Lecture 6: Logistical classification and regression

Wengang Mao (Marine Technology)
Department of Mechanics and Maritime Sciences,
Chalmers University of Technology,
Goteborg, Sweden

1

Contents of this lecture

Simple Logistic classification

Multiclass classification

Logistic regression

- Decision boundary
- Regression: parameter estimation

2 This lecture is largely referred from Prof. Ng's course! 2023-04-11

2

Logistic Classification

Email: Spam / Not Spam?

Online advertisement: interest (Yes / No)?

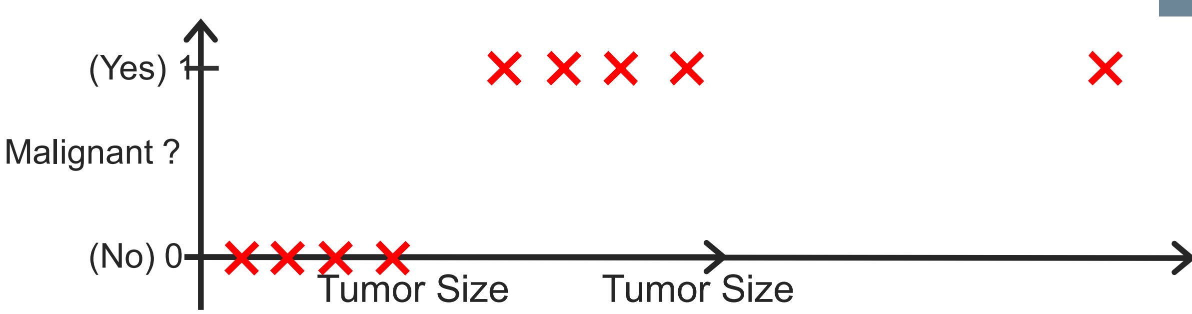
Tumor: Malignant / Benign ?

$y \in \{0, 1\}$ 0: “Negative Class” (e.g., benign tumor)
1: “Positive Class” (e.g., malignant tumor)

2023-04-11

3

Logistic Classification



Threshold classifier output $h_\theta(x)$ at 0.5:

If $h_\theta(x) \geq 0.5$, predict “y = 1”

If $h_\theta(x) < 0.5$, predict “y = 0”

