Logistic regression

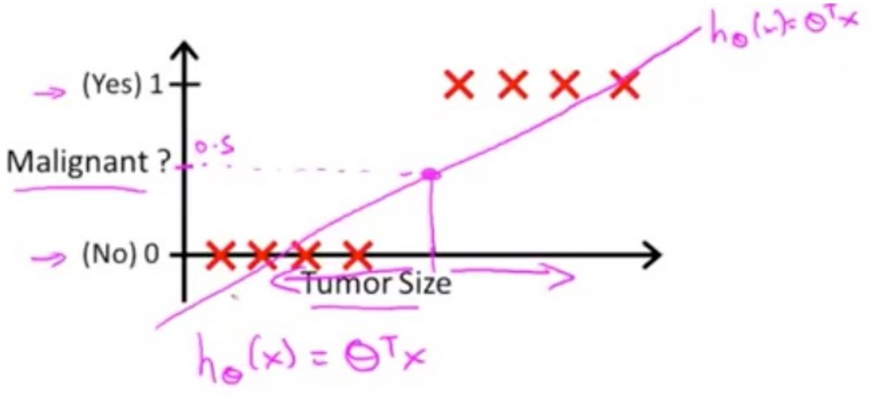
Dr. Huiping Cao

Outline

- Motivation
- Parameter fitting
- Cost function

Motivation (1)

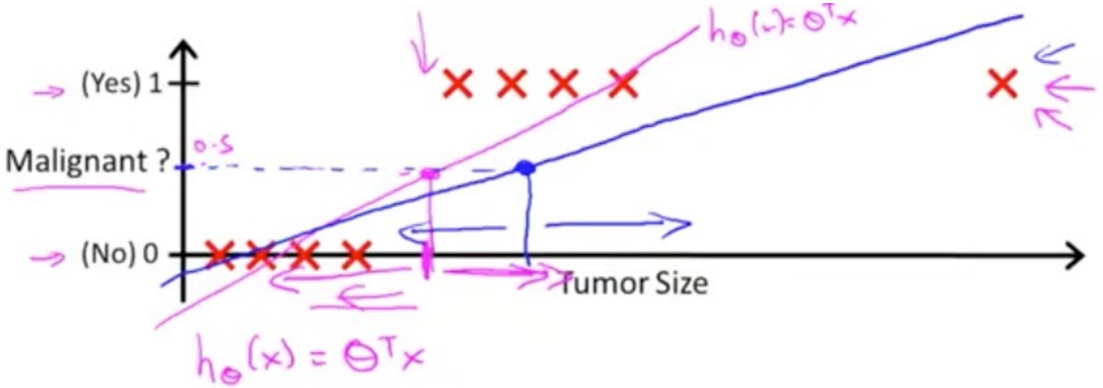
- Given {x$^{(1)}$, …, x$^{(n)}$} and corresponding {y $^{(1)}$, …, y$^{(n)}$}
- Can utilize linear regression: $h_\theta(x) = \theta^T x$



- The threshold classifier output
  - If $h_\theta(x) \geq 0.5, predict\ y = 1$
  - If $h_\theta(x) < 0.5, predict\ y = 0$

Motivation (2)

- Issue with this approach. Example (add one extra non-critical point)



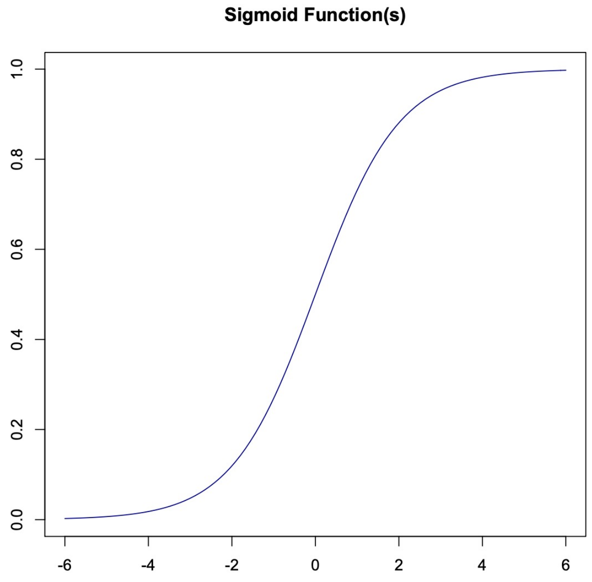
- If we run linear regression, the line will be different. Everything to the right of a point, we predictive it to be positive.
- Directly applying linear regression to do classification generally does not work well.

