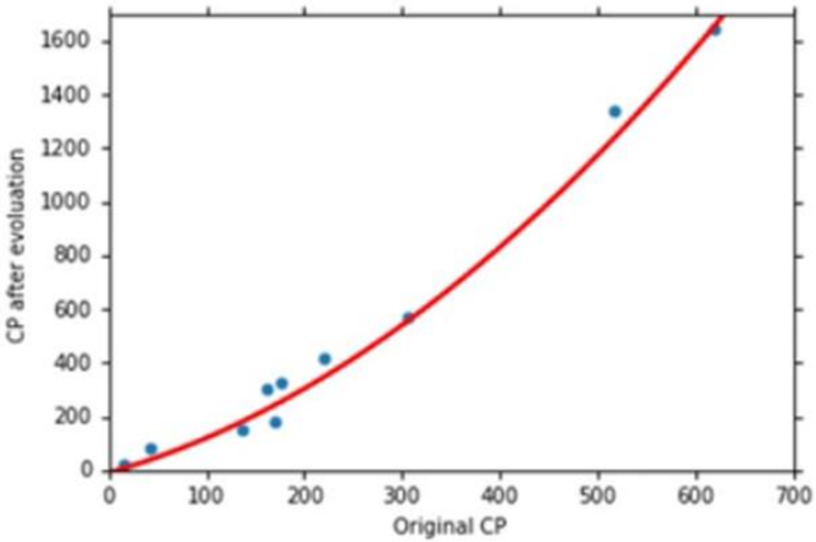
$w_1 = 1.0$, $w_2 = 2.7 \times 10^{-3}$

Average Error = 15.4

Testing:

Average Error = 18.4





线性回归实例

Selecting another Model

$$y = b + w_1 \cdot x_{cp} + w_2 \cdot (x_{cp})^2 + w_3 \cdot (x_{cp})^3$$

Best Function

$b = 6.4$, $w_1 = 0.66$

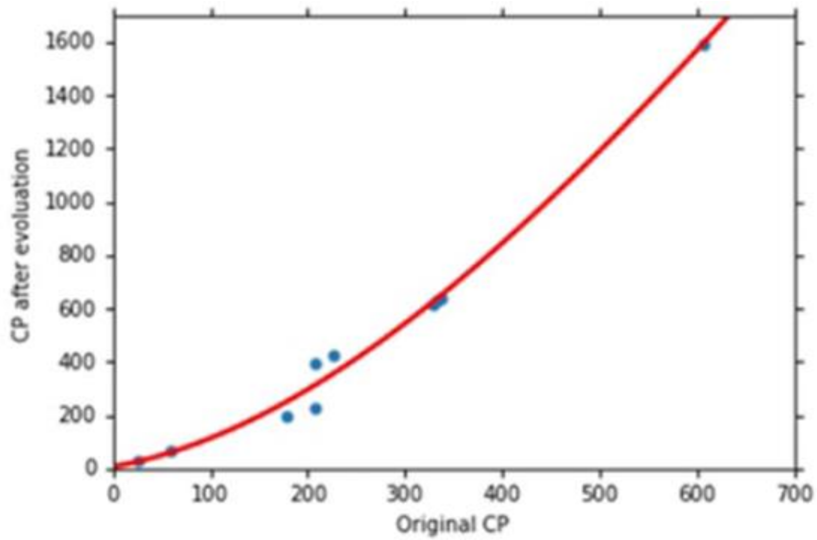
$w_2 = 4.3 \times 10^{-3}$

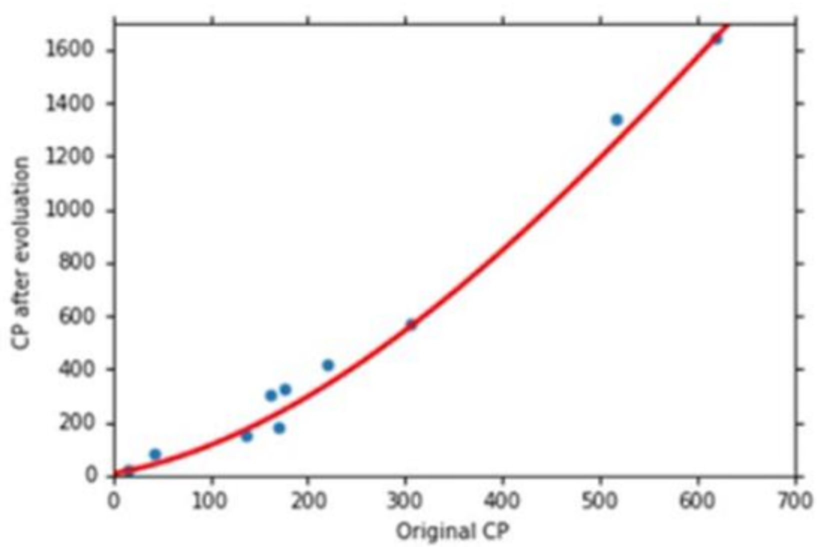
$w_3 = -1.8 \times 10^{-6}$

Average Error = 15.3

Testing:

Average Error = 18.1

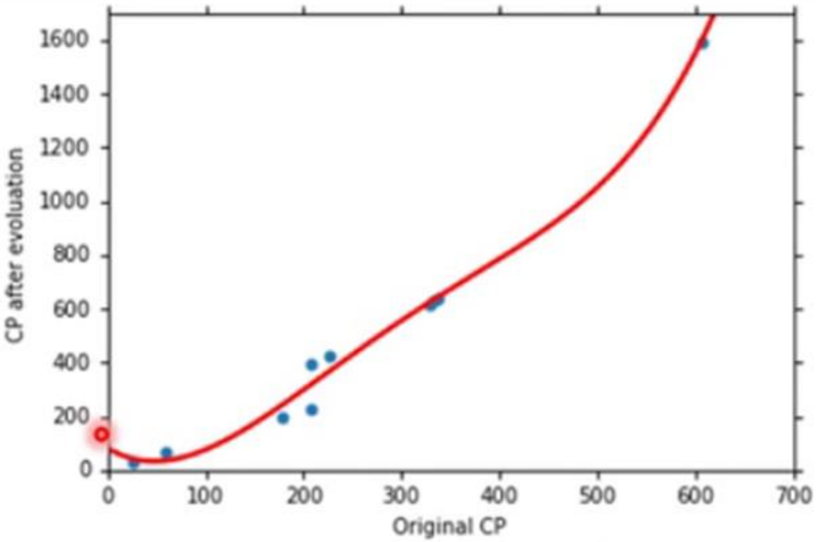




线性回归实例

Selecting another Model

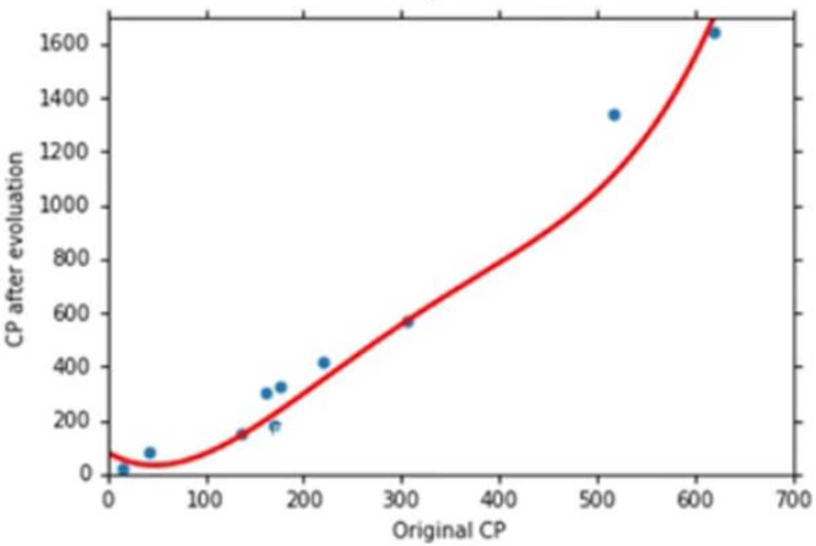
$$y = b + w_1 \cdot x_{cp} + w_2 \cdot (x_{cp})^2 + w_3 \cdot (x_{cp})^3 + w_4 \cdot (x_{cp})^4$$



Best Function

Average Error = 14.9

Testing:

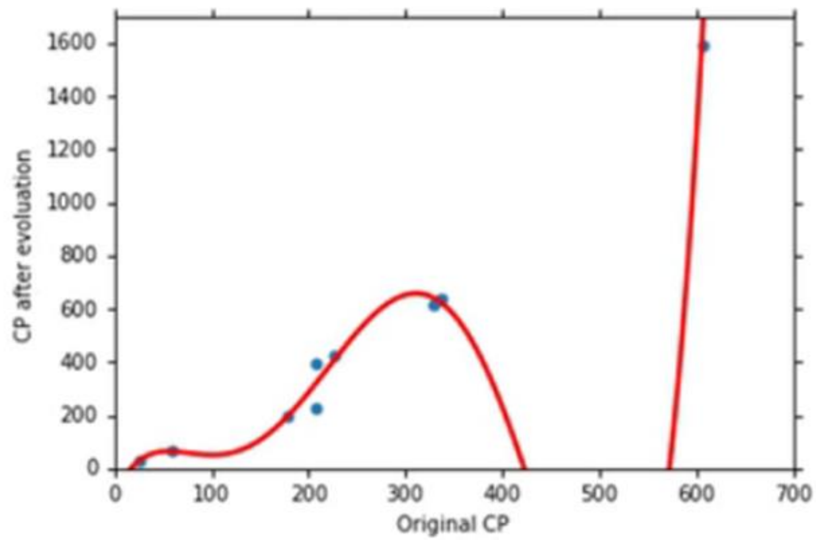


Average Error = 28.8

线性回归实例

Selecting another Model

$$y = b + w_1 \cdot x_{cp} + w_2 \cdot (x_{cp})^2 + w_3 \cdot (x_{cp})^3 + w_4 \cdot (x_{cp})^4 + w_5 \cdot (x_{cp})^5$$