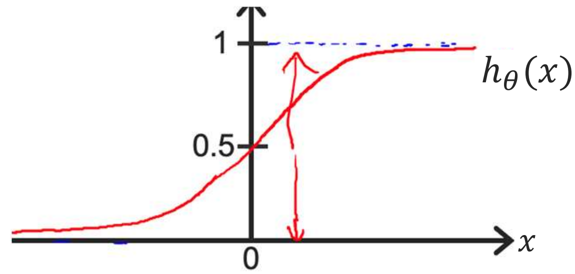
2023-04-11

4

Logistic Classification

Classification: y = 0 or 1

$h_\theta(x)$ can be > 1 or < 0



Logistic Regression: $0 \leq h_\theta(x) \leq 1$

2023-04-11

5

Multiclass classification

Email foldering/tagging: Work, Friends, Family, Hobby

y=1 y=2 y=3 y=4

Medical diagrams: Not ill, Cold, Flu

y=1 2 3

Weather: Sunny, Cloudy, Rain, Snow

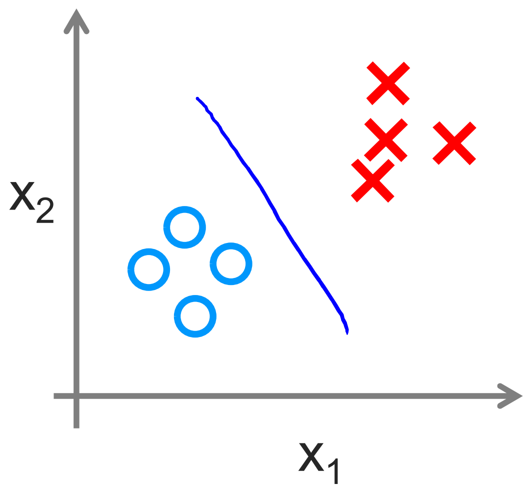
y=1 2 3 4

0 1 2 3

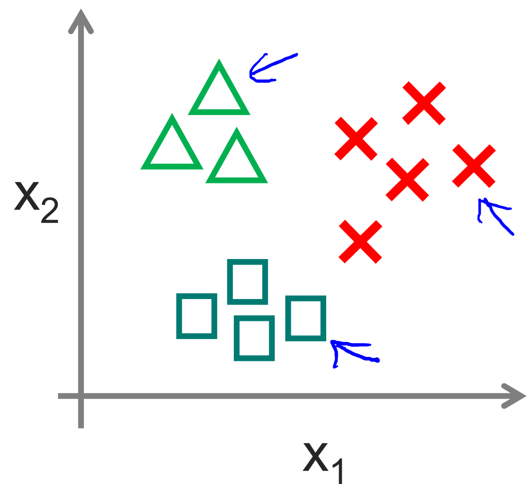
2023-04-11

6

Binary classification:

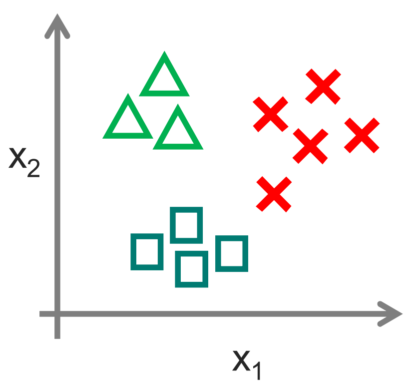


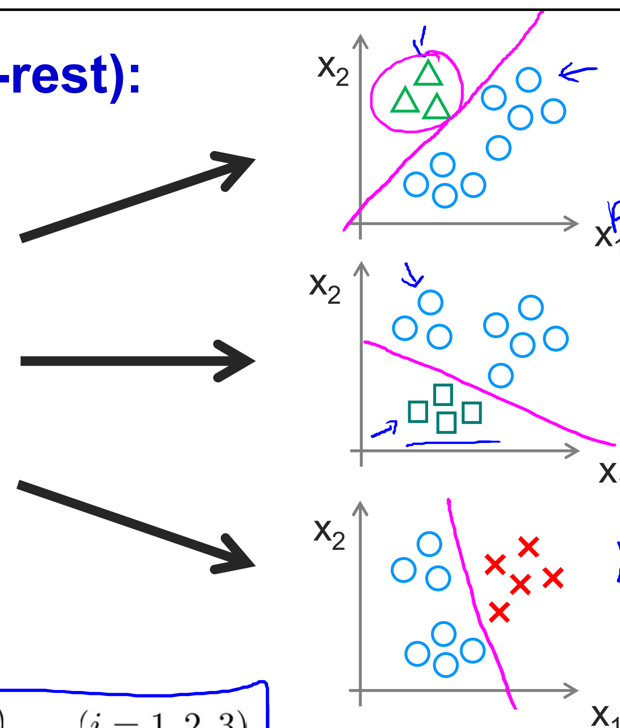
Multi-class classification:



7

One-vs-all (one-vs-rest):





$h_\theta^{(1)}(x)$

$P(y=1|x;\theta)$

$h_\theta^{(2)}(x)$

$h_\theta^{(3)}(x)$

Class 1: △

Class 2: □

Class 3: ✕

$$\boxed{h_\theta^{(i)}(x) = P(y=i|x;\theta) \qquad (i=1,2,3)}$$

8

Multiclass classification: One-vs-all

- Train a logistic regression classifier $h_\theta^{(i)}(x)$ for each class i to predict the probability that $y=i$.
- On a new input x, to make a prediction, pick the class i that maximizes

