Regularization

KAUSHIK ROY

SRC JUMP

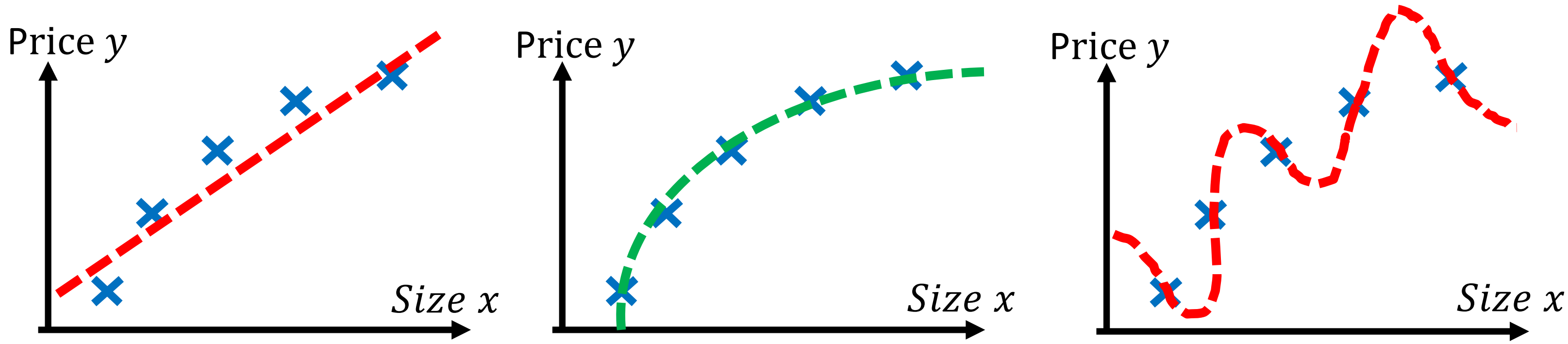
C-BRIC Center for Brain Inspired Computing

Motivation

➢ Linear regression & overfitting

E.g. Predict housing prices based on house size.

- Feature: $x = Size(feet^2)$
- Prediction: $y = Price$ ($)
- linear (polynomial) regression



$\theta_0 + \theta_1 x$ — Under fitting (high bias)

$\theta_0 + \theta_1 x + \theta_2 x^2$ — "Good" fitting

$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$ — Over fitting (high variance)

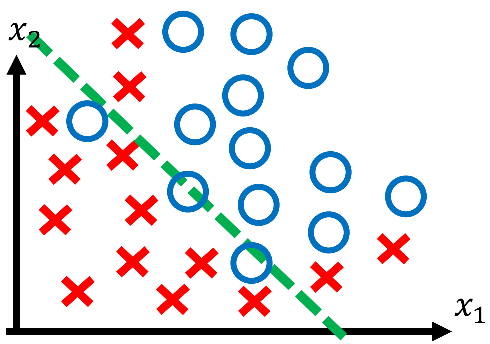
Overfitting: The learned hypothesis may fit the training set very well ($J(\theta) \approx 0$), but fail to generalize to new examples (predict prices on new examples).

Motivation

- Logistic regression overfitting

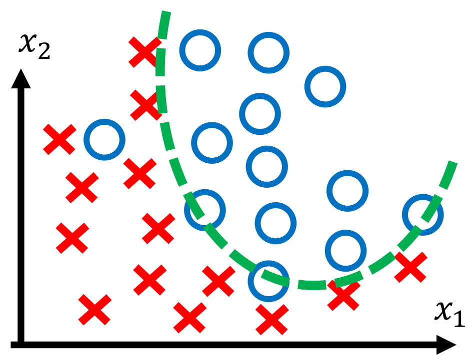
E.g. Housing sale prediction to a potential buyer

- Features: $x_1 = Size(feet^2)$; $x_2 = Price$ (\$)
- Prediction: $y = 1$, predict house will be sold
  $\mathrm{y} = 0$, predict house will not be sold
- Hypothesis: logistic regression



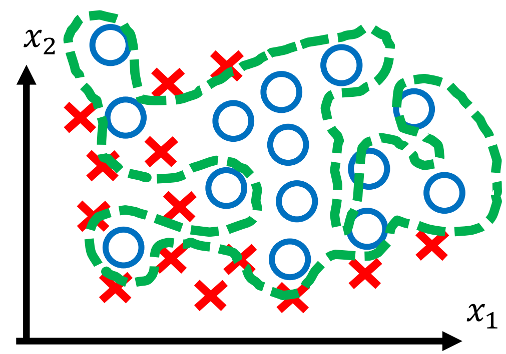
$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

Under fitting (high bias)



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2)$$

"Good" fitting



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \theta_6 x_1^3 x_2 + \cdots)$$

Over fitting (high variance)

Overfitting

➤ Addressing overfitting

Option 1:

Reduce number of features

- Manually select which features to keep.
- Model selection algorithm

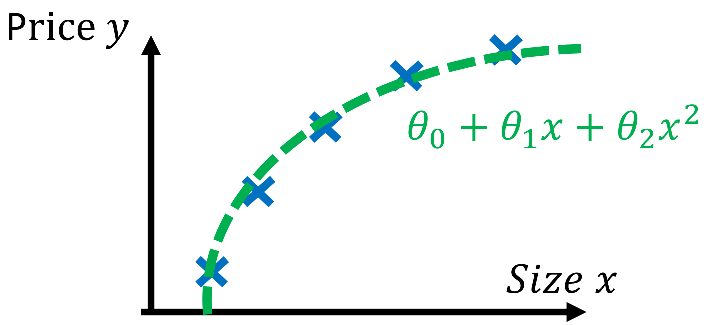
Option 2:

Regularization.