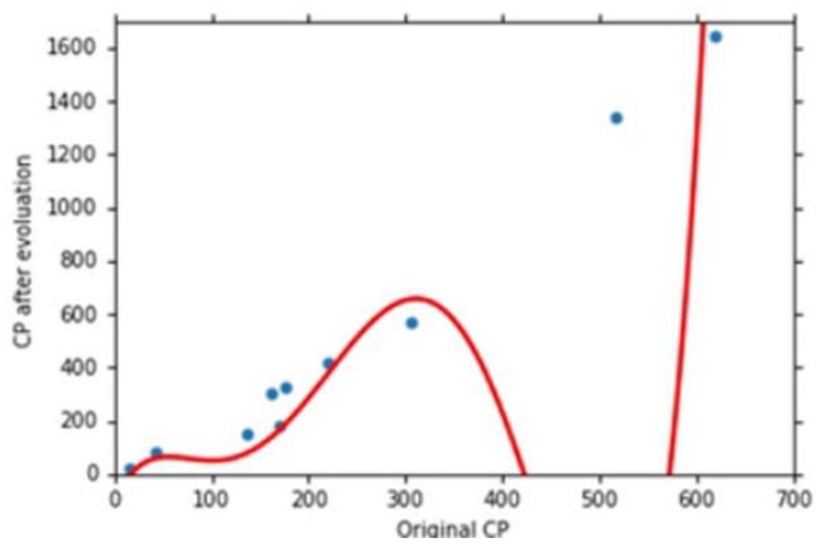


Best Function

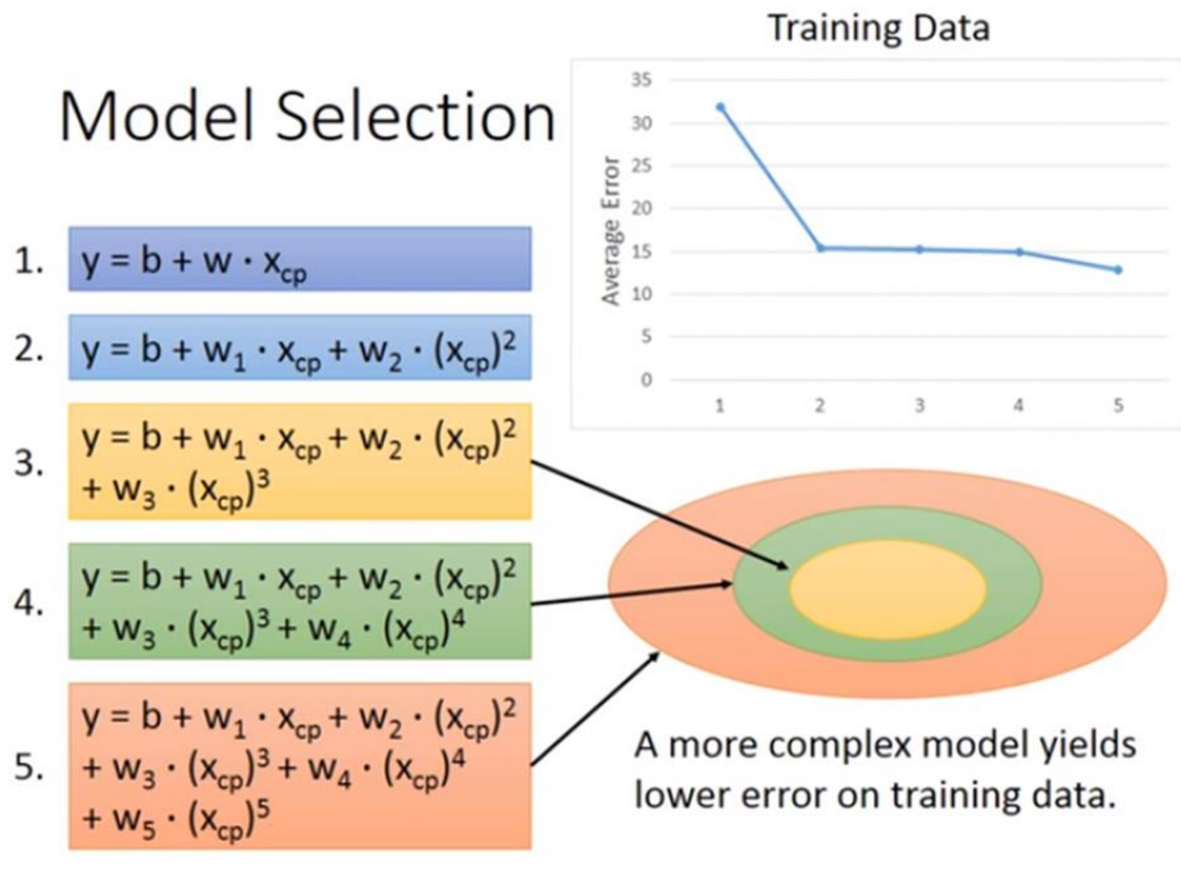
Average Error = 12.8

Testing:

Average Error = 232.1

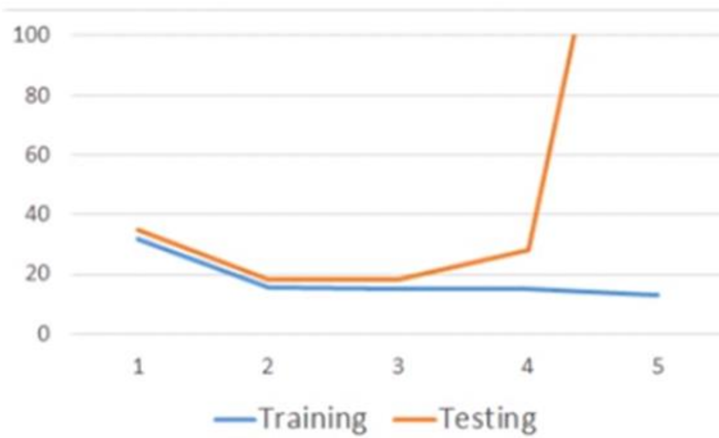


线性回归实例


Training Data
Model Selection
1. $y = b + w \cdot x_{cp}$
2. $y = b + w_1 \cdot x_{cp} + w_2 \cdot (x_{cp})^2$
3. $y = b + w_1 \cdot x_{cp} + w_2 \cdot (x_{cp})^2 + w_3 \cdot (x_{cp})^3$
4. $y = b + w_1 \cdot x_{cp} + w_2 \cdot (x_{cp})^2 + w_3 \cdot (x_{cp})^3 + w_4 \cdot (x_{cp})^4$
5. $y = b + w_1 \cdot x_{cp} + w_2 \cdot (x_{cp})^2 + w_3 \cdot (x_{cp})^3 + w_4 \cdot (x_{cp})^4 + w_5 \cdot (x_{cp})^5$
Average Error
35
30
25
20
15
10
5
0
1
2
3
4
5
A more complex model yields lower error on training data.

线性回归实例

Model Selection



	Training	Testing
1	31.9	35.0
2	15.4	18.4
3	15.3	18.1
4	14.9	28.2
5	12.8	232.1

线性回归实例

Model Selection

	Training	Testing
1	31.9	35.0
2	15.4	18.4
3	15.3	18.1
4	14.9	28.2
5	12.8	232.1

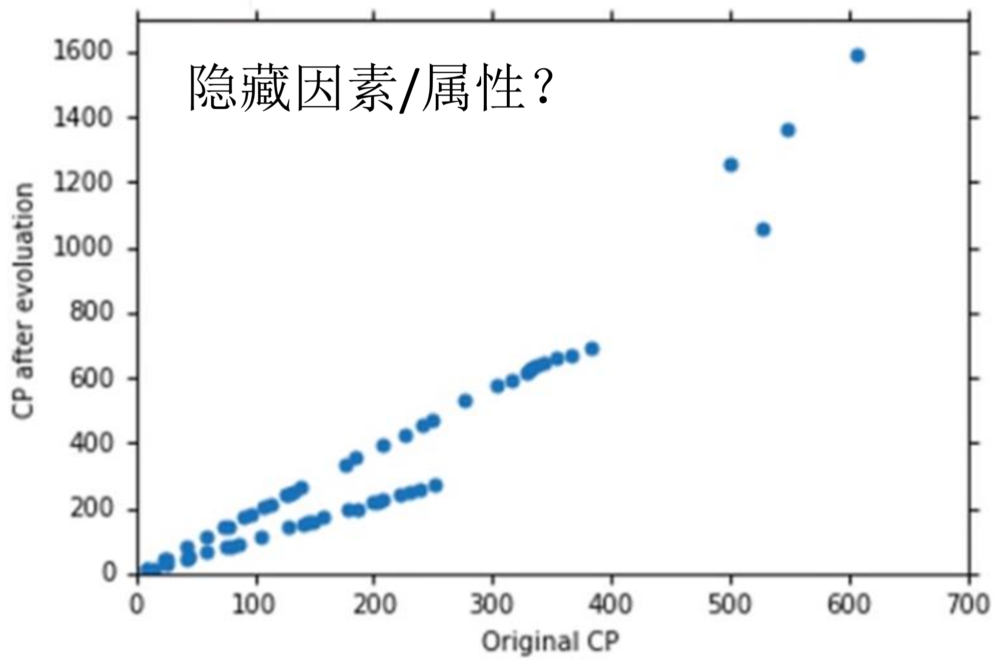
线性回归实例

Model Selection

	Training	Testing
1	31.9	35.0
2	15.4	18.4
3	15.3	18.1
4	14.9	28.2
5	12.8	232.1

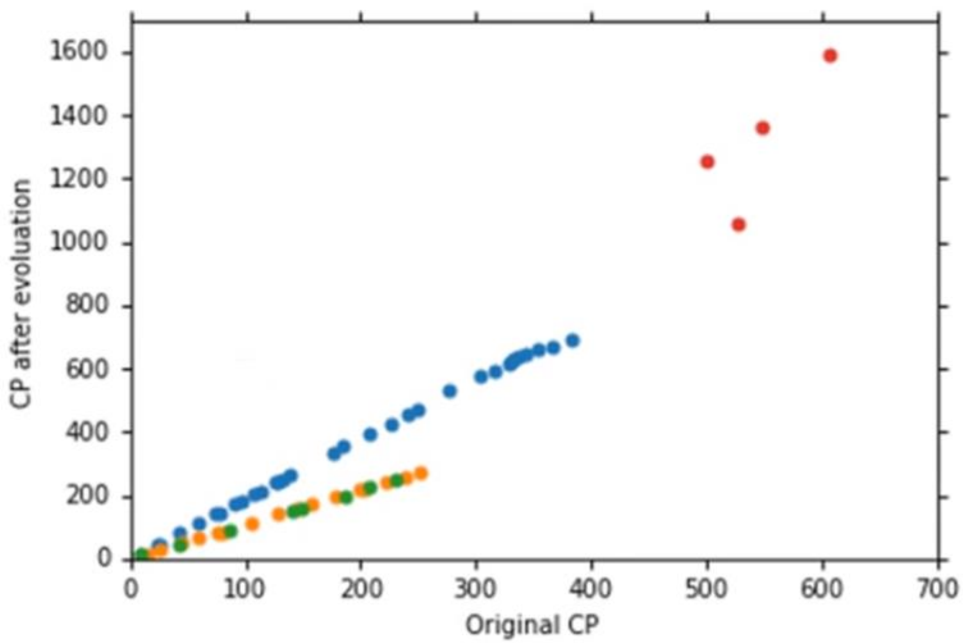
线性回归实例

Let’s collect more data（60只）



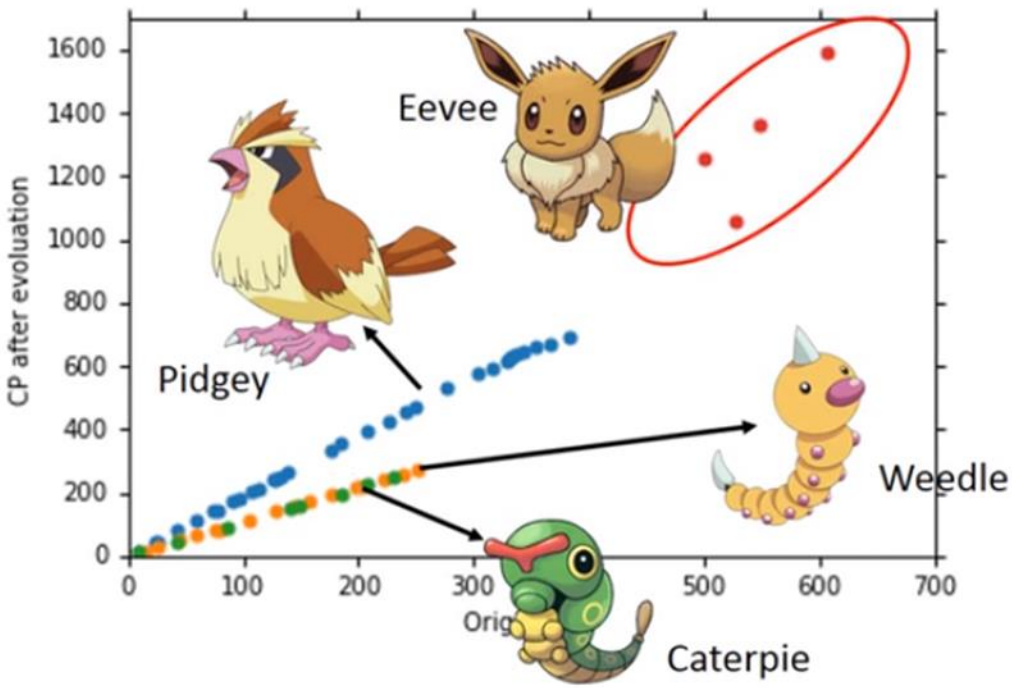
线性回归实例

What are the hidden factors?



线性回归实例

What are the hidden factors?



线性回归实例

Back to step 1: Redesign the Model

x_s = species of x

x

↓

If x_s = Pidgey:	$y = b_1 + w_1 \cdot x_{cp}$
If x_s = Weedle:	$y = b_2 + w_2 \cdot x_{cp}$
If x_s = Caterpie:	$y = b_3 + w_3 \cdot x_{cp}$
If x_s = Eevee:	$y = b_4 + w_4 \cdot x_{cp}$

↓

y

线性回归实例

Back to step 1: Redesign the Model

$$y = b + \sum w_i x_i$$

Linear model?

$$
\begin{aligned}
y = & \, b_1 \cdot \delta(x_s = \text{Pidgey}) \\
& + w_1 \cdot \delta(x_s = \text{Pidgey}) x_{cp} \\
& + b_2 \cdot \delta(x_s = \text{Weedle}) \\
& + w_2 \cdot \delta(x_s = \text{Weedle}) x_{cp} \\
& + b_3 \cdot \delta(x_s = \text{Caterpie}) \\
& + w_3 \cdot \delta(x_s = \text{Caterpie}) x_{cp} \\
& + b_4 \cdot \delta(x_s = \text{Eevee}) \\
& + w_4 \cdot \delta(x_s = \text{Eevee}) x_{cp}
\end{aligned}
$$

$\delta(x_s = \text{Pidgey})$

$$
\begin{cases}
=1 & \text{If } x_s = \text{Pidgey} \\
=0 & \text{otherwise}
\end{cases}
$$

If $x_s = \text{Pidgey}$

线性回归实例

Back to step 1: Redesign the Model

$$y = b + \sum w_i x_i$$

Linear model?

$y = b_1 \cdot \boxed{1}$

$+w_1 \cdot \boxed{1}$

$+b_2 \cdot \boxed{0}$

$+w_2 \cdot \boxed{0}$

$+b_3 \cdot \boxed{0}$

$+w_3 \cdot \boxed{0}$

$+b_4 \cdot \boxed{0}$

$+w_4 \cdot \boxed{0}$

$\delta(x_s = \text{Pidgey})$

$$\begin{cases} =1 & \text{If } x_s = \text{Pidgey} \\ =0 & \text{otherwise} \end{cases}$$

If $x_s = \text{Pidgey}$

$$y = b_1 + w_1 \cdot x_{cp}$$

线性回归实例

Back to step 1: Redesign the Model

$$
\begin{aligned}
y = {} & b_1 \cdot \delta(x_s = \text{Pidgey}) \\
& + w_1 \cdot \delta(x_s = \text{Pidgey}) x_{cp} \\
& + b_2 \cdot \delta(x_s = \text{Weedle}) \\
& + w_2 \cdot \delta(x_s = \text{Weedle}) x_{cp} \\
& + b_3 \cdot \delta(x_s = \text{Caterpie}) \\
& + w_3 \cdot \delta(x_s = \text{Caterpie}) x_{cp} \\
& + b_4 \cdot \delta(x_s = \text{Eevee}) \\
& + w_4 \cdot \delta(x_s = \text{Eevee}) x_{cp}
\end{aligned}
$$

$$y = \boxed{b} + \sum \boxed{w_i} \boxed{x_i}$$

Linear model?

$$\delta(x_s = \text{Pidgey})$$

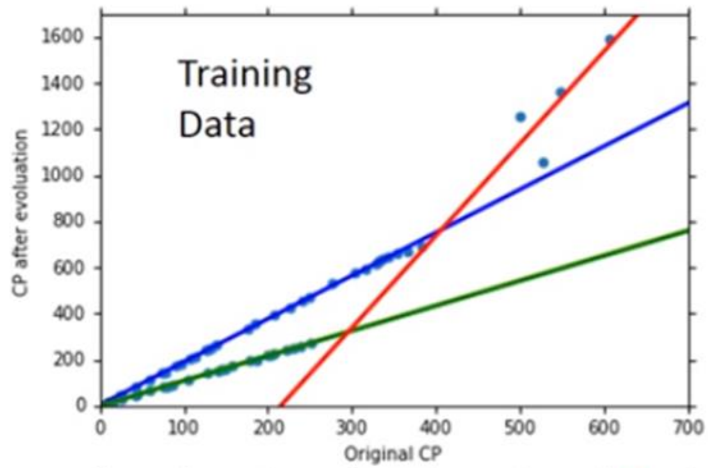
$$\begin{cases} =1 & \text{If } x_s = \text{Pidgey} \\ =0 & \text{otherwise} \end{cases}$$

If $x_s = \text{Pidgey}$

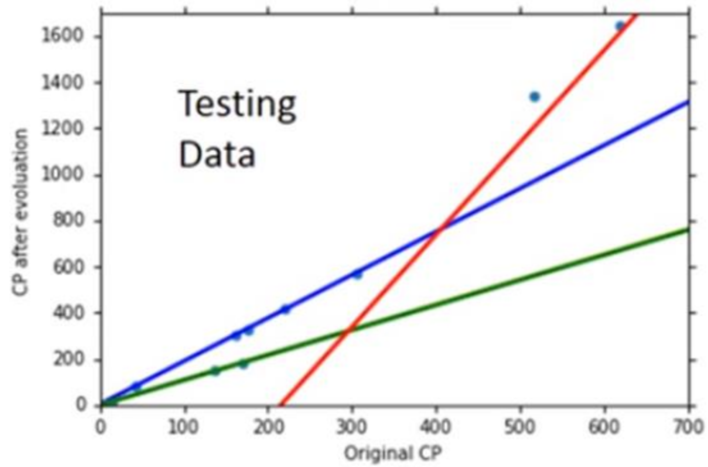
$$y = b_1 + w_1 \cdot x_{cp}$$

线性回归实例

Average error
= 3.8

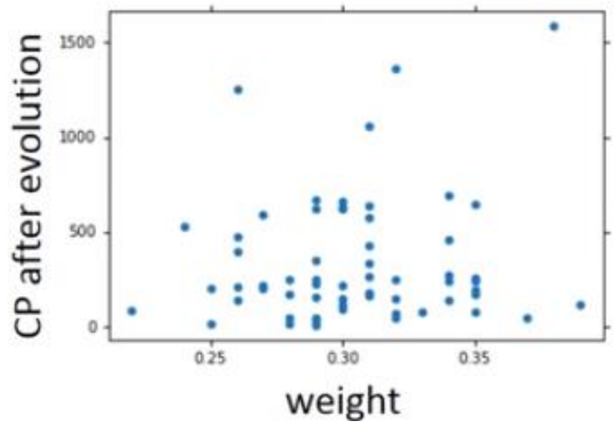


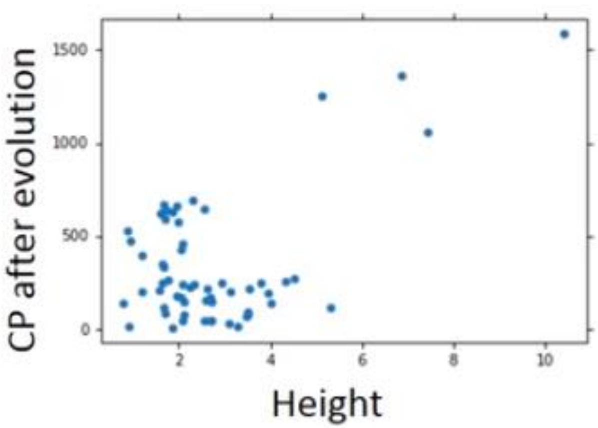
Average error
= 14.3

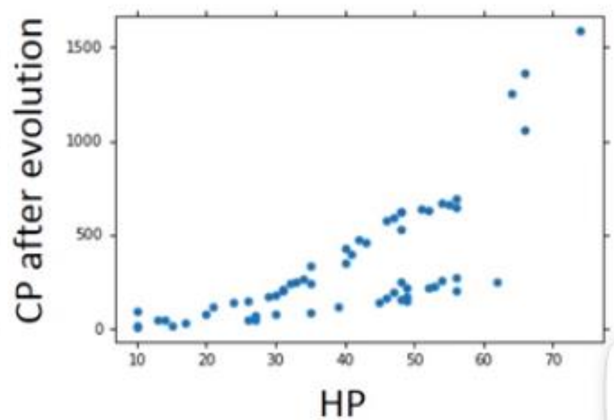


线性回归实例

Are there any other hidden factors?

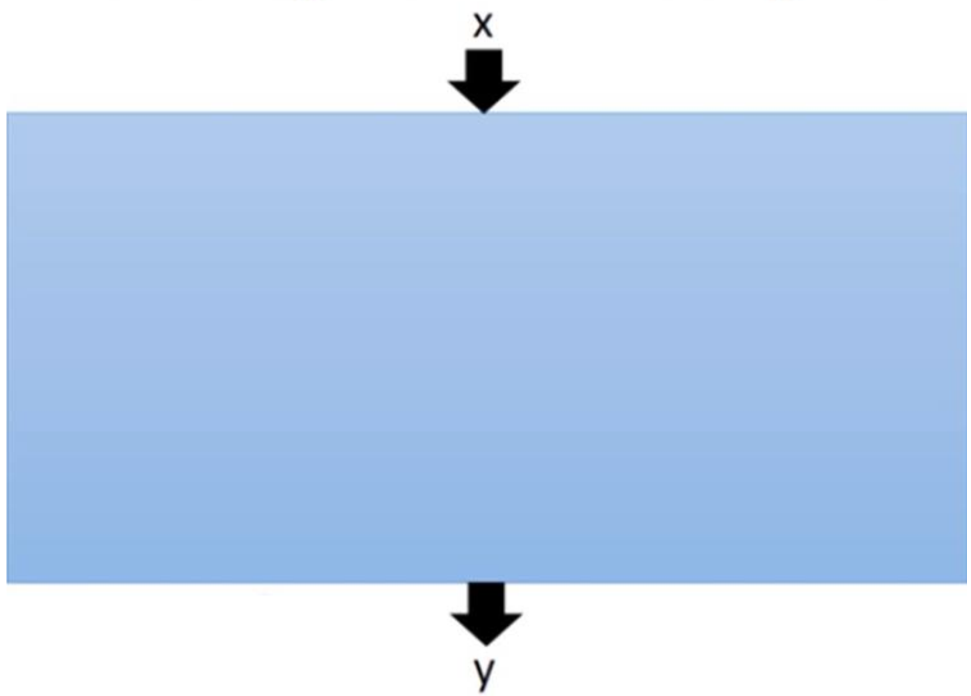






线性回归实例

Back to step 1:
Redesign the Model Again



线性回归实例

Back to step 1:
Redesign the Model Again

x

If x_s = Pidgey: $y' = b_1 + w_1 \cdot x_{cp} + w_5 \cdot (x_{cp})^2$

If x_s = Weedle: $y' = b_2 + w_2 \cdot x_{cp} + w_6 \cdot (x_{cp})^2$

If x_s = Caterpie: $y' = b_3 + w_4 \cdot x_{cp} + w_7 \cdot (x_{cp})^2$

If x_s = Eevee: $y' = b_4 + w_4 \cdot x_{cp} + w_8 \cdot (x_{cp})^2$

$$y = y' + w_9 \cdot x_{hp} + w_{10} \cdot (x_{hp})^2 + w_{11} \cdot x_h + w_{12} \cdot (x_h)^2 + w_{13} \cdot x_w + w_{14} \cdot (x_w)^2$$

y

Training Error = 1.9

Testing Error = 102.3

Overfitting!

线性回归实例

Back to step 2: Regularization

$$y = b + \sum w_i x_i$$

The functions with smaller w_i are better

$$L = \sum_n \left(\hat{y}^n - \left(b + \sum w_i x_i \right) \right)^2 \quad \boxed{+\lambda \sum (w_i)^2}$$

- Why smooth functions are preferred?

$$y = b + \sum w_i x_i$$

$+w_i \Delta x_i$ $\qquad$ $+\Delta x_i$

- If some noises corrupt input x_i when testing

A smoother function has less influence.

线性回归实例

Regularization

smoother

λ	Training	Testing
0	1.9	102.3
1	2.3	68.7
10	3.5	25.7
100	4.1	11.1
1000	5.6	12.8
10000	6.3	18.7
100000	8.5	26.8

- Training error: larger λ, considering the training error less
- We prefer smooth function, but don't be too smooth.