$$\max_i h_\theta^{(i)}(x)$$

2023-04-11

9

Logistic regression

❖Decision boundary

❖Regression: parameter estimation

2023-04-11

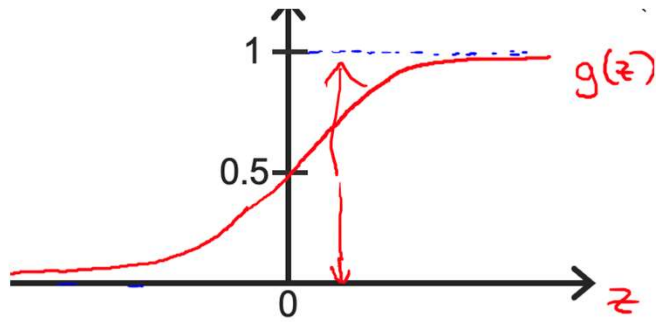
10

Logistic Regression Model

Want $0 \leq h_\theta(x) \leq 1$

$$h_\theta(x) = g(\ \theta^T x\ )$$

$$g(z) = \frac{1}{1+e^{-z}}$$



Transfer function: Sigmoid function; Logistic function

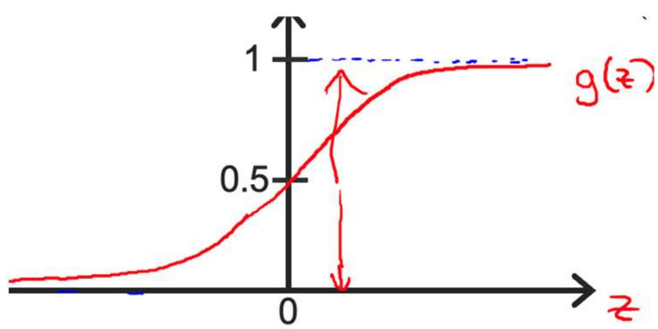
2023-04-11

11

Logistic regression

$$h_\theta(x) = g(\theta^T x)$$

$$g(z) = \frac{1}{1+e^{-z}}$$



Suppose predict "$y = 1$" if $h_\theta(x) \geq 0.5$

predict "$y = 0$" if $h_\theta(x) < 0.5$

2023-04-11

12

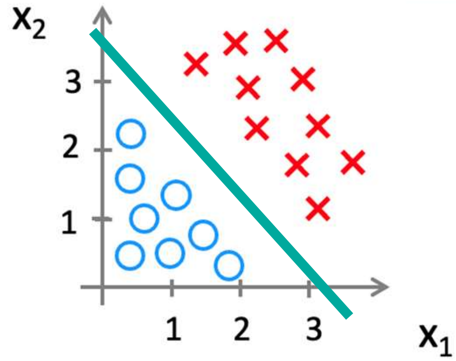
Decision Boundary

$$h_\theta(x) = g(\theta^T x)$$

$$g(z) = \frac{1}{1+e^{-z}}$$

Suppose predict "$y = 1$" if $h_\theta(x) \geq 0.5$

predict "$y = 0$" if $h_\theta(x) < 0.5$



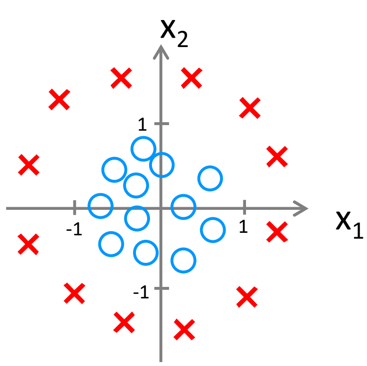
$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

Predict "$y = 1$" if $-3 + x_1 + x_2 \geq 0$

2023-04-11

13

Non-linear decision boundaries



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

Predict "$y = 1$" if $-1 + x_1^2 + x_2^2 \geq 0$

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

2023-04-11

14

Regression: parameter estimation

Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots, (x^{(m)}, y^{(m)})\}$

m examples $\quad x \in \begin{bmatrix} x_0 \\ x_1 \\ \dots \\ x_n \end{bmatrix} \quad x_0 = 1, y \in \{0, 1\}$