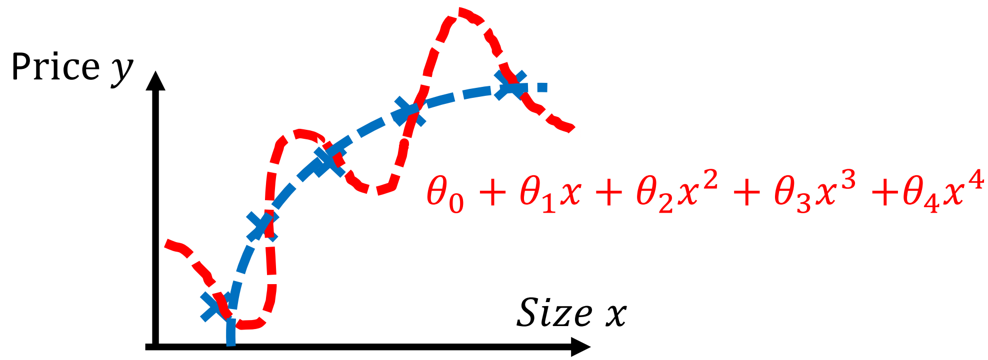
- Keep all the features, but reduce magnitude/values of parameters θ_j.
- Works well when we have a lot of features, each of which contributes a bit to predicting y.

Regularization

- Intuition



Good" fitting



Over fitting (high variance)

Suppose we penalize parameters θ_3 and θ_4 by adding two additional items $K_1\theta_3^2$and $K_2\theta_4^2$ to the overfitting hypothesis, in which $K_1 \gg 1$ and $K_2 \gg 1$ (e.g. $K_1 = 1000$ and $K_2 = 1000$)

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left[\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2 + K_1\theta_3^2 + K_2\theta_4^2\right]$$

In the learning process, to minimize the cost function $J(\theta)$, both θ_3 and θ_4 must be very small,

$\theta_3 \approx 0$ and $\theta_4 \approx 0$.

and the original overfitting hypothesis $\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$ becomes a less Overfitting -- close to $\theta_0 + \theta_1 x + \theta_2 x^2$.

Regularization

➢ Formulation

Smaller values for parameters $\theta_0, \theta_1, \theta_2, \ldots, \theta_n$ lead to:

- "Simpler" hypothesis
- Less prone to overfitting

Penalize all parameters but θ_0 by adding an additional term $\lambda \sum_{i=1}^{n} \theta_j^2$ to the cost function:

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} \left[\left(h_\theta\left(x^{(i)}\right) - y^{(i)} \right)^2 + \lambda \sum_{i=1}^{n} \theta_j^2 \right], \ (\lambda \text{ is regularization parameter})$$

Note, choosing too large a λ will lead to underfitting, because all parameters but θ_0 will be penalized and become too small -- the hypothesis becomes a constant value close to θ_0 e.g. hypothesis $\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4 + \ \ldots \approx \theta_0$, if θ_1, θ_1,…, $\theta_n \approx 0$.

How to choose a proper regularization parameter λ?

Regularized Learning Models

➢ Regularized linear regression

Recall regularized cost function for linear regression:

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left[\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2 + \lambda\sum_{i=1}^{n}\theta_j^2\right],$$

(λ is regularization parameter)

Regularized gradient descent algorithm for linear regression:

Repeat {

$$\theta_0 := \theta_0 - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)x_0^{(i)}$$

$$\theta_j := \theta_j - \alpha\left[\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)x_j^{(i)} + \frac{\lambda}{m}\theta_j\right]$$

$$= \theta_j\left(1 - \alpha\frac{\lambda}{m}\right) - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)x_j^{(i)}$$

}

Update rule of θ_0 remains identical as the original gradient descent formulation because θ_0 is not penalized. Update rules of other parameters are modified by adding a regularization term $\frac{\lambda}{m}\theta_j$ to the original gradient descent formula.

Regularized Learning Models

➢ Regularized logistic regression

Regularized cost function for logistic regression:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[log\left(h_\theta\left(x^{(i)}\right)\right) * y^{(i)} + log\left(1 - h_\theta\left(x^{(i)}\right)\right) * \left(1 - y^{(i)}\right)\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

$$(i = 1,2,\dots,m)$$

Regularized gradient descent algorithm for logistic regression:

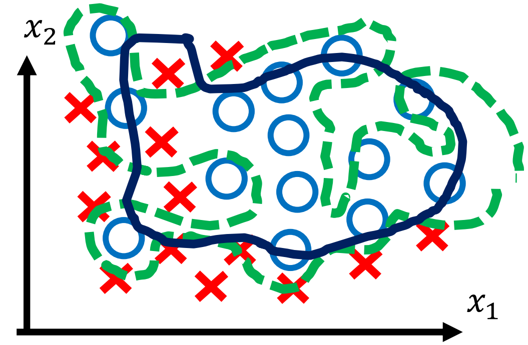
Repeat {

$$\theta_0 := \theta_0 - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)x_0^{(i)}$$

$$\theta_j := \theta_j - \alpha\left[\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)x_j^{(i)} + \frac{\lambda}{m}\theta_j\right]$$

$$= \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta)$$

}



Overfitting model is improved and the design boundary becomes smoother.