Logistic regression - Hypothesis

- Logistic regression, generate output in [0, 1]: $0 \le h_\theta(x) \le 1$
- Define $h_\theta(x)$ to be $g(\theta^T x)$
- Utilize a logistic function (or sigmoid function) $g(z) = \frac{e^z}{1+e^z}$ (or, rewritten as $\frac{1}{1+e^{-z}}$), get the hypothesis

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

- Sigmoid function

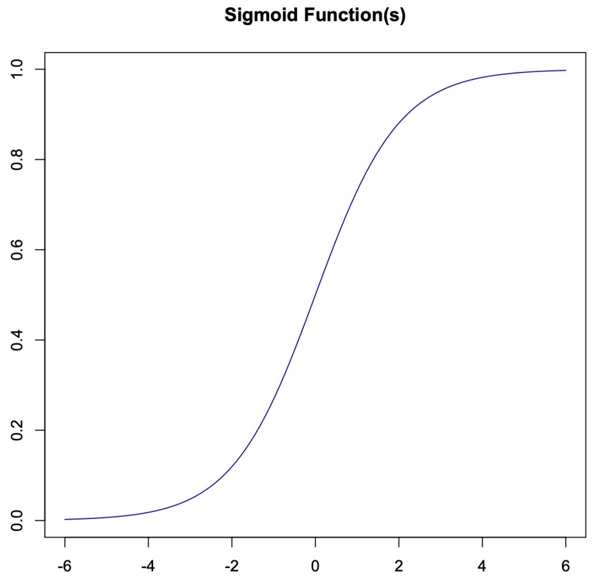


Logistic regression - Hypothesis (cont.)

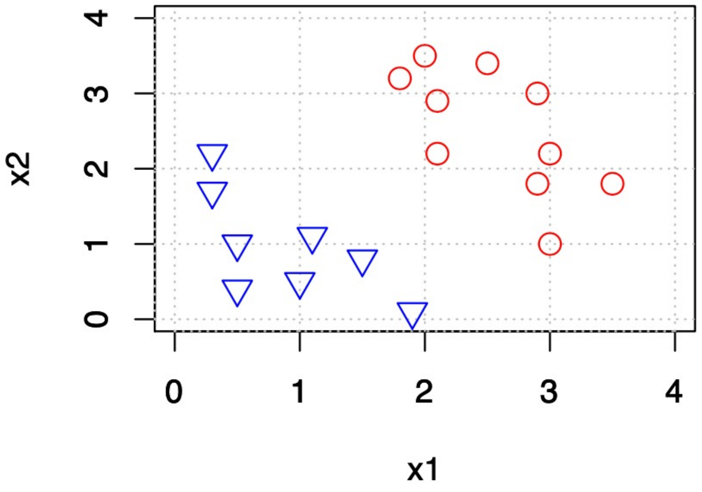
- **Interpretation** of hypothesis output
  - $h_\theta(x)$: for the input x, the estimated probability that y = 1.
- **Example**: if y = 1 means a tumor is malignant,
  - $h_\theta(x) = 0.7$ tells that 70% chance the tumor is malignant.
- $\mathbf{h_\theta(x) = P(y = 1|x;\theta)}$: Probability that y = 1 given x, parameterized by θ.
  - $P(y = 0|x;\theta) + P(y = 1|x;\theta) = 1$
  - $P(y = 0|x;\theta) = 1 - P(y = 1|x;\theta)$

Logistic regression - Decision boundary

- When will we predict $y = 0$ or $y = 1$?
- Suppose that
  - we predict $y = 1$ if $h_\theta(x) \geq 0.5$
  - we predict $y = 0$ if $h_\theta(x) < 0.5$
- Re-examine the **sigmoid function**
  - when $z \geq 0$, $g(z) \geq 0.5$; in this case, we predict $y = 1$
    Equivalently, when $\theta^T x \geq 0$, $h_\theta(x) = g(\theta^T x) \geq 0.5$
  - when $z < 0$, $g(z) < 0.5$; in this case, we predict $y = -1$
    Equivalently, when $\theta^T x < 0$, $h_\theta(x) = g(\theta^T x) < 0.5$



Decision boundary (cont.)