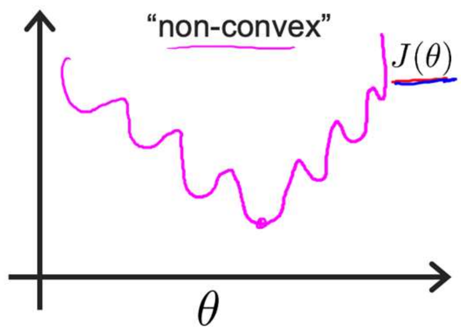
$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

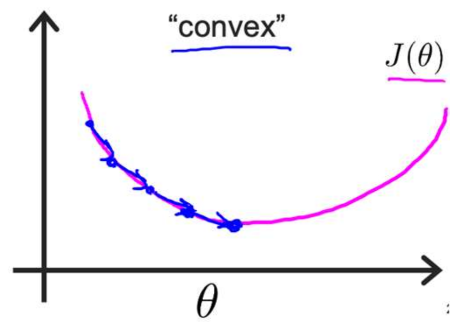
How to choose parameters θ?

Logistical regression: cost function (1)

$$\mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)}) = \frac{1}{2}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

Linear regression: $J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{2}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$



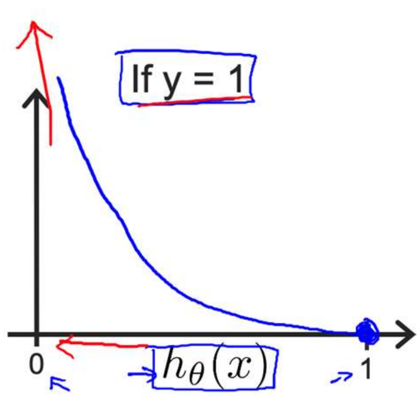


2023-04-11

17

Logistic regression: cost function (2)

$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$



Cost $= 0$ if $y = 1, h_\theta(x) = 1$
But as $h_\theta(x) \to 0$
$Cost \to \infty$

Captures intuition that if $h_\theta(x) = 0$, (predict $P(y = 1|x;\theta) = 0$), but $y = 1$, we'll penalize learning algorithm by a very large cost.

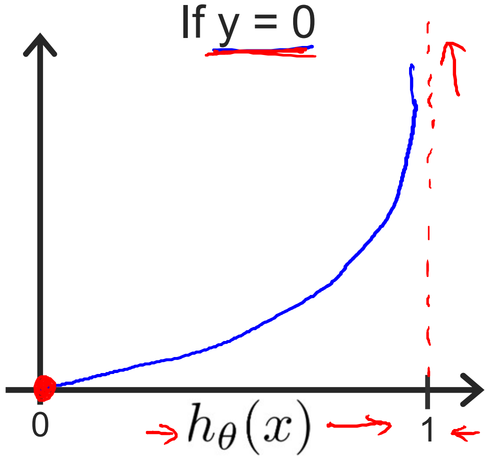
2023-04-11

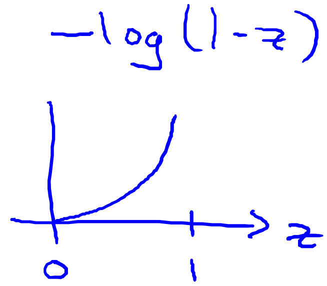
18

Logistic regression: cost function (3)

$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ \boxed{-\log(1 - h_\theta(x))} & \text{if } y = 0 \end{cases}$$

If y = 0





2023-04-11

19

Logistic regression: parameter estim.

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

$$= -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log\left(1 - h_\theta(x^{(i)})\right)\right]$$

To fit parameters $\boldsymbol{\theta}$:

$$\min_\theta J(\theta)$$

To make a prediction given new x :

Output $h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$

2023-04-11

20

Estimation method: Gradient Descent (1)

$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log (1 - h_\theta(x^{(i)}))]$$

Want $\min_\theta J(\theta)$:

Repeat $\{$

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

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

(simultaneously update all θ_j)

$\}$

2023-04-11

21

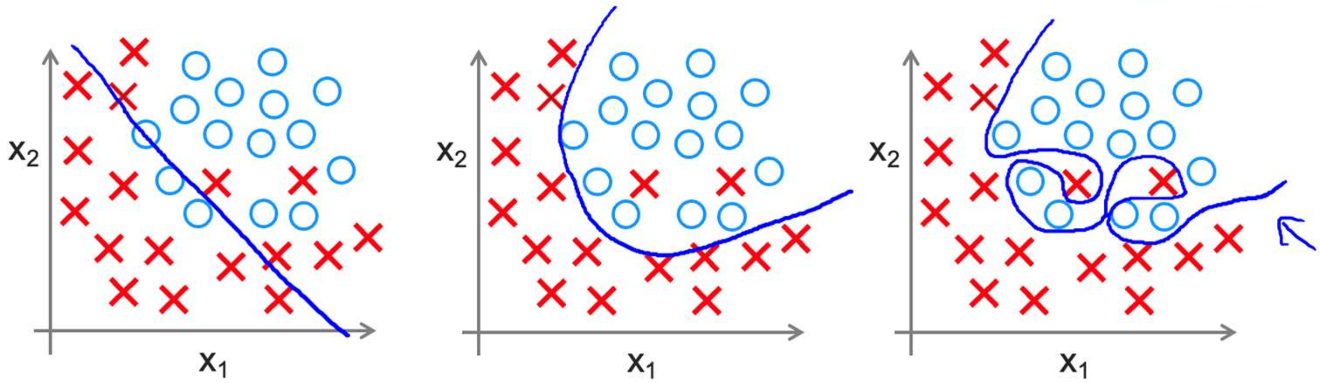
Regularization

To avoid problem of overfitting

2023-04-11

22

Example: Logistic regression



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

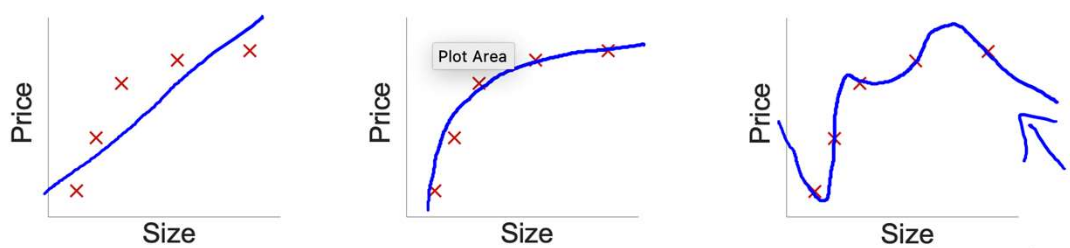
(g = sigmoid function)

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

$g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2$
$+\theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2$
$+\theta_5 x_1^2 x_2^3 + \theta_6 x_1^3 x_2 + \dots)$

23

Example: Linear regression (housing prices)



$\rightarrow \theta_0 + \theta_1 x$

"Underfit" "High bias"

$\rightarrow \theta_0 + \theta_1 x + \theta_2 x^2$

"Just right"

$\rightarrow \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

"Overfit" "High variance"

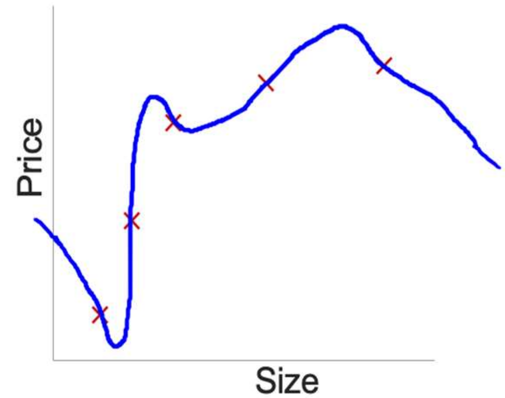
Overfitting: If we have too many features, the learned hypothesis may fit the training set very well ($J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 \approx 0$), but fail to generalize to new examples (predict prices on new examples).

24

Overfitting issues by an example

An example: we would like to predict the house price y in terms of different features $x_1, x_2, \ldots, x_{100}$.

$x_1 =$ size of house
$x_2 =$ no. of bedrooms
$x_3 =$ no. of floors
$x_4 =$ age of house
$x_5 =$ average income in neighborhood
$x_6 =$ kitchen size
$\vdots$
x_{100}



2023-04-11

25

Reducing overfitting:

Options:

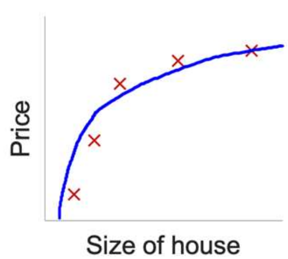
1. Reduce number of features.
   - Manually select which features to keep.
   - Model selection algorithm.
2. Regularization.
   - Keep all the features, but reduce magnitude/values of parameters θ_j.
   - Works well when we have a lot of features, each of which contributes a bit to predicting y.

2023-04-11

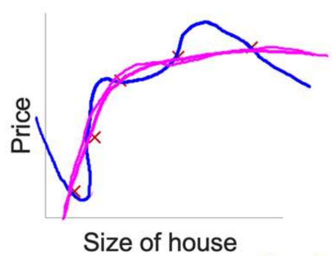
26

Regularization: Cost function

Ideas explanation for the regularization



$\theta_0 + \theta_1 x + \theta_2 x^2$



$\theta_0 + \theta_1 x + \theta_2 x^2 + \cancel{\theta_3 x^3} + \cancel{\theta_4 x^4}$

Suppose we penalize and make θ_3, θ_4 really small.

$$\rightarrow \min_\theta \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 + 1000\,\theta_3^2 + 1000\,\theta_4^2$$

$\theta_3 \approx 0 \qquad \theta_4 \approx 0$

Regularization in mathematical form

Small values for parameters: $\boldsymbol{\theta_1, \theta_2, \dots, \theta_{100}}$
- "Simpler" hypothesis
- Less prone to overfitting

Housing:
- Features: $\boldsymbol{x_1, x_2, \dots, x_{100}}$
- Parameters: $\boldsymbol{\theta_1, \theta_2, \dots, \theta_{100}}$

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

$\cancel{\theta_0}$ $\theta_1, \theta_2, \theta_3, \dots, \theta_{100}$ $\cancel{\theta_0}$

2023-04-11

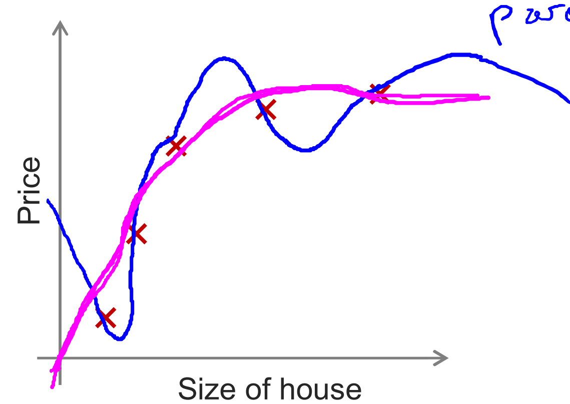
29

Regularization.

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

regularization parameter

$$\min_\theta J(\theta)$$



2023-04-11

30

In regularized linear regression, we choose θ to minimize

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

What if λ is set to an extremely large value (perhaps for too large for our problem, say $\lambda = 10^{10}$)?

- Algorithm works fine; setting λ to be very large can't hurt it
- Algorithm fails to eliminate overfitting.
- Algorithm results in underfitting. (Fails to fit even training data well).
- Gradient descent will fail to converge.

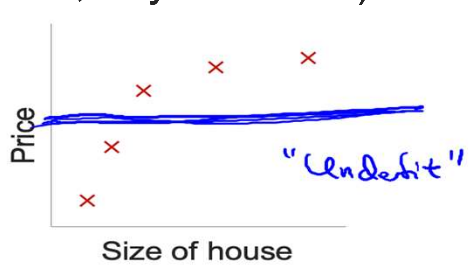
2023-04-11

31

In regularized linear regression, we choose $\boldsymbol{\theta}$ to minimize

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

What if λ is set to an extremely large value (perhaps for too large for our problem, say $\lambda = 10^{10}$)?



$h_\theta(x)$

$$\theta_0 + \cancel{\theta_1 x} + \cancel{\theta_2 x^2} + \cancel{\theta_3 x^3} + \cancel{\theta_4 x^4}$$

2023-04-11

32

Method 1: mathematical normal equation

$$X = \begin{bmatrix} (x^{(1)})^T \\ \vdots \\ (x^{(m)})^T \end{bmatrix} \qquad y = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(m)} \end{bmatrix} \quad \mathbb{R}^m$$

$m \times (n+1)$

$$\min_\theta J(\theta)$$

$\frac{\partial}{\partial \theta_j} J(\theta) \overset{\text{set}}{=} 0$

$$\theta = \left(X^T X + \lambda \underbrace{\begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & & & \ddots & \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}}_{(n+1)\times(n+1)} \right)^{-1} X^T y$$

E.g. $n=2$ $\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

2023-04-11

33

Method 2: Gradient descent

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \underbrace{\sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_0^{(i)}}_{\frac{\partial}{\partial \theta_0} J(\theta)}$$

Repeat the following steps:

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

$(j = \cancel{0}, 1, 2, 3, \ldots, n)$

}

$$\theta_j := \boxed{\theta_j (1 - \alpha \frac{\lambda}{m})} - \boxed{\alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}} \rightarrow J(\theta)$$

$1 - \alpha \frac{\lambda}{m} < 1$ $\quad 0.99 \quad \theta_j \times 0.99 \quad \boxed{\theta_j^2}$

2023-04-11

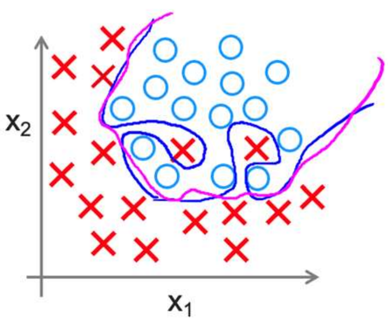
34

Regularized logistic regression

2023-04-11

35

Regularized logistic regression



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

New cost function with “penalty/ regularization” term:

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

$\boxed{\theta_1, \theta_2, \dots, \theta_n}$

2023-04-11

36

Gradient descent for logistical regression

Set up the initial value for the gradient analysis

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

Repeat the following iterations:

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

$$(j = \cancel{0}, 1, 2, 3, \ldots, n)$$

$\theta_1, \ldots, \theta_n$

$\frac{\partial}{\partial \theta_j} J(\theta)$

$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$

2023-04-11

37

Advanced optimization

```
function [jVal, gradient] = costFunction(theta)
```

`jVal = [` code to compute $J(\theta)$ `];`

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

`gradient(1) = [`code to compute $\frac{\partial}{\partial\theta_0} J(\theta)$ `];`

$$\frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_0^{(i)}$$

`gradient(2) = [`code to compute $\frac{\partial}{\partial\theta_1} J(\theta)$ `];`

$$\frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_1^{(i)} - \frac{\lambda}{m}\theta_1$$

`gradient(3) = [`code to compute $\frac{\partial}{\partial\theta_2} J(\theta)$ `];`

$$\frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_2^{(i)} - \frac{\lambda}{m}\theta_2$$

⋮

`gradient(n+1) = [`code to compute $\frac{\partial}{\partial\theta_n} J(\theta)$ `];`

2023-04-11

38

AVANCEZ

1829

CHALMERS

UNIVERSITY OF TECHNOLOGY

39