- Training data: red circle (class 1), blue triangle (class -1)

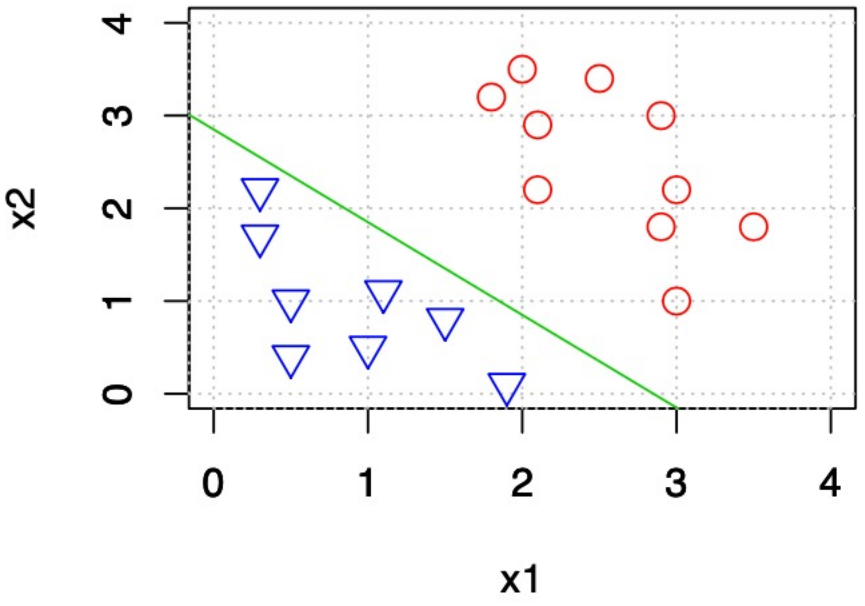
- $h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$

- Suppose that we have the hypothesis $\theta = \begin{pmatrix} -3 \\ 1 \\ 1 \end{pmatrix}$

- How do we make prediction?
  - Predict y = 1 if $-3 + x_1 + x_2 \geq 0$ (i.e., $x_1 + x_2 \geq 3$)

Decision boundary (cont.)

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



- If we draw a line $x_1 + x_2 = 3$, the points above this line are predicted to be y = 1; the points below this line are predicted to be y = 0.
- Line $x_1 + x_2 = 3$ is called decision boundary, which separates the regions for prediction of y = 0 and y = 1.

Fit the parameters θ (1)

- Cost function for linear regression

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

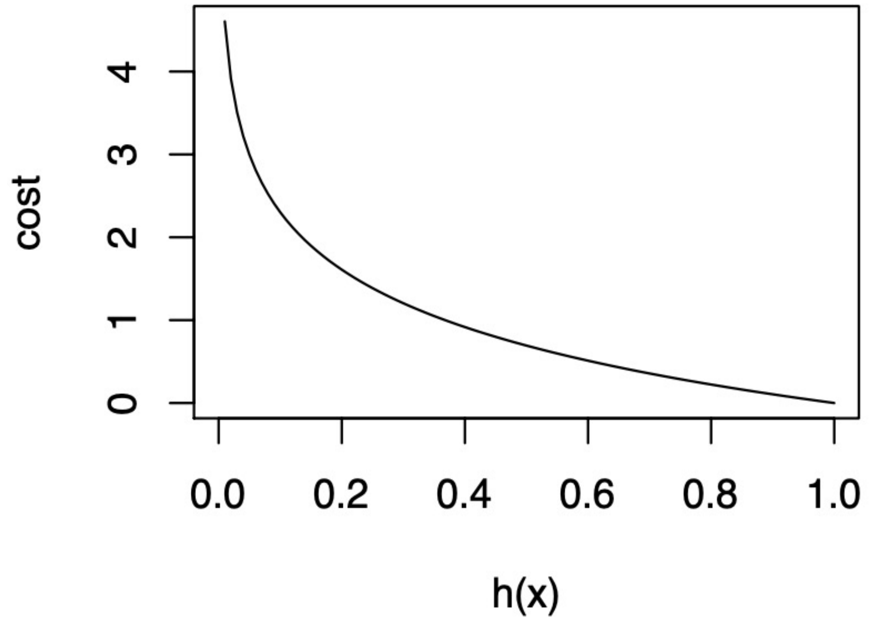
where $cost(h_\theta(x), y) = \frac{1}{2}(h_\theta(x) - y)^2$

Logistic regression - Cost function (cont.)

- The cost function for logistic regression is defined as follows:

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

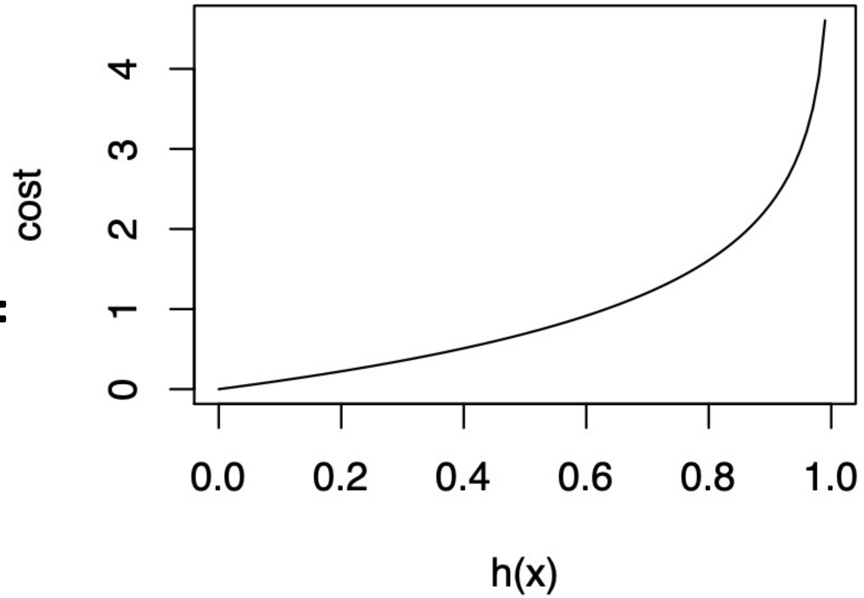
- What does this cost function look like when y = 1?
- When y = 1, this cost function has many good properties.
  - If y = 1 and $h_\theta(x) = 1$, then Cost =0.
    If y = 1 and $h_\theta(x) \rightarrow 0$, Cost $\rightarrow \infty$.
    Captures **intuition**: if y = 1 (actual class) and $h_\theta(x) = 0$ (predict $P(y = 1|x;\theta) = 0$; absolutely impossible), we'll penalize the learning algorithm by a very large cost.



Logistic regression - Cost function (cont.)

- The cost function for logistic regression is defined as follows:

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



- What does this cost function look **like when y = 0?**
- When y = 0, this cost function has many good properties.
  - If y = 0 and $h_\theta(x) = 0$, then Cost =0.
    If y = 0 and $h_\theta(x) \rightarrow 1$, Cost $\rightarrow \infty$.
    Captures **intuition**: if y = 0 (actual class) and $h_\theta(x) = 1$ (predict $P(y = 1|x;\theta) = 1$; absolutely impossible)

Cost function - rewriting

- Cost function

$$J(\theta) = \frac{1}{n}\sum_{i=1}^{n} cost(h_\theta(x), y)$$

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

- The cost function can be rewritten as

$$\text{cost}(h_\theta(\mathbf{x}), y) = -y\log(h_\theta(\mathbf{x})) - (1-y)\log(1 - h_\theta(\mathbf{x}))$$

Cost function - rewriting (cont.)

- Cost function

$$
\begin{aligned}
J(\theta) &= \tfrac{1}{n}\textstyle\sum_{i=1}^{n} \mathrm{cost}(h_\theta(\mathbf{x}^{(i)}), y^{(i)}) \\
&= -\tfrac{1}{n}(\textstyle\sum_{i=1}^{n} y^{(i)} \log(h_\theta(\mathbf{x}^{(i)})) + (1 - y^{(i)}) \log(1 - h_\theta(\mathbf{x}^{(i)})))
\end{aligned}
$$

- Goal: $min_\theta J(\theta)$
- Algorithm:

Repeat{

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

(simultaneously update all θ_js)

}

Gradient Descent

Repeat{

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

(simultaneously update all θ_js)

}

- Since $\frac{\partial}{\partial \theta_j} J(\theta) = \frac{1}{n} \sum_{i=1}^{n} (h_\theta(\mathbf{x}^{(i)}) - y^{(i)})\mathbf{x}^{(i)}$, we get
- Repeat {

$$\theta_j = \theta_j - \alpha \sum_{i=1}^{n} (h_\theta(\mathbf{x}^{(i)}) - y^{(i)})\mathbf{x}^{(i)}$$

(simultaneously update all θ_j)

}
- Algorithm looks identical to linear regression!
- The difference is the definition of $h_\theta(\mathbf{x}^{(i)})$.

To make a prediction

- To make a prediction given new **x**:
- Output

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

- The meaning is p(y = 1|x; θ)

Code example

```
>>> from sklearn.datasets import load_iris
>>> from sklearn.linear_model import LogisticRegression

>>> X, y = load_iris(return_X_y=True)

>>> clf = LogisticRegression(random_state=0).fit(X, y)
>>> clf.predict(X[:2, :])
array([0, 0])

>>> clf.predict_proba(X[:2, :])
array([[9.8...e-01, 1.8...e-02, 1.4...e-08], [9.7...e-01, 2.8...e-02, ...e-08]])
```

References

- Logistic Regression